Strategy Synthesis for Linear Arithmetic Games

Azadeh Farzan[1]   Zachary Kincaid[2]

[1]University of Toronto

[2]Princeton University

January 12, 2018

Many problems in formal methods can be formulated as ***logical games***.

Many problems in formal methods can be formulated as ***logical games***.

- Functional synthesis
  - $\forall$ inputs, $\exists$ output s.t. specification holds

Many problems in formal methods can be formulated as ***logical games***.

- Functional synthesis
  - $\forall$ inputs, $\exists$ output s.t. specification holds
- Reactive synthesis
  - $\forall$ event_1, $\exists$ response_1 s.t. avoid bad state *and*
    $\forall$ event_2, $\exists$ response_2 s.t. avoid bad state *and*
    $\ldots$

Many problems in formal methods can be formulated as ***logical games***.

- Functional synthesis
  - $\forall$ inputs, $\exists$ output s.t. specification holds
- Reactive synthesis
  - $\forall$ event_1, $\exists$ response_1 s.t. avoid bad state *and*
    $\forall$ event_2, $\exists$ response_2 s.t. avoid bad state *and*
    $\ldots$

This paper:

> Algorithms for synthesizing winning strategies for satisfiability and reachability games in the theory of linear arithmetic.

Satisfiability games

Game interpretation

$$\varphi \triangleq \underbrace{\exists w. \forall x. \exists y. \forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \lor 2w < y) \land (z < y \lor x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false

Game interpretation

$$\varphi \triangleq \underbrace{\exists w. \forall x. \exists y. \forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \lor 2w < y) \land (z < y \lor x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false
- A play of this game: SAT and UNSAT take turns picking elements of $\mathbb{Q}$.

$$[\qquad\qquad\qquad\qquad]$$

Game interpretation

$$\varphi \triangleq \underbrace{\exists w. \forall x. \exists y. \forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \vee 2w < y) \wedge (z < y \vee x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false
- A play of this game: SAT and UNSAT take turns picking elements of $\mathbb{Q}$.

$$[w \mapsto 1; \qquad\qquad\qquad\qquad]$$

Game interpretation

$$\varphi \triangleq \underbrace{\exists w. \forall x. \exists y. \forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \vee 2w < y) \wedge (z < y \vee x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false
- A play of this game: SAT and UNSAT take turns picking elements of $\mathbb{Q}$.

$$[w \mapsto 1; x \mapsto \frac{2}{3}; \qquad]$$

Game interpretation

$$\varphi \triangleq \underbrace{\exists w. \forall x. \exists y. \forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \vee 2w < y) \wedge (z < y \vee x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false
- A play of this game: SAT and UNSAT take turns picking elements of $\mathbb{Q}$.

$$[w \mapsto 1; x \mapsto \frac{2}{3}; y \mapsto -1; \qquad]$$

Game interpretation

$$\varphi \triangleq \underbrace{\exists w. \forall x. \exists y. \forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \vee 2w < y) \wedge (z < y \vee x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false
- A play of this game: SAT and UNSAT take turns picking elements of $\mathbb{Q}$.

$$[w \mapsto 1; x \mapsto \frac{2}{3}; y \mapsto -1; z \mapsto 1]$$

Game interpretation

$$\varphi \triangleq \underbrace{\exists w. \forall x. \exists y. \forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \vee 2w < y) \wedge (z < y \vee x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false
- A play of this game: SAT and UNSAT take turns picking elements of $\mathbb{Q}$.

$$[w \mapsto 1; x \mapsto \frac{2}{3}; y \mapsto -1; z \mapsto 1]$$

The SAT player wins if the corresponding structure is a model of the matrix.

Game interpretation

$$\varphi \triangleq \underbrace{\exists w.\forall x.\exists y.\forall z.}_{\textit{quantifier prefix}} \underbrace{(y < 1 \vee 2w < y) \wedge (z < y \vee x < z)}_{\textit{matrix}}$$

- Two players: SAT and UNSAT
  - SAT wants to make the formula true
  - UNSAT wants to make the formula false
- A play of this game: SAT and UNSAT take turns picking elements of $\mathbb{Q}$.

$$[w \mapsto 1; x \mapsto \frac{2}{3}; y \mapsto -1; z \mapsto 1]$$

  The SAT player wins if the corresponding structure is a model of the matrix.
- φ is satisfiable $\iff$ SAT has a winning strategy

$$\forall x.\forall y.\exists lub.\ \underbrace{lub \geq x \wedge lub \geq y}_{\text{upper bound}} \wedge \underbrace{[\forall ub.(ub \geq x \wedge ub \geq y) \implies ub \geq lub]}_{\text{least}}$$

$$\forall x. \forall y. \exists \mathit{lub}. \forall \mathit{ub}. \mathit{lub} \geq x \land \mathit{lub} \geq y \land [(\mathit{ub} \geq x \land \mathit{ub} \geq y) \implies \mathit{ub} \geq \mathit{lub}]$$

$$\forall x. \forall y. \exists \mathit{lub}. \forall \mathit{ub}. \mathit{lub} \geq x \wedge \mathit{lub} \geq y \wedge [(\mathit{ub} \geq x \wedge \mathit{ub} \geq y) \implies \mathit{ub} \geq \mathit{lub}]$$

Winning strategy:

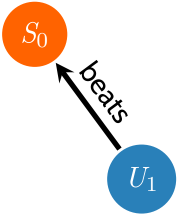
$$\mathit{lub}(x, y) = \textbf{if } x \geq y \textbf{ then } x \textbf{ else } y$$

SimSat: SAT via mutual strategy improvement

[Farzan & Kincaid, IJCAI 2016]

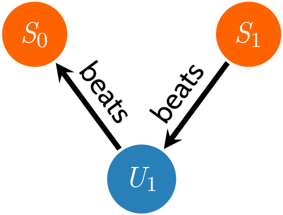
SimSat: SAT via mutual strategy improvement

[Farzan & Kincaid, IJCAI 2016]



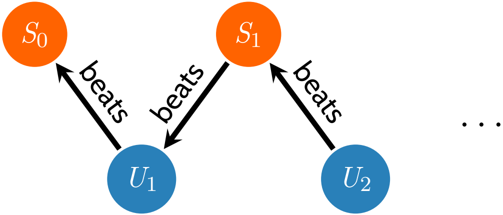
SimSat: SAT via mutual strategy improvement

[Farzan & Kincaid, IJCAI 2016]



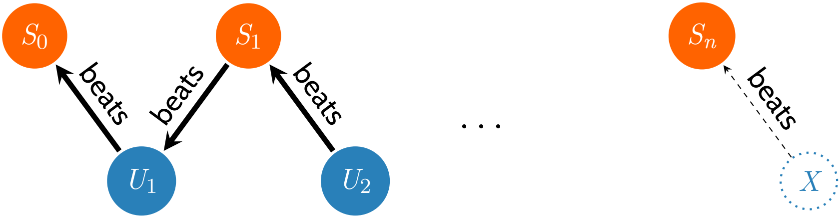
SimSat: SAT via mutual strategy improvement

[Farzan & Kincaid, IJCAI 2016]



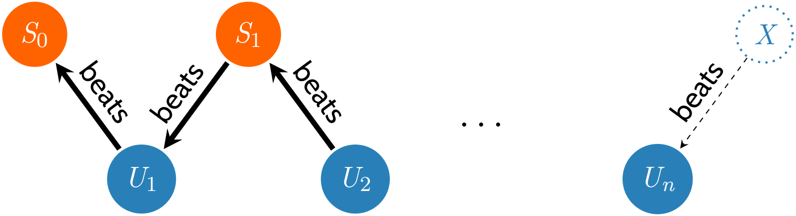
SimSat: SAT via mutual strategy improvement

[Farzan & Kincaid, IJCAI 2016]



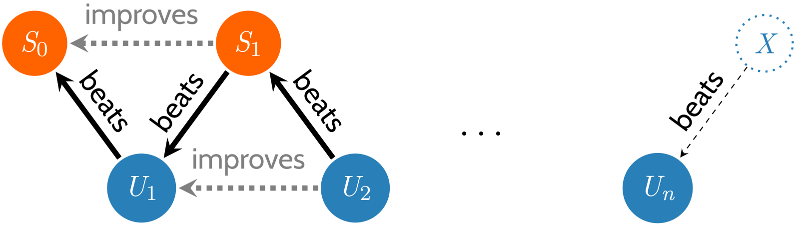
SimSat: SAT via mutual strategy improvement

[Farzan & Kincaid, IJCAI 2016]



SimSat: SAT via mutual strategy improvement

[Farzan & Kincaid, IJCAI 2016]



Strategy skeletons

$$\forall x. \forall y. \exists \mathit{lub}. \forall \mathit{ub}. \mathit{lub} \geq x \wedge \mathit{lub} \geq y \wedge [(\mathit{ub} \geq x \wedge \mathit{ub} \geq y) \implies \mathit{ub} \geq \mathit{lub}]$$

Strategy skeletons

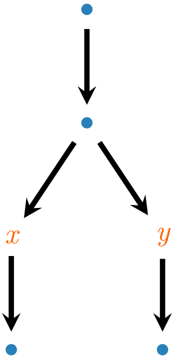
$$\forall x. \forall y. \exists \textit{lub}. \forall \textit{ub}. \textit{lub} \geq x \wedge \textit{lub} \geq y \wedge [(\textit{ub} \geq x \wedge \textit{ub} \geq y) \implies \textit{ub} \geq \textit{lub}]$$

$\forall x$

$\forall y$

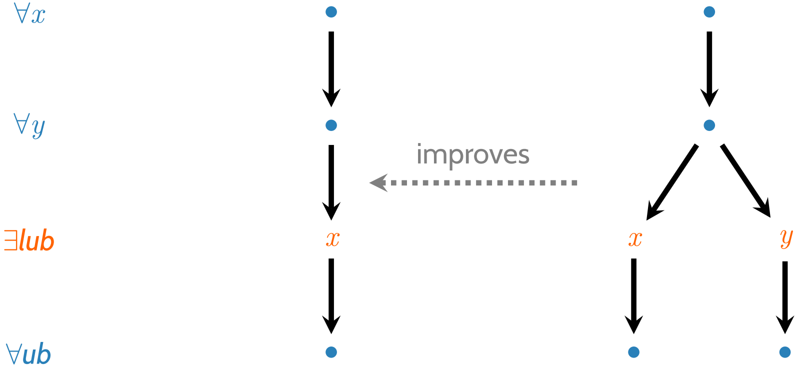
$\exists \textit{lub}$

$\forall \textit{ub}$



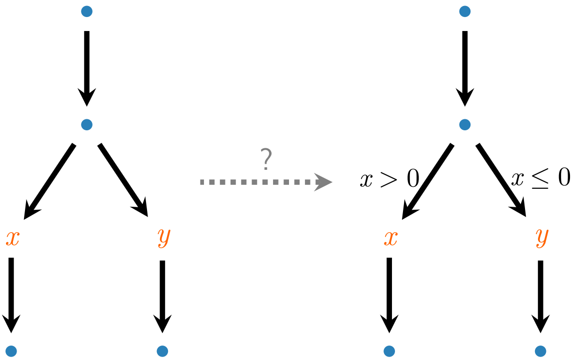
Strategy skeletons

$$\forall x.\forall y.\exists lub.\forall ub.lub \geq x \wedge lub \geq y \wedge [(ub \geq x \wedge ub \geq y) \implies ub \geq lub]$$



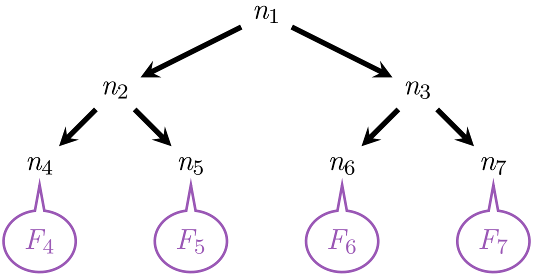
From skeletons to strategies

$$\forall x.\forall y.\exists lub.\forall ub.lub \geq x \land lub \geq y \land [(ub \geq x \land ub \geq y) \implies ub \geq lub]$$



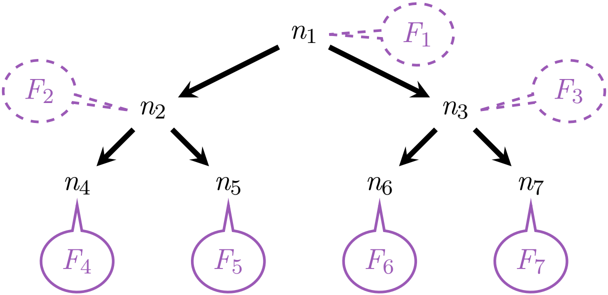
Tree interpolation (special case)

Given tree with leaves labeled by formulas s.t. the conjunction of all labels is inconsistent:



Tree interpolation (special case)

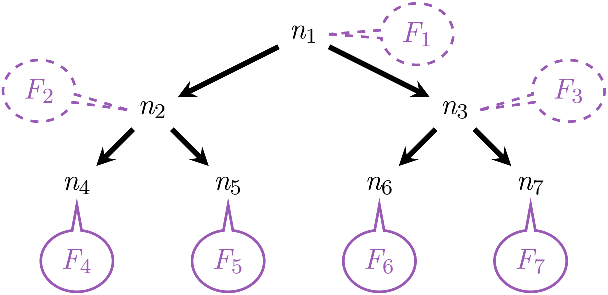
Given tree with leaves labeled by formulas s.t. the conjunction of all labels is inconsistent:



We can find labels for internal nodes s.t.:

Tree interpolation (special case)

Given tree with leaves labeled by formulas s.t. the conjunction of all labels is inconsistent:

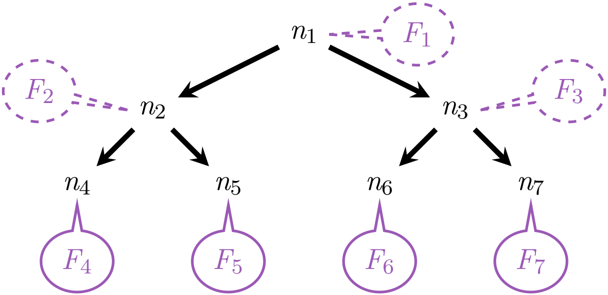


We can find labels for internal nodes s.t.:

- label of root is *false*

Tree interpolation (special case)

Given tree with leaves labeled by formulas s.t. the conjunction of all labels is inconsistent:

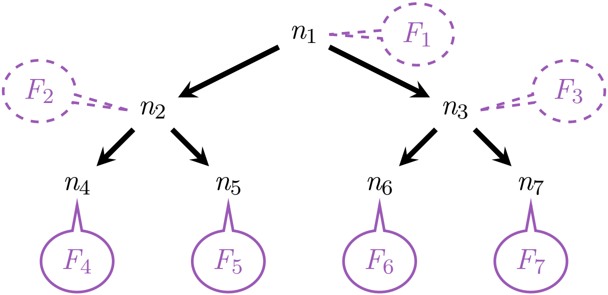


We can find labels for internal nodes s.t.:

- label of root is *false*
- For all nodes $n_i : F_i$
  - conjunction of children's labels implies F_i

Tree interpolation (special case)

Given tree with leaves labeled by formulas s.t. the conjunction of all labels is inconsistent:

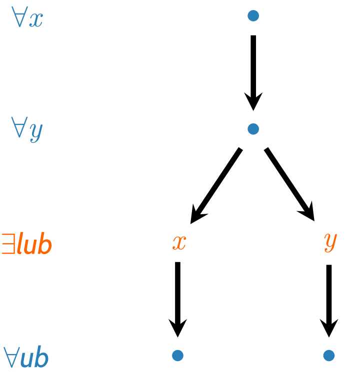


We can find labels for internal nodes s.t.:

- label of root is *false*
- For all nodes $n_i : F_i$
  - conjunction of children's labels implies F_i
  - F_i uses only symbols common to descendents & non-descendents

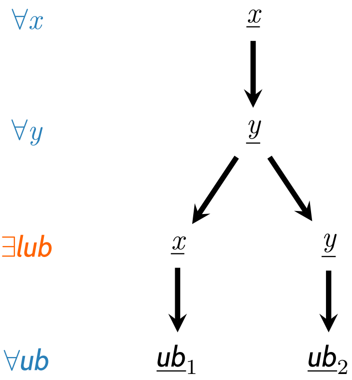
Strategy synthesis

$$\forall x. \forall y. \exists \textit{lub}. \forall \textit{ub}. \underbrace{\textit{lub} \geq x \wedge \textit{lub} \geq y \wedge [(\textit{ub} \geq x \wedge \textit{ub} \geq y) \implies \textit{ub} \geq \textit{lub}]}_{F(x, y, \textit{lub}, \textit{ub})}$$



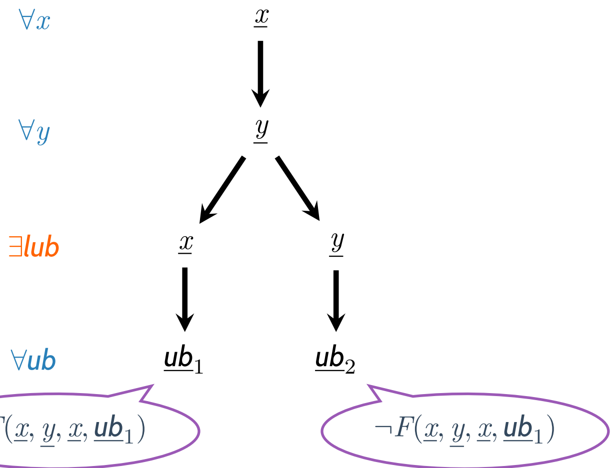
Strategy synthesis

$$\forall x.\forall y.\exists \mathit{lub}.\forall \mathit{ub}.\underbrace{\mathit{lub} \geq x \land \mathit{lub} \geq y \land [(\mathit{ub} \geq x \land \mathit{ub} \geq y) \implies \mathit{ub} \geq \mathit{lub}]}_{F(x,y,\mathit{lub},\mathit{ub})}$$



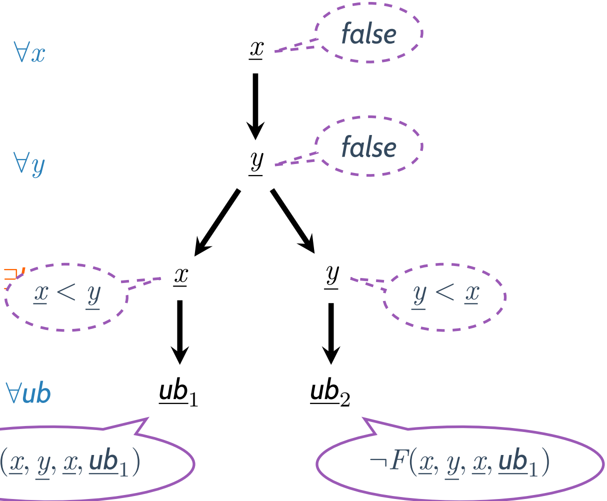
Strategy synthesis

$$\forall x.\forall y.\exists lub.\forall ub.\ \underbrace{lub \geq x \wedge lub \geq y \wedge [(ub \geq x \wedge ub \geq y) \implies ub \geq lub]}_{F(x,y,lub,ub)}$$

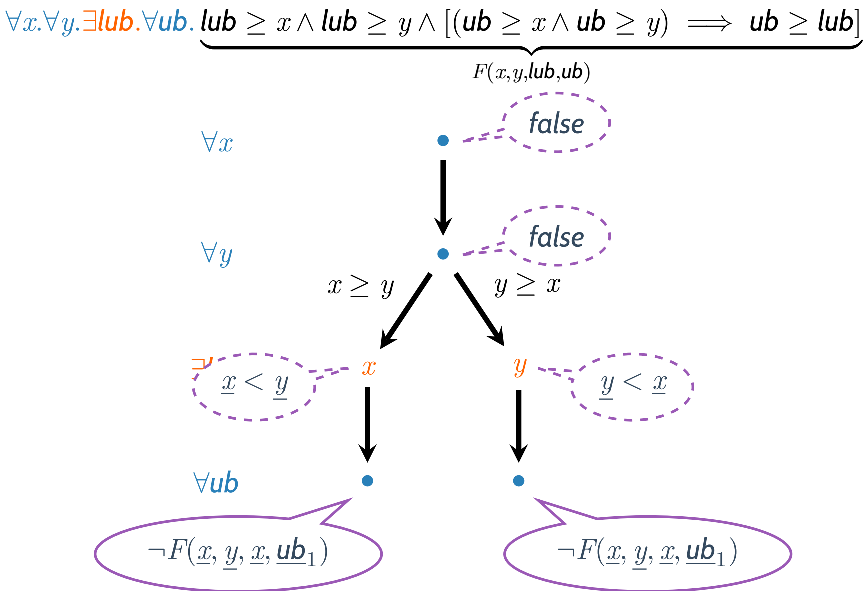


Strategy synthesis

$$\forall x.\forall y.\exists \mathit{lub}.\forall \mathit{ub}.\ \underbrace{\mathit{lub} \geq x \wedge \mathit{lub} \geq y \wedge [(\mathit{ub} \geq x \wedge \mathit{ub} \geq y) \implies \mathit{ub} \geq \mathit{lub}]}_{F(x,y,\mathit{lub},\mathit{ub})}$$



Strategy synthesis



Experiments

Name	Alchemist-CSDT	CVC4-1.5.1	SimSynth
max15	Timeout	3.3s	Timeout
array_search15	Timeout	0.1s	3.0s
array_sum8_15	Timeout	0.0s	0.3s
tenfunc2	0.0s	0.1s	0.1s
polynomial4	0.0s	21.0s	0.0s
hms	Timeout	Timeout	0.0s
scaleweights	Timeout	0.1s	0.3s
lub10	Timeout	38.1s	4.0s
inverse10	Timeout	Timeout	2.4s
round10	Error	Timeout	8.8s
puzzle35	Timeout	Timeout	0.1s
puzzle35_opt	Timeout	Unknown	0.2s

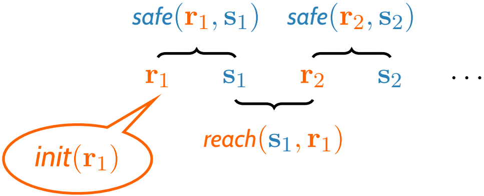
Reachability games

Definition

A *reachability game of dimension* d consists of three formulas:

- *init*$(x_1, \ldots, x_d)$: initial game state (chosen by REACH)
- *reach*$(x_1, \ldots, x_d, x'_1, \ldots, x'_d)$: moves of REACH
- *safe*$(x_1, \ldots, x_d, x'_1, \ldots, x'_d)$: moves of SAFE

REACH and SAFE alternate picking positions in $\mathbb{Q}^d$

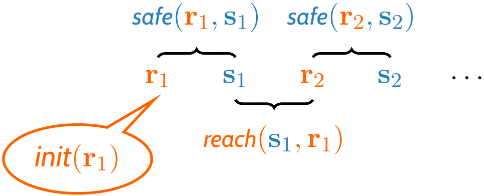


Definition

A *reachability game of dimension* d consists of three formulas:

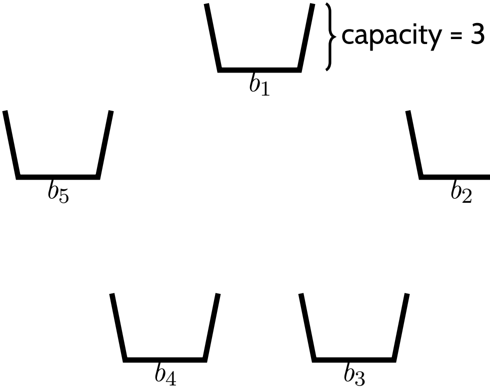
- *init*$(x_1, \ldots, x_d)$: initial game state (chosen by REACH)
- *reach*$(x_1, \ldots, x_d, x'_1, \ldots, x'_d)$: moves of REACH
- *safe*$(x_1, \ldots, x_d, x'_1, \ldots, x'_d)$: moves of SAFE

REACH and SAFE alternate picking positions in $\mathbb{Q}^d$

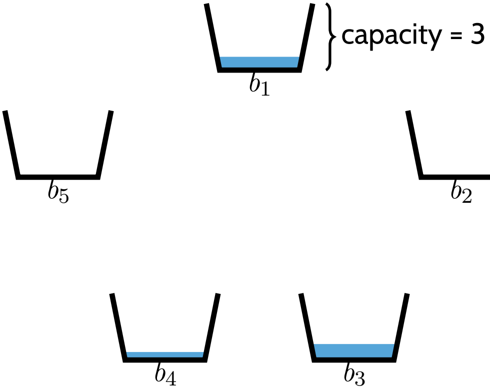


First player to make an illegal move loses.

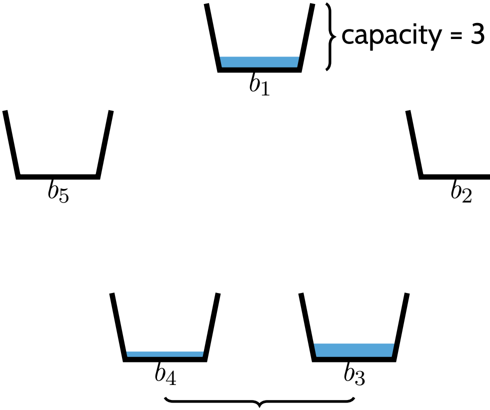
Cinderella-Stepmother



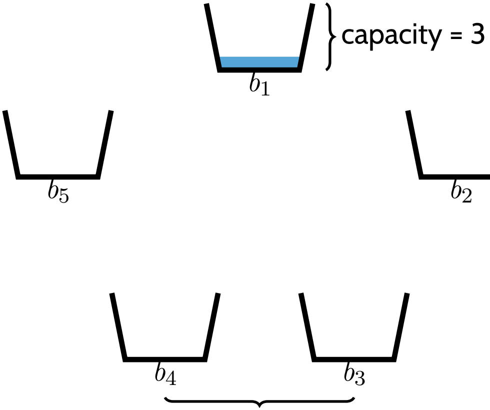
Cinderella-Stepmother



Cinderella-Stepmother



Cinderella-Stepmother



Formalizing Cinderella-Stepmother

$$\textit{init} \triangleq \left(\sum_{i=1}^{5} b_i = 1\right) \wedge \bigwedge_{i=1}^{5} b_i \geq 0$$

$$\textit{reach} \triangleq \sum_{i=1}^{5} b'_i = 1 + \left(\sum_{i=1}^{5} b_i\right) \wedge \bigwedge_{i=1}^{5} b'_i \geq b_i$$

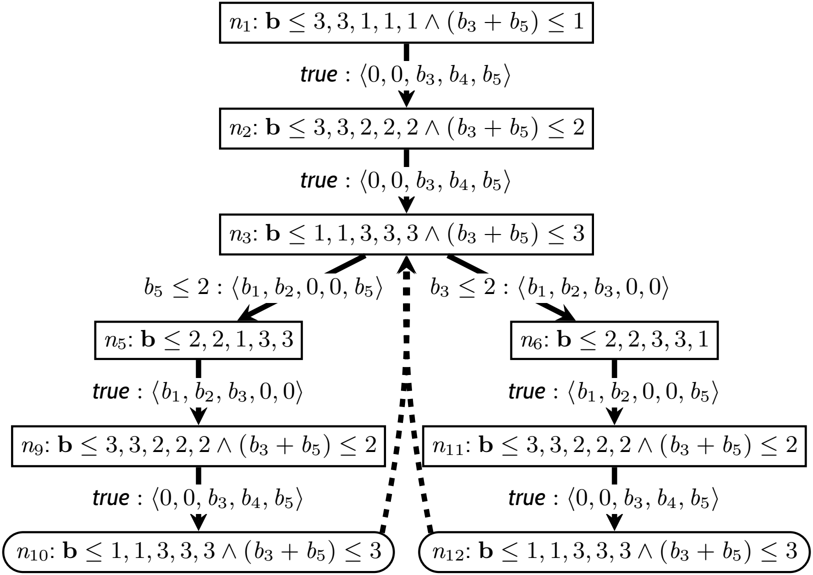
$$\textit{safe} \triangleq \neg\textit{overflow} \wedge \bigvee \textit{empty}_i$$

where

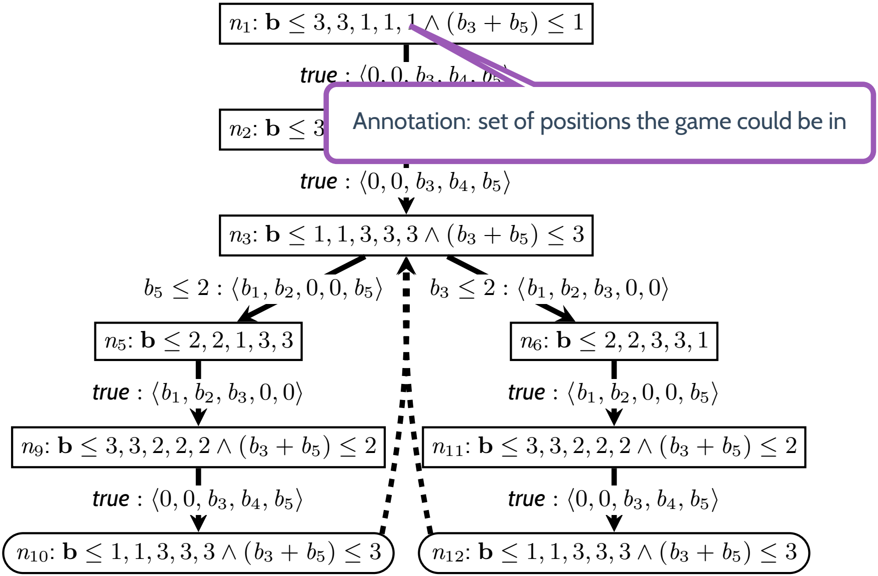
$$\textit{overflow} \triangleq \left(\bigvee_{i=1}^{5} b_i > 3\right)$$

$$\textit{empty}_i \triangleq b'_i, b'_{i+1}, b'_{i+2}, b'_{i+3}, b'_{i+4} = 0, 0, b_{i+2}, b_{i+3}, b_{i+4}$$

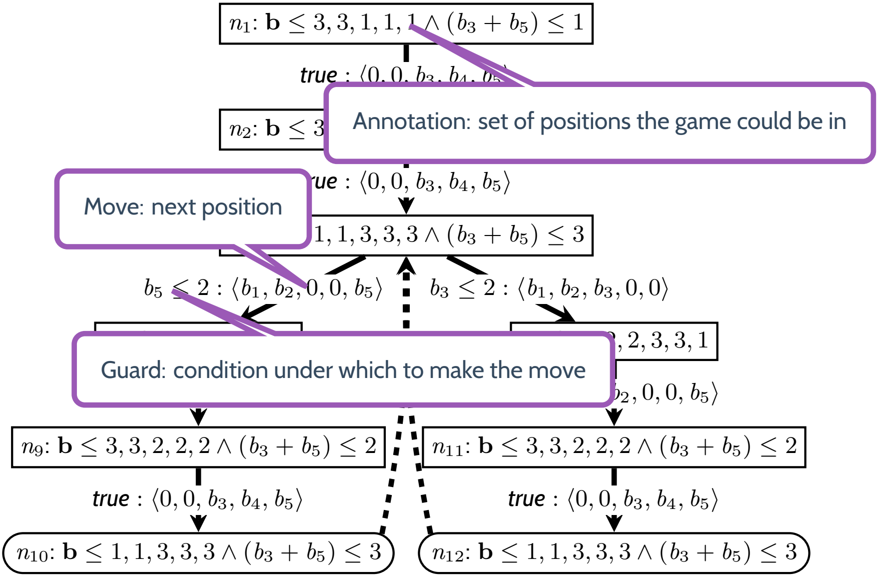
Safety trees



Safety trees

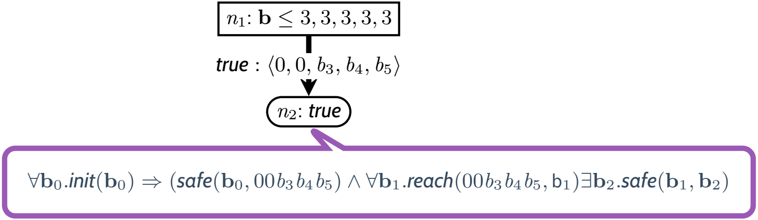


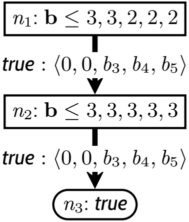
Safety trees



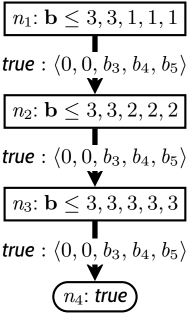
n_1: *true*

$$\forall \mathbf{b}_0.\mathit{init}(\mathbf{b}_0) \Rightarrow \exists \mathbf{b}_1.\mathit{safe}(\mathbf{b}_0, \mathbf{b}_1)$$

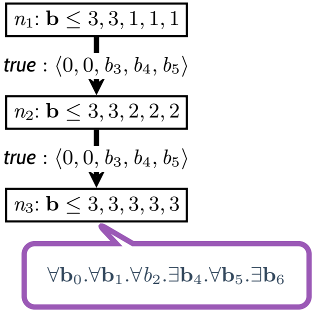

n_1: $\mathbf{b} \leq 3, 3, 3, 3, 3$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: *true*
$\forall \mathbf{b}_0 . init(\mathbf{b}_0) \Rightarrow (safe(\mathbf{b}_0, 00 b_3 b_4 b_5) \wedge \forall \mathbf{b}_1 . reach(00 b_3 b_4 b_5, \mathtt{b}_1) \exists \mathbf{b}_2 . safe(\mathbf{b}_1, \mathbf{b}_2)$



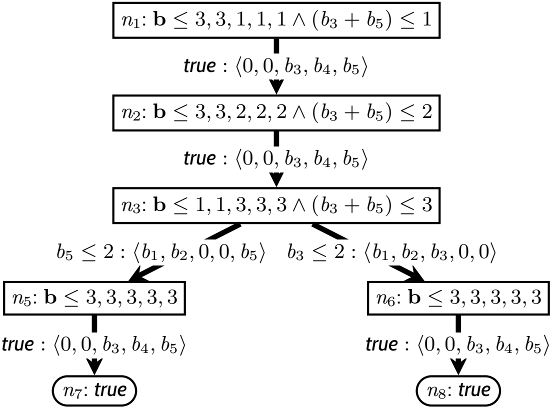
n_1: $\mathbf{b} \leq 3, 3, 2, 2, 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 3, 3, 3$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: *true*

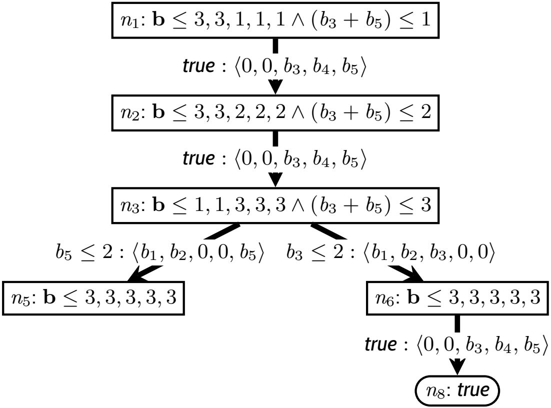


n_1: $\mathbf{b} \leq 3,3,1,1,1$
true : $\langle 0,0,b_3,b_4,b_5\rangle$
n_2: $\mathbf{b} \leq 3,3,2,2,2$
true : $\langle 0,0,b_3,b_4,b_5\rangle$
n_3: $\mathbf{b} \leq 3,3,3,3,3$
true : $\langle 0,0,b_3,b_4,b_5\rangle$
n_4: *true*

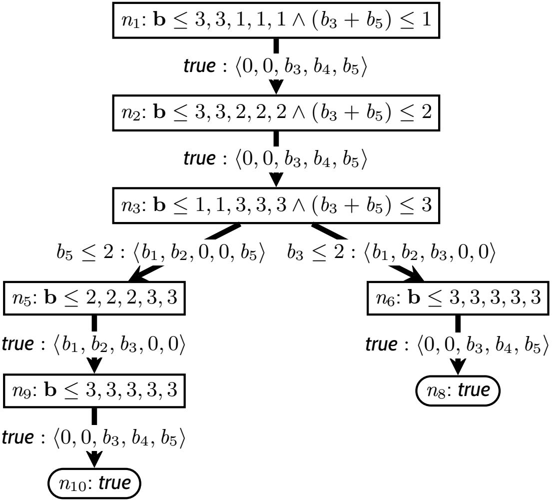


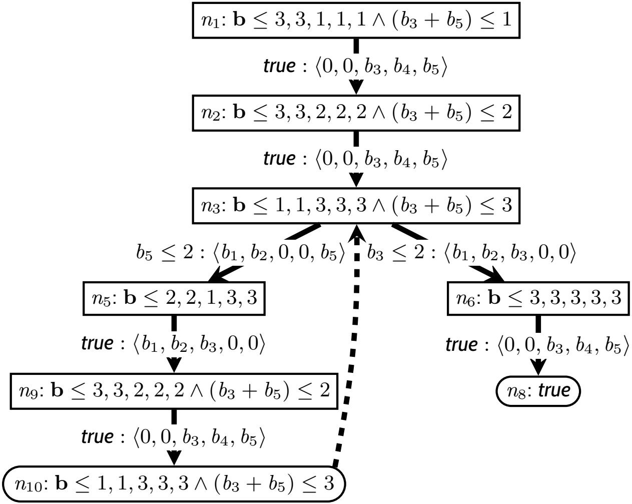
n_1: $\mathbf{b} \leq 3, 3, 1, 1, 1$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 2, 2, 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: $\mathbf{b} \leq 3, 3, 3, 3, 3$
$\forall \mathbf{b}_0 . \forall \mathbf{b}_1 . \forall b_2 . \exists \mathbf{b}_4 . \forall \mathbf{b}_5 . \exists \mathbf{b}_6$


n_1: $\mathbf{b} \leq 3, 3, 1, 1, 1 \wedge (b_3 + b_5) \leq 1$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 3$
$b_5 \leq 2 : \langle b_1, b_2, 0, 0, b_5 \rangle$
$b_3 \leq 2 : \langle b_1, b_2, b_3, 0, 0 \rangle$
n_5: $\mathbf{b} \leq 3, 3, 3, 3, 3$
n_6: $\mathbf{b} \leq 3, 3, 3, 3, 3$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_7: *true*
n_8: *true*

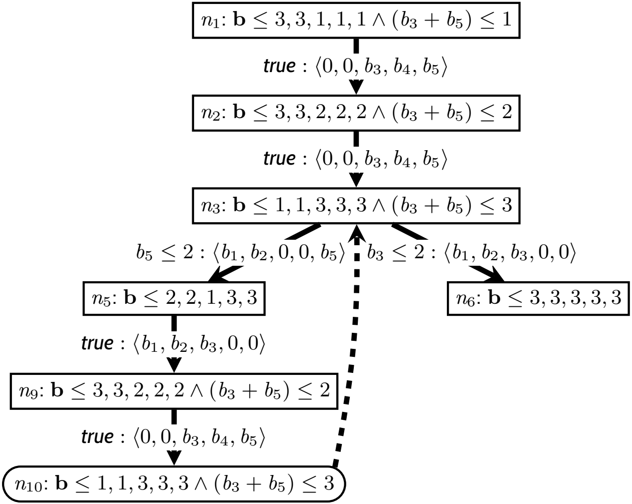


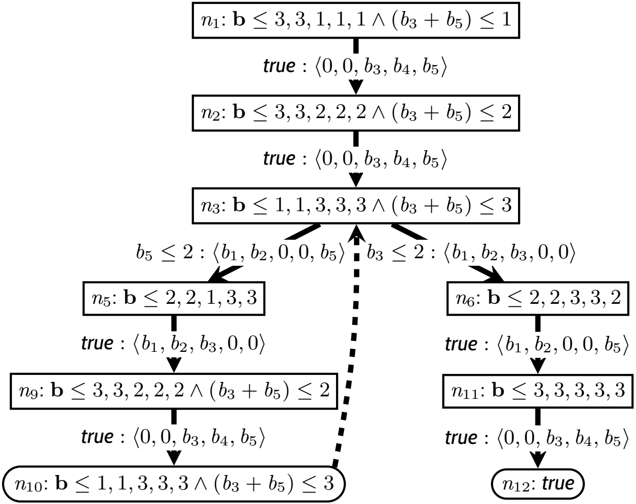
n_1: $\mathbf{b} \leq 3, 3, 1, 1, 1 \wedge (b_3 + b_5) \leq 1$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 3$
$b_5 \leq 2 : \langle b_1, b_2, 0, 0, b_5 \rangle$
$b_3 \leq 2 : \langle b_1, b_2, b_3, 0, 0 \rangle$
n_5: $\mathbf{b} \leq 3, 3, 3, 3, 3$
n_6: $\mathbf{b} \leq 3, 3, 3, 3, 3$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_8: *true*


n_1: $\mathbf{b} \leq 3, 3, 1, 1, 1 \wedge (b_3 + b_5) \leq 1$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 3$
$b_5 \leq 2 : \langle b_1, b_2, 0, 0, b_5 \rangle$
$b_3 \leq 2 : \langle b_1, b_2, b_3, 0, 0 \rangle$
n_5: $\mathbf{b} \leq 2, 2, 2, 3, 3$
n_6: $\mathbf{b} \leq 3, 3, 3, 3, 3$
true : $\langle b_1, b_2, b_3, 0, 0 \rangle$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_9: $\mathbf{b} \leq 3, 3, 3, 3, 3$
n_8: *true*
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_{10}: *true*

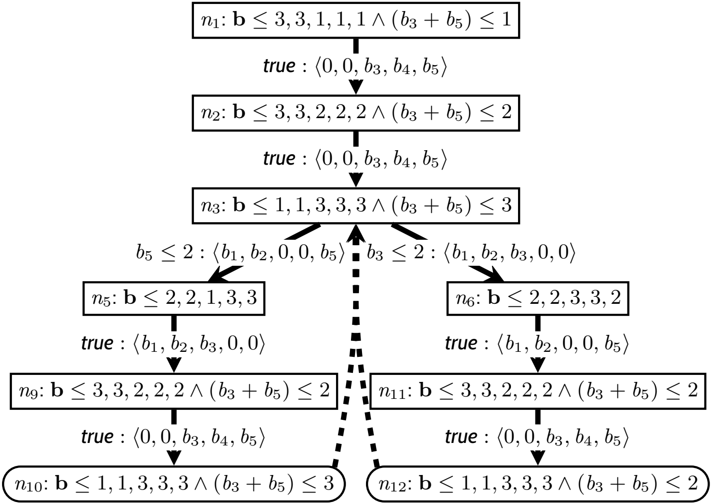


n1: b ≤ 3, 3, 1, 1, 1 ∧ (b3 + b5) ≤ 1
true : ⟨0, 0, b3, b4, b5⟩
n2: b ≤ 3, 3, 2, 2, 2 ∧ (b3 + b5) ≤ 2
true : ⟨0, 0, b3, b4, b5⟩
n3: b ≤ 1, 1, 3, 3, 3 ∧ (b3 + b5) ≤ 3
b5 ≤ 2 : ⟨b1, b2, 0, 0, b5⟩
b3 ≤ 2 : ⟨b1, b2, b3, 0, 0⟩
n5: b ≤ 2, 2, 1, 3, 3
n6: b ≤ 3, 3, 3, 3, 3
true : ⟨b1, b2, b3, 0, 0⟩
true : ⟨0, 0, b3, b4, b5⟩
n9: b ≤ 3, 3, 2, 2, 2 ∧ (b3 + b5) ≤ 2
n8: true
true : ⟨0, 0, b3, b4, b5⟩
n10: b ≤ 1, 1, 3, 3, 3 ∧ (b3 + b5) ≤ 3


n_1: $\mathbf{b} \leq 3, 3, 1, 1, 1 \wedge (b_3 + b_5) \leq 1$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 3$
$b_5 \leq 2 : \langle b_1, b_2, 0, 0, b_5 \rangle$
$b_3 \leq 2 : \langle b_1, b_2, b_3, 0, 0 \rangle$
n_5: $\mathbf{b} \leq 2, 2, 1, 3, 3$
n_6: $\mathbf{b} \leq 3, 3, 3, 3, 3$
true : $\langle b_1, b_2, b_3, 0, 0 \rangle$
n_9: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_{10}: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 3$



n_1: $\mathbf{b} \leq 3, 3, 1, 1, 1 \land (b_3 + b_5) \leq 1$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 2, 2, 2 \land (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: $\mathbf{b} \leq 1, 1, 3, 3, 3 \land (b_3 + b_5) \leq 3$
$b_5 \leq 2 : \langle b_1, b_2, 0, 0, b_5 \rangle$
$b_3 \leq 2 : \langle b_1, b_2, b_3, 0, 0 \rangle$
n_5: $\mathbf{b} \leq 2, 2, 1, 3, 3$
n_6: $\mathbf{b} \leq 2, 2, 3, 3, 2$
true : $\langle b_1, b_2, b_3, 0, 0 \rangle$
true : $\langle b_1, b_2, 0, 0, b_5 \rangle$
n_9: $\mathbf{b} \leq 3, 3, 2, 2, 2 \land (b_3 + b_5) \leq 2$
n_{11}: $\mathbf{b} \leq 3, 3, 3, 3, 3$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_{10}: $\mathbf{b} \leq 1, 1, 3, 3, 3 \land (b_3 + b_5) \leq 3$
n_{12}: *true*



n_1: $\mathbf{b} \leq 3, 3, 1, 1, 1 \wedge (b_3 + b_5) \leq 1$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_2: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_3: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 3$
$b_5 \leq 2 : \langle b_1, b_2, 0, 0, b_5 \rangle$
$b_3 \leq 2 : \langle b_1, b_2, b_3, 0, 0 \rangle$
n_5: $\mathbf{b} \leq 2, 2, 1, 3, 3$
n_6: $\mathbf{b} \leq 2, 2, 3, 3, 2$
true : $\langle b_1, b_2, b_3, 0, 0 \rangle$
true : $\langle b_1, b_2, 0, 0, b_5 \rangle$
n_9: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
n_{11}: $\mathbf{b} \leq 3, 3, 2, 2, 2 \wedge (b_3 + b_5) \leq 2$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
true : $\langle 0, 0, b_3, b_4, b_5 \rangle$
n_{10}: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 3$
n_{12}: $\mathbf{b} \leq 1, 1, 3, 3, 3 \wedge (b_3 + b_5) \leq 2$

Cinderella-Stepmother

Capacity	Winner	Time
c=3	Cinderella	2.2s
c=2.5	Cinderella	53.8s
c=2	Cinderella	68.9s
c=1.8	–	Timeout
c=1.7	Stepmother	2.5s
c=1.6	Stepmother	1.5s
c=1.5	Stepmother	1.4s
c=1.4	Stepmother	0.2s

Cinderella-Stepmother

Capacity	Winner	Time
c=3	Cinderella	2.2s
c=2.5	Cinderella	53.8s
c=2	Cinderella	68.9s
c=1.8	–	Timeout
c=1.7	Stepmother	2.5s
c=1.6	Stepmother	1.5s
c=1.5	Stepmother	1.4s
c=1.4	Stepmother	0.2s

"the problem becomes more challenging for $1.5 \leq c < 3$ …in such cases fully automated strategy synthesis seems unrealistic, and computer-assisted proofs driven by user-provided hints or templates are more plausible."

– Beyene, Chaudhuri, Popeea & Rybalchenko; POPL'14

Summary

- Complete procedure for satisfiability games
  - Extends LRA decision procedure to strategy synthesis
- Semi-algorithm for reachability games
  - Synthesize strategies for bounded games, then generalize
  - Complete for finite REACH strategies