Solving Temporal Problems using SMT

Alessandro Cimatti Andrea Micheli Marco Roveri

Embedded Systems Unit
Fondazione Bruno Kessler, Trento, Italy
cimatti@fbk.eu

4th July 2013

SAT/SMT Summer School 2013

Outline

Outline

Scheduling for planning applications

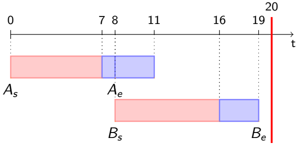
The motivating problem

Planning subject to temporal constraints
We may or may not (more realistically) assume that the executor can always control action duration.

Scheduling for planning applications

The motivating problem

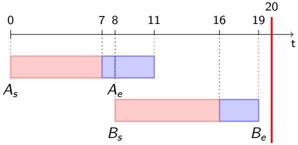
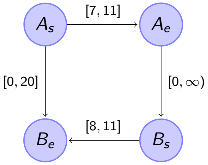
Planning subject to temporal constraints
We may or may not (more realistically) assume that the executor can always control action duration.

	Uncertainty Type	
	No Uncertainty	Uncertainty
Activities		
TimePoints		

Scheduling for planning applications

The motivating problem

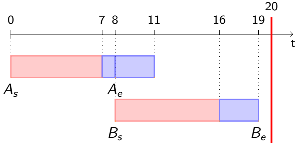
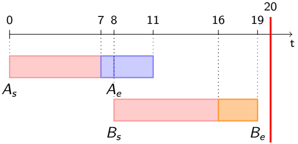
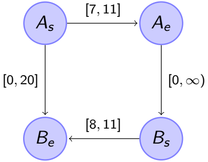
Planning subject to temporal constraints
We may or may not (more realistically) assume that the executor can always control action duration.

<table>
<tr><td></td><td colspan="2">Uncertainty Type</td></tr>
<tr><td></td><td>No Uncertainty</td><td>Uncertainty</td></tr>
<tr><td>Activities</td><td></td><td></td></tr>
<tr><td>TimePoints</td><td></td><td></td></tr>
</table>

Scheduling for planning applications

The motivating problem

Planning subject to temporal constraints
We may or may not (more realistically) assume that the executor can always control action duration.

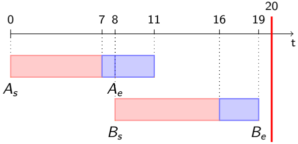
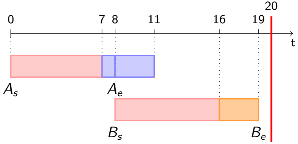
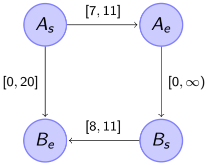
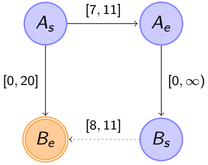
	Uncertainty Type	
	No Uncertainty	Uncertainty
Activities		
TimePoints		

Scheduling for planning applications

The motivating problem

Planning subject to temporal constraints

We may or may not (more realistically) assume that the executor can always control action duration.

	Uncertainty Type	
	No Uncertainty	Uncertainty
Activities		
TimePoints		

Temporal Problems

We need formalisms to describe **temporal knowledge** possibly in presence of uncertainty.

Temporal Problems

We need formalisms to describe **temporal knowledge** possibly in presence of uncertainty.

Temporal Problems

A set of time-valued variables called **time points** constrained by difference constraints.

Temporal Problems

We need formalisms to describe **temporal knowledge** possibly in presence of uncertainty.

Temporal Problems

A set of time-valued variables called **time points** constrained by difference constraints.

Temporal Problems with Uncertainty

A Temporal Problem that distinguishes **controllable** and **uncontrollable** time points and divides constraints in **contingent** (assumptions) and **free** constraints (requirements) to model the temporal uncertainty.

Temporal Problems Formalization

Definition

A Temporal Problem is a tuple (X_c, C_f).

- $X_c \doteq \{b_1, ..., b_n\}$ is the set of *time points*
- $C_f \doteq \{cf_1, ..., cf_h\}$ is the set of *free constraints*

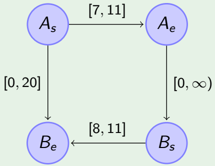
$$cf_i \doteq \bigvee_{j=1}^{D_i} (x_{i,j} - y_{i,j}) \in [l^f_{i,j}, u^f_{i,j}]$$

- $l^f_{i,j}, u^f_{i,j} \in \mathbb{R} \cup \{+\infty, -\infty\}$
- $l^f_{i,j} \leq u^f_{i,j}$
- D_i is the number of disjuncts for the i-th free constraint

- $x_{i,j}, y_{i,j} \in X_c$
- $x_{i,j} \neq y_{i,j}$.

Temporal Problem Example

Example

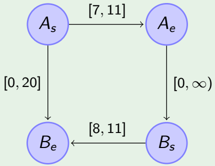


A_s, A_e, B_s, B_e are **Time Points (X_c)**

$\longrightarrow$ represents **Free Constraints (C_f)**

Temporal Problem Example

Example



A_s, A_e, B_s, B_e are **Time Points (X_c)**

$\longrightarrow$ represents **Free Constraints (C_f)**

Taxonomy

Let $X_c \doteq \{x_1, ..., x_k\}$.

STP	TCSP	DTP
No disjunctions $(x_i - x_j) \in [l, u]$	Interval disjunctions $(x_i - x_j) \in \bigcup_w [l_w, u_w]$	Arbitrary disjunctions $\bigvee_w ((x_{i_w} - x_{j_w}) \in [l_w, u_w])$

Temporal Problem Solution

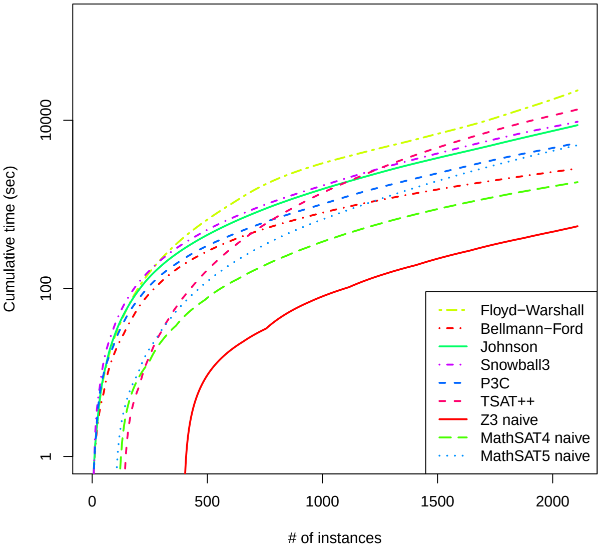
Consistency

Given a temporal problem, we can check if there exists an assignment to time points that fulfills all the constraints.
This can be done by a single call to an SMT($\mathcal{DL}$) solver!

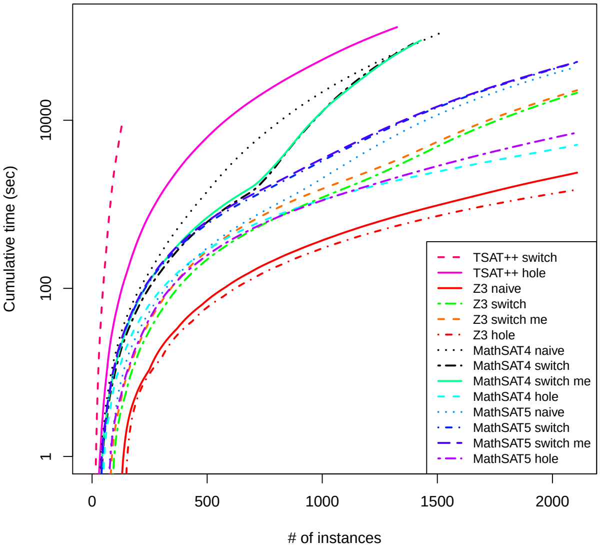
Minimal Network

Sometimes we want to retain *flexibility*, hence we do not decide all the time points upfront but we propagate the constraints to be as strict as possible without loosing solutions.
Requires runtime propagation!

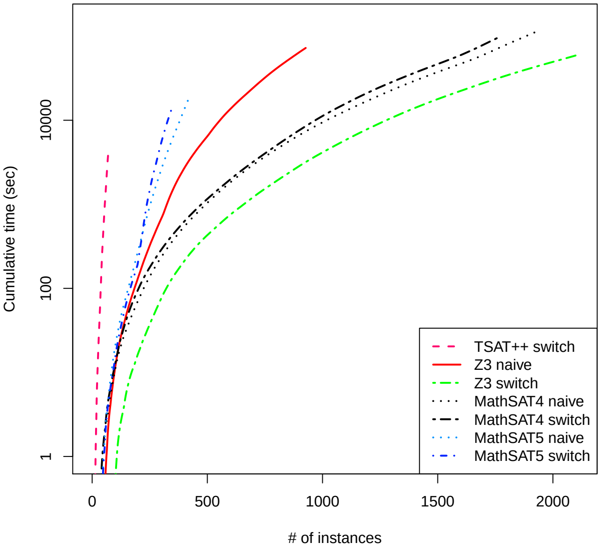
Consistency of *STP*



Consistency of *TCSP*



Consistency of *DTP*



Temporal Problems with Uncertainty Formalization

Definition

A Temporal Problem with Uncertainty is a tuple (X_c, X_u, C_c, C_f).

- $X_c \doteq \{b_1, ..., b_n\}$ is the set of *controllable time points*
- $X_u \doteq \{e_1, ..., e_m\}$ is the set of *uncontrollable time points*
- $C_c \doteq \{cc_1, ..., cc_m\}$ is the set of *contingent constraints*
- $C_f \doteq \{cf_1, ..., cf_h\}$ is the set of *free constraints*

$$cc_i \doteq \bigvee_{j=1}^{E_i}(e_i - b_i) \in [l^c_{i,j}, u^c_{i,j}] \qquad cf_i \doteq \bigvee_{j=1}^{D_i}(x_{i,j} - y_{i,j}) \in [l^f_{i,j}, u^f_{i,j}]$$

- $l^{c/f}_{i,j}, u^{c/f}_{i,j} \in \mathbb{R} \cup \{+\infty, -\infty\}$
- $l^{c/f}_{i,j} \leq u^{c/f}_{i,j}$
- D_i is the number of disjuncts for the i-th free constraint
- E_i is the number of disjuncts for the i-th contingent constraint
- $x_{i,j}, y_{i,j} \in X_c \cup X_u$
- $x_{i,j} \neq y_{i,j}$.

Temporal Uncertainty Characterization

Uncertainty can be seen as a **game** between an *Executor* and the adversarial *Nature*.

Temporal Uncertainty Characterization

Uncertainty can be seen as a **game** between an *Executor* and the adversarial *Nature*.

Rules

- The *Executor* schedules a set of **Controllable Time Points (X_c)**

Temporal Uncertainty Characterization

Uncertainty can be seen as a **game** between an *Executor* and the adversarial *Nature*.

Rules

- The *Executor* schedules a set of **Controllable Time Points (X_c)**
- The *Executor* must fulfill a set of temporal constraints called **Free Constraints (C_f)**

Temporal Uncertainty Characterization

Uncertainty can be seen as a **game** between an *Executor* and the adversarial *Nature*.

Rules

- The *Executor* schedules a set of **Controllable Time Points (X_c)**
- The *Executor* must fulfill a set of temporal constraints called **Free Constraints (C_f)**
- The *Nature* tries to prevent the success of the executor scheduling a set of **Uncontrollable Time Points (X_u)**

Temporal Uncertainty Characterization

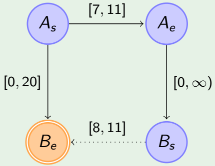
Uncertainty can be seen as a **game** between an *Executor* and the adversarial *Nature*.

Rules

- The *Executor* schedules a set of **Controllable Time Points (X_c)**
- The *Executor* must fulfill a set of temporal constraints called **Free Constraints (C_f)**
- The *Nature* tries to prevent the success of the executor scheduling a set of **Uncontrollable Time Points (X_u)**
- The *Nature* must fulfill a set of temporal constraints called **Contingent Constraints (C_c)**

Temporal Problem with Uncertainty Example

Example

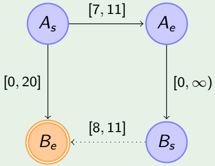


A_s, A_e, B_s are **Controllable Time Points (X_c)**
B_e is an **Uncontrollable Time Point (X_u)**

$\longrightarrow$ represents **Free Constraints (C_f)**
$\cdots\rightarrow$ represents **Contingent Constraints (C_c)**

Temporal Problem with Uncertainty Example

Example



A_s, A_e, B_s are **Controllable Time Points** (X_c)
B_e is an **Uncontrollable Time Point** (X_u)

$\longrightarrow$ represents **Free Constraints** (C_f)
$\cdots\rightarrow$ represents **Contingent Constraints** (C_c)

Taxonomy

Let $\{x_1, ..., x_k\} \doteq X_c \cup X_u$.

STPU	TCSPU	DTPU
No disjunctions $(x_i - x_j) \in [l, u]$	Interval disjunctions $(x_i - x_j) \in \bigcup_w [l_w, u_w]$	Arbitrary disjunctions $\bigvee_w ((x_{i_w} - x_{j_w}) \in [l_w, u_w])$

Temporal Problem with Uncertainty Solution

Three possible degrees of **Controllability**

Strong Controllability (No observation)

Find a *fixed schedule* for controllable time points that fulfills all the free constraints for every possible assignment to uncontrollable time points fulfilling contingent constraints.

Temporal Problem with Uncertainty Solution

Three possible degrees of **Controllability**

Strong Controllability (No observation)

Find a *fixed schedule* for controllable time points that fulfills all the free constraints for every possible assignment to uncontrollable time points fulfilling contingent constraints.

Dynamic Controllability (Past observation)

Find a *strategy*, that depends on past observations only, for scheduling controllable time points that fulfills all the free constraints for every possible assignment to uncontrollable time points fulfilling contingent constraints.

Temporal Problem with Uncertainty Solution

Three possible degrees of **Controllability**

Strong Controllability (No observation)

Find a *fixed schedule* for controllable time points that fulfills all the free constraints for every possible assignment to uncontrollable time points fulfilling contingent constraints.

Dynamic Controllability (Past observation)

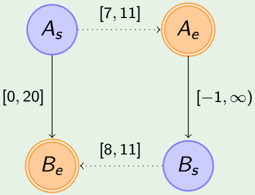
Find a *strategy*, that depends on past observations only, for scheduling controllable time points that fulfills all the free constraints for every possible assignment to uncontrollable time points fulfilling contingent constraints.

Weak Controllability (Full observation)

Find a *strategy* for scheduling controllable time points that fulfills all the free constraints for every possible assignment to uncontrollable time points fulfilling contingent constraints.

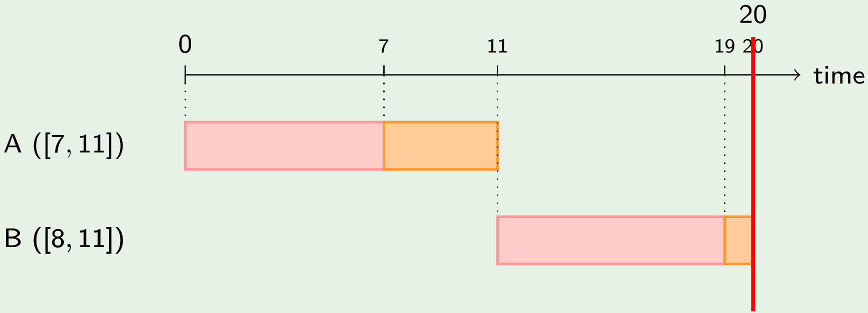
Schedules and Strategies Examples

Example



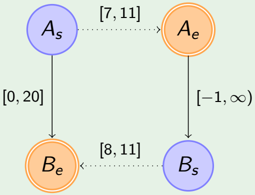
Fixed Schedule (Strong Controllability)

- *start*(A) at 0
- *start*(B) at 11



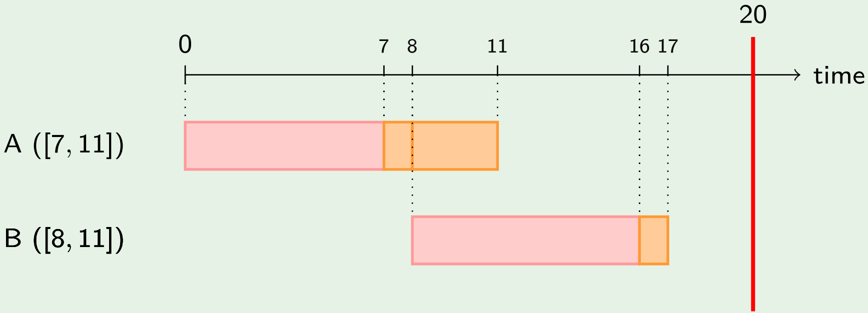
Schedules and Strategies Examples

Example



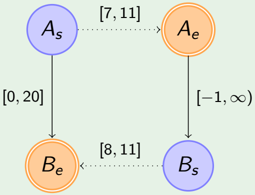
Dynamic Strategy (Dynamic Controllability)

- *start*(A) at 0
- *start*(B) at A_e



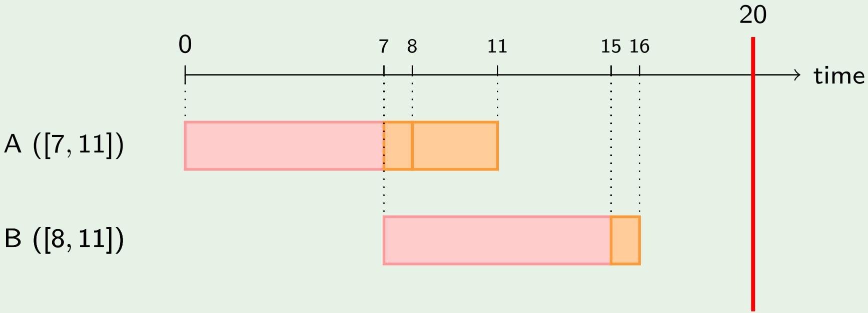
Schedules and Strategies Examples

Example



Clairvoyant Strategy (Weak Controllability)

- $start(A)$ at 0
- $start(B)$ at $A_e - 1$



Temporal Problem Taxonomy Recap

Let $\{x_1, ..., x_k\}$ be the set of all time points of the temporal problem (with uncertainty).

		Uncertainty Type	
		No Uncertainty	Uncertainty
Constraint Type	No disjunctions $(x_i - x_j) \in [l, u]$	STP	STPU
	Interval disjunctions $(x_i - x_j) \in \bigcup_w [l_w, u_w]$	TCSP	TCSPU
	Arbitrary disjunctions $\bigvee_w ((x_{i_w} - x_{j_w}) \in [l_w, u_w])$	DTP	DTPU

Outline

Satisfiability Modulo Theory (*SMT*)

SMT is the problem of deciding satisfiability of a first-order Boolean combination of theory atoms in a given theory T.

Given a formula ϕ, ϕ is satisfiable if there exists a model μ such that $\mu \models \phi$.

Satisfiability Modulo Theory (*SMT*)

SMT is the problem of deciding satisfiability of a first-order Boolean combination of theory atoms in a given theory T.

Given a formula ϕ, ϕ is satisfiable if there exists a model μ such that $\mu \models \phi$.

Example

$\phi \doteq (\forall x.(x > 0) \vee (y \geq x)) \wedge (z \geq y)$
is satisfiable in the theory of real arithmetic because

$$\mu = \{(y, 6),\ (z, 8)\}$$

is a model that satisfies ϕ.

Satisfiability Modulo Theory (*SMT*)

SMT is the problem of deciding satisfiability of a first-order Boolean combination of theory atoms in a given theory T.

Given a formula ϕ, ϕ is satisfiable if there exists a model μ such that $\mu \models \phi$.

Example

$\phi \doteq (\forall x.(x > 0) \vee (y \geq x)) \wedge (z \geq y)$ is satisfiable in the theory of real arithmetic because

$$\mu = \{(y, 6),\ (z, 8)\}$$

is a model that satisfies ϕ.

Theories

Various theories can be used.

In this work:

- $\mathcal{LRA}$ (*Linear Real Arithmetic*)
- $\mathcal{QF\_LRA}$ (*Quantifier-Free Linear Real Arithmetic*)

AllSMT

The AllSMT problem

- $\vec{b}$ is a vector of **Boolean** variables
- $\phi(\vec{x}, \vec{b})$ is a quantifier-free *SMT* formula in some theory T

We want a quantifier-free formula $\psi(\vec{b})$ such that $\psi(\vec{b}) \Leftrightarrow \exists \vec{x}.\phi(\vec{x}, \vec{b})$

```
procedure ALLSMT(φ(x⃗, b⃗))
    res(b⃗) ← ⊥
    while SMT(φ(x⃗, b⃗)) do
        model ← GETMODEL()
        cube(b⃗) ← ⊤
        for all b_i ∈ b⃗ do
            if b_i ∈ model then cube(b⃗) ← cube(b⃗) ∧ b_i
            else if ¬b_i ∈ model then cube(b⃗) ← cube(b⃗) ∧ ¬b_i
        res(b⃗) ← res(b⃗) ∨ cube(b⃗)
        φ(x⃗, b⃗) ← φ(x⃗, b⃗) ∧ ¬cube(b⃗)
    return res(b⃗)
```

Quantifier Elimination

Quantifier Elimination Definition

A theory T has quantifier elimination if for every formula Φ, there exists another formula Φ_{QF} without quantifiers which is *equivalent* to it (modulo the theory T)

Quantifier Elimination

Quantifier Elimination Definition

A theory T has quantifier elimination if for every formula Φ, there exists another formula Φ_{QF} without quantifiers which is *equivalent* to it (modulo the theory T)

Example

$\mathcal{LRA}$ theory admits quantifier elimination, but elimination algorithms are very costly (doubly exponential in the size of the original formula).

$(\exists x.(x \geq 2y + z) \wedge (x \leq 3z + 5)) \leftrightarrow (2y - 2z - 5 \leq 0)$

Quantifier Elimination

Quantifier Elimination Definition

A theory T has quantifier elimination if for every formula Φ, there exists another formula Φ_{QF} without quantifiers which is *equivalent* to it (modulo the theory T)

Example

$\mathcal{LRA}$ theory admits quantifier elimination, but elimination algorithms are very costly (doubly exponential in the size of the original formula).

$(\exists x.(x \geq 2y + z) \wedge (x \leq 3z + 5)) \leftrightarrow (2y - 2z - 5 \leq 0)$

Note: Quantifier Elimination more general than AllSMT! Here we can leave (some/all) theory variables unquantified!

Quantifier Elimination for $\mathcal{LRA}$

Various techniques

- Fourier-Motzkin
- Loos-Weisspfenning
- ...

Fourier-Motzkin Elimination

- Procedure that eliminates a variable from a **conjunction** of linear inequalities.
- It can be applied to a general $\mathcal{LRA}$ formula by computing the DNF and applying the technique to each disjunct.
- The complexity is doubly exponential: in the number of variable to quantify and in the size of the DNF formula.

Fourier-Motzkin Elimination

Let $\psi \doteq \exists x_r . \bigwedge_{i=0}^{N} \sum_{k=1}^{M} a_{ik} x_k \leq b_i$ be the problem we want to solve, where x_r is the variable to eliminate.

We have three kinds of inequalities in a system of linear inequalities:

- $x_r \geq A_h$, where $A_h \doteq b_i - \sum_{k=1}^{r_i-1} a_{ik} x_k$, for $h \in [1, H_A]$
- $x_r \leq B_h$, where $B_h \doteq b_i - \sum_{k=1}^{r_i-1} a_{ik} x_k$, for $h \in [1, H_B]$
- Inequalities in which x_r has no role. Let ϕ be the conjunction of those inequalities.

The system is **equivalent** to $(max_{h=1}^{H_A}(A_h) \leq x_r \leq min_{h=1}^{H_b}(B_h)) \wedge \phi$ and to $(max_{h=1}^{H_A}(A_h) \leq min_{h=1}^{H_b}(B_h)) \wedge \phi$

max and *min* are not linear functions, but we can mimic the formula by using a quadratic number of linear inequalities:

$$\psi \Leftrightarrow (\bigwedge_{i=0}^{H_A} \bigwedge_{j=0}^{H_B} A_i \leq B_j) \wedge \phi$$

Fourier-Motzkin Example

Fourier Motzkin Example: Step 1

Let $\psi \doteq \forall z.((z \geq 4) \to ((x < z) \wedge (y < z)))$.

We convert all the quantifiers in existentials and we compute the DNF of the quantified part of the formula.

$\psi \Leftrightarrow \neg\exists z.((z \geq 4) \wedge \neg((x < z) \wedge (y < z)))$

$\psi \Leftrightarrow \neg\exists z.((z \geq 4) \wedge (\neg(x < z) \vee \neg(y < z)))$

$\psi \Leftrightarrow \neg\exists z.(((z \geq 4) \wedge \neg(x < z)) \vee ((z \geq 4) \wedge \neg(y < z)))$

Fourier Motzkin Example: Step 2

For every disjunct, we apply the Fourier-Motzkin Elimination:

$((z \geq 4) \wedge (z \leq x)) \Leftrightarrow (4 \leq x)$

$((z \geq 4) \wedge (z \leq y)) \Leftrightarrow (4 \leq y)$

Then, we rebuild the formula:

$\psi \Leftrightarrow \neg((4 \leq x) \vee (4 \leq y))$

$\psi \Leftrightarrow ((x < 4) \wedge (y < 4))$

Outline

Strong Controllability

Intuition

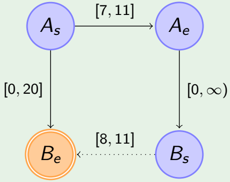
Search for a **Fixed Schedule** that fulfills all free the constraints in every situation.

Strong Controllability

Intuition

Search for a **Fixed Schedule** that fulfills all free the constraints in every situation.

Example



Schedule:

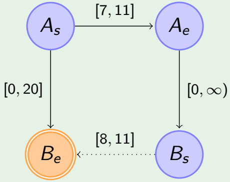
Var	Time
A_s	0
A_e	8
B_s	9

Strong Controllability

Intuition

Search for a **Fixed Schedule** that fulfills all free the constraints in every situation.

Example



Schedule:

Var	Time
A_s	0
A_e	8
B_s	9

Definition

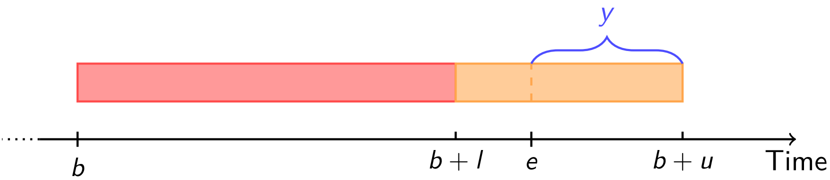
A temporal problem with uncertainty is **Strongly Controllable** if

$$\exists \vec{X}_c . \forall \vec{X}_u . (C_c(\vec{X}_c, \vec{X}_u) \rightarrow C_f(\vec{X}_c, \vec{X}_u))$$

where $\vec{X}_c$ and $\vec{X}_u$ are the vectors of controllable and uncontrollable time points respectively, $C_c(\vec{X}_c, \vec{X}_u)$ are the contingent constraints and $C_f(\vec{X}_c, \vec{X}_u)$ are the free constraints.

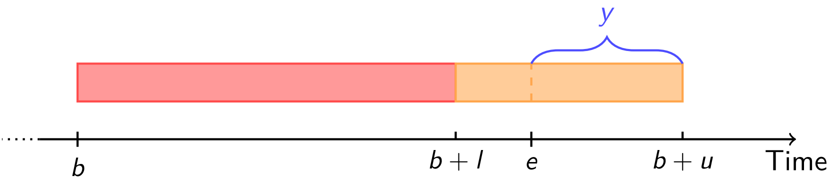
First step: Uncontrollability Isolation

Let $e \in X_u$ and $b \in X_c$.
For every contingent constraint $(e - b) \in [l, u]$, we introduce an offset $y \doteq b + u - e$.



First step: Uncontrollability Isolation

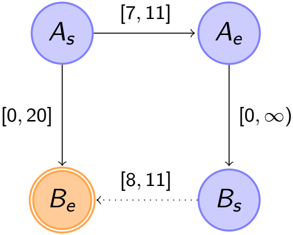
Let $e \in X_u$ and $b \in X_c$.
For every contingent constraint $(e - b) \in [l, u]$, we introduce an offset $y \doteq b + u - e$.



Definition

- Let $\vec{Y}_u$ be the offsets for a given Temporal Problem with Uncertainty
- Let $\Gamma(\vec{Y}_u)$ be the rewritten Contingent Constraints
- Let $\Psi(\vec{X}_c, \vec{Y}_u)$ the rewritten Free Constraints.

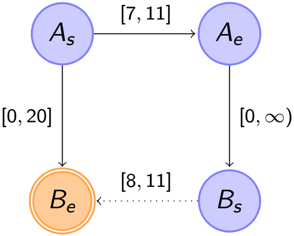
Uncontrollability Isolation: example



Original formulation

$$\begin{aligned}
&\exists A_s, A_e, B_s.\, \forall B_e. \\
&\quad ((B_e - B_s) \in [8, 11]) \rightarrow (((A_e - A_s) \in [7, 11]) \\
&\qquad\qquad\qquad\qquad\qquad\quad \wedge ((B_e - A_s) \in [0, 20]) \\
&\qquad\qquad\qquad\qquad\qquad\quad \wedge ((B_s - A_e) \in [0, \infty)))
\end{aligned}$$

Uncontrollability Isolation: example



Original formulation

$$
\begin{aligned}
&\exists A_s, A_e, B_s.\, \forall B_e. \\
&\quad ((B_e - B_s) \in [8, 11]) \rightarrow (((A_e - A_s) \in [7, 11]) \\
&\qquad\qquad \wedge ((B_e - A_s) \in [0, 20]) \\
&\qquad\qquad \wedge ((B_s - A_e) \in [0, \infty)))
\end{aligned}
$$

Rewritten formulation with Y_{B_e} offset

$$
\begin{aligned}
&\exists A_s, A_e, B_s.\, \forall Y_{B_e}. \\
&\quad (Y_{B_e} \in [0, 3]) \rightarrow (((A_e - A_s) \in [7, 11]) \\
&\qquad\qquad \wedge (((B_s + 11 - Y_{B_e}) - A_s) \in [0, 20]) \\
&\qquad\qquad \wedge ((B_s - A_e) \in [0, \infty)))
\end{aligned}
$$

- $\vec{Y}_u = [Y_{B_e}]$
- $\Gamma(\vec{Y}_u) = (Y_{B_e} \in [0, 3])$
- $\Psi(\vec{X}_c, \vec{Y}_u) = (((A_e - A_s) \in [7, 11]) \wedge \ldots \in [0, \infty)))$

Outline

Direct and Naïve encodings

Direct Encoding

Strong Controllability definition is by itself an encoding in SMT($\mathcal{LRA}$)

$$\exists \vec{X}_c . \forall \vec{X}_u . (C_c(\vec{X}_c, \vec{X}_u) \rightarrow C_f(\vec{X}_c, \vec{X}_u))$$

Direct and Naïve encodings

Direct Encoding

Strong Controllability definition is by itself an encoding in SMT($\mathcal{LRA}$)

$$\exists \vec{X}_c . \forall \vec{X}_u . (C_c(\vec{X}_c, \vec{X}_u) \rightarrow C_f(\vec{X}_c, \vec{X}_u))$$

Naïve Encoding

Thanks to uncontrollability isolation, Strong Controllability can be rewritten as follows.

$$\exists \vec{X}_c . \forall \vec{Y}_u . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$$

Distributed Encoding

Idea: because of the cost of quantifier elimination, many small quantifications can be solved more efficiently than a big single one.

Distributed Encoding

Idea: because of the cost of quantifier elimination, many small quantifications can be solved more efficiently than a big single one.

Starting Point

We assume $\Psi(\vec{X}_c, \vec{Y}_u)$

$$\Psi(\vec{X}_c, \vec{Y}_u) \doteq \bigwedge_h \psi_h(\vec{X}_{c_h}, \vec{Y}_{u_h})$$

Distributed Encoding

Idea: because of the cost of quantifier elimination, many small quantifications can be solved more efficiently than a big single one.

Starting Point

We assume $\Psi(\vec{X}_c, \vec{Y}_u)$

$$\Psi(\vec{X}_c, \vec{Y}_u) \doteq \bigwedge_h \psi_h(\vec{X}_{c_h}, \vec{Y}_{u_h})$$

Distributed Encoding

From the Naïve Encoding we can derive a Distributed Encoding, by pushing the quantifications:

$$\exists \vec{X}_c. \bigwedge_h \forall \vec{Y}_{u_h}.(\neg\Gamma(\vec{Y}_u)|_{Y_{u_h}} \vee \psi_h(\vec{X}_{c_h}, \vec{Y}_{u_h}))$$

Eager $\forall$ Elimination Encoding

Idea: Starting from *Distributed Encoding*, we can eliminate quantifiers during the encoding, producing a $\mathcal{QF\_LRA}$ formula.

Eager $\forall$ Elimination Encoding

Idea: Starting from *Distributed Encoding*, we can eliminate quantifiers during the encoding, producing a $\mathcal{QF\_LRA}$ formula.

Encoding

Let

$$\psi_h^{\Gamma}(\vec{X}_{c_h}) \doteq \neg\exists\vec{Y}_{u_h}.(\Gamma(\vec{Y}_{u_h})|_{Y_{u_h}} \wedge \neg\psi_h(\vec{X}_{c_h}, \vec{Y}_{u_h}))$$

1. Resolve $\psi_h^{\Gamma}(\vec{X}_{c_h})$ for every clause independently using a quantifier elimination procedure
2. Solve the $\mathcal{QF\_LRA}$ encoding:

$$\exists\vec{X}_c. \bigwedge_h \psi_h^{\Gamma}(\vec{X}_{c_h})$$

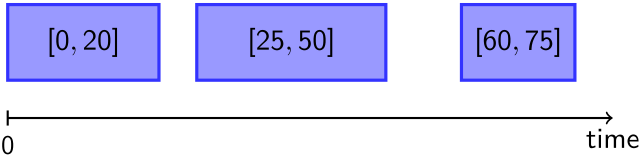
Outline

1. Temporal Problems (with Uncertainty)
2. Background
3. Strong Controllability via SMT
   - DTPU encodings
   - TCSPU specific encodings
   - Experimental Evaluation
4. Weak Controllability via SMT
   - Decision Problem
   - Strategies
   - Linear strategies
   - Piecewise linear strategies
   - Experimental Evaluation
5. Conclusion

Exploit *TCSPU* structure

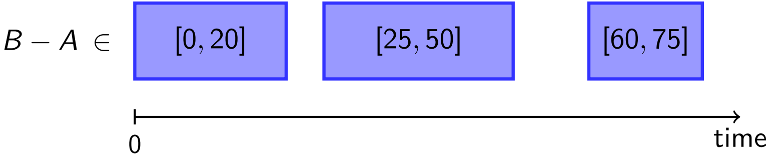
Consider a single *TCSPU* constraint:

$B - A \in$



Exploit *TCSPU* structure

Consider a single *TCSPU* constraint:



Encoding TCSPU constraints in 2-CNF (Hole Encoding)

$$
\begin{aligned}
&((B-A) > 0) \\
\wedge\ &((B-A) < 20) \vee ((B-A) > 25) \\
\wedge\ &((B-A) < 50) \vee ((B-A) > 60) \\
\wedge\ &((B-A) < 75)
\end{aligned}
$$

Static quantification *TCSPU*

Idea: Exploit Hole Encoding for *TCSPU* to statically resolve quantifiers in the Eager $\forall$ elimination encoding.

Static quantification *TCSPU*

Idea: Exploit Hole Encoding for *TCSPU* to statically resolve quantifiers in the Eager $\forall$ elimination encoding.

Approach

Hole Encoding gives us a 2-CNF formula. We can enumerate all the possible (8) cases and statically resolve the quantification.

Static quantification *TCSPU*

Idea: Exploit Hole Encoding for *TCSPU* to statically resolve quantifiers in the Eager $\forall$ elimination encoding.

Approach

Hole Encoding gives us a 2-CNF formula. We can enumerate all the possible (8) cases and statically resolve the quantification.

Cases

Let $b_i, b_j \in X_c$, $e_i, e_j \in X_u$.
The only possible clauses in the Hole Encoding are in the form:

- $(b_i - b_j) \leq k$
- $(e_i - b_j) \leq k$
- $(b_i - e_j) \leq k$
- $(e_i - e_j) \leq k$
- $(b_i - b_j) \leq k_1 \vee (b_i - b_j) \geq k_2$
- $(e_i - b_j) \leq k_1 \vee (e_i - b_j) \geq k_2$
- $(b_i - e_j) \leq k_1 \vee (b_i - e_j) \geq k_2$
- $(e_i - e_j) \leq k_1 \vee (e_i - e_j) \geq k_2$

Static quantification *TCSPU* (Example)

Let $b \in X_c$, $e \in X_u$ and let y_e be the offset for e.
Let C be a hole-encoded clause of the *TCSPU* problem.

$$C \doteq (b - e) \leq u \vee (b - e) \geq l$$

Static quantification *TCSPU* (Example)

Let $b \in X_c$, $e \in X_u$ and let y_e be the offset for e.
Let C be a hole-encoded clause of the *TCSPU* problem.

$$C \doteq (b - e) \leq u \vee (b - e) \geq l$$

In the eager $\forall$ elimination encoding we have

$$\begin{aligned}&\neg\exists y_e.((y \geq 0) \wedge (y \leq u_e - l_e) \wedge \\ &\quad \neg(((b - (b_e + u - y_e)) \leq u) \vee ((b - (b_e + u - y_e)) \geq l)).\end{aligned}$$

Static quantification *TCSPU* (Example)

Let $b \in X_c$, $e \in X_u$ and let y_e be the offset for e.
Let C be a hole-encoded clause of the *TCSPU* problem.

$$C \doteq (b - e) \leq u \vee (b - e) \geq l$$

In the eager $\forall$ elimination encoding we have

$$\begin{aligned} &\neg\exists y_e.((y \geq 0) \wedge (y \leq u_e - l_e) \wedge \\ &\quad \neg(((b - (b_e + u - y_e)) \leq u) \vee ((b - (b_e + u - y_e)) \geq l)). \end{aligned}$$

The formula can be **statically** simplified

$$\begin{aligned} R \doteq &((l - b + b_e + u_e \leq 0) \vee (l - b + b_e + l_e > 0)) \wedge \\ &((l - b + b_e + l_e < 0) \vee (b - b_e - u - l_e \leq 0)) \end{aligned}$$

Static quantification *TCSPU* (Example)

Let $b \in X_c$, $e \in X_u$ and let y_e be the offset for e.
Let C be a hole-encoded clause of the *TCSPU* problem.

$$C \doteq (b - e) \leq u \vee (b - e) \geq l$$

In the eager $\forall$ elimination encoding we have

$$\begin{aligned} &\neg \exists y_e.((y \geq 0) \wedge (y \leq u_e - l_e) \wedge \\ &\quad \neg(((b - (b_e + u - y_e)) \leq u) \vee ((b - (b_e + u - y_e)) \geq l)). \end{aligned}$$

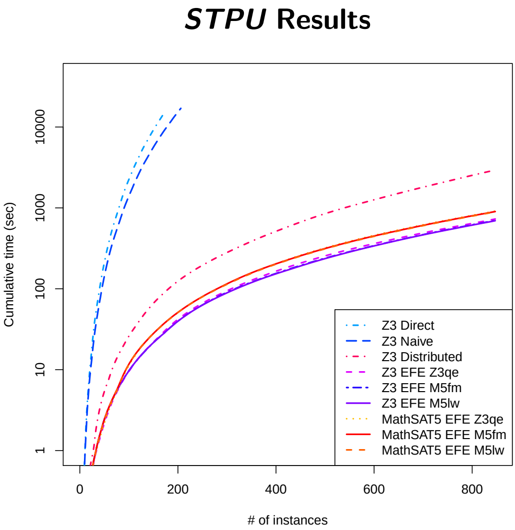
The formula can be **statically** simplified

$$\begin{aligned} R \doteq\, &((l - b + b_e + u_e \leq 0) \vee (l - b + b_e + l_e > 0)) \wedge \\ &((l - b + b_e + l_e < 0) \vee (b - b_e - u - l_e \leq 0)) \end{aligned}$$

Whenever a clause matches the structure of C we can derive $\psi_h^\Gamma(\vec{X}_{c_h})$ by substituting appropriate values for l, u, b_e, l_e and u_e in R.

Outline

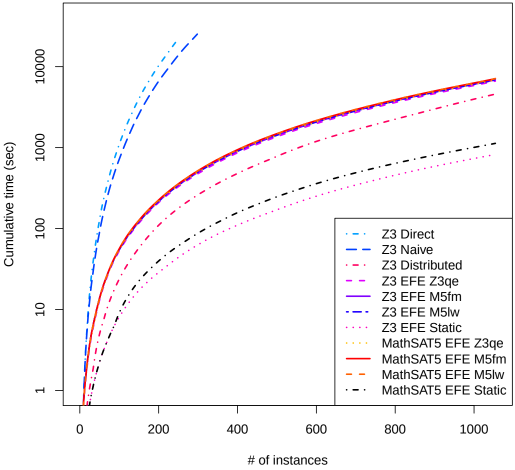
Strong Controllability Results



- Random instance generator
- *SMT* solvers:
  - Z3 ($\mathcal{QF\_LRA}$, $\mathcal{LRA}$)
  - MathSAT5 ($\mathcal{QF\_LRA}$)
- Quantification techniques:
  - Z3 simplifier
  - Fourier-Motzkin
  - Loos-Weispfenning
  - Static quantification for *TCSPU*

Strong Controllability Results

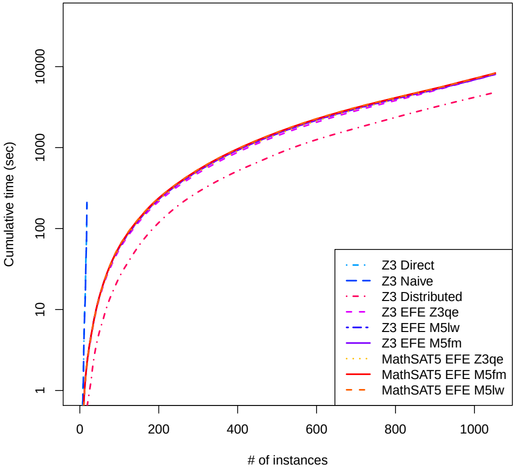
TCSPU **Results**



- Random instance generator
- *SMT* solvers:
  - Z3 ($\mathcal{QF\_LRA}$, $\mathcal{LRA}$)
  - MathSAT5 ($\mathcal{QF\_LRA}$)
- Quantification techniques:
  - Z3 simplifier
  - Fourier-Motzkin
  - Loos-Weispfenning
  - Static quantification for *TCSPU*

Strong Controllability Results

DTPU Results



- Random instance generator
- *SMT* solvers:
  - Z3 ($\mathcal{QF\_LRA}$, $\mathcal{LRA}$)
  - MathSAT5 ($\mathcal{QF\_LRA}$)
- Quantification techniques:
  - Z3 simplifier
  - Fourier-Motzkin
  - Loos-Weispfenning
  - Static quantification for *TCSPU*

Outline

1. Temporal Problems (with Uncertainty)
2. Background
3. Strong Controllability via SMT
   - DTPU encodings
   - TCSPU specific encodings
   - Experimental Evaluation
4. Weak Controllability via SMT
   - Decision Problem
   - Strategies
   - Linear strategies
   - Piecewise linear strategies
   - Experimental Evaluation
5. Conclusion

Outline

Naïve Encoding of the Decision Problem

Intuition

The definition of Weak Controllability, can be seen as a $SMT(\mathcal{LRA})$ formula by interpreting time points as real variables.

Naïve Encoding of the Decision Problem

Intuition

The definition of Weak Controllability, can be seen as a $SMT(\mathcal{LRA})$ formula by interpreting time points as real variables.

Naïve Encoding

A temporal problem is weakly controllable if and only if

$$\forall \vec{X}_u . \exists \vec{X}_c . (C_c(\vec{X}_c, \vec{X}_u) \rightarrow C_f(\vec{X}_c, \vec{X}_u))$$

is valid.

Decision Encodings

Naïve Rewritten Encoding: $\forall \vec{Y}_u . \exists \vec{X}_c . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$

Decision Encodings

Naïve Rewritten Encoding: $\forall \vec{Y}_u . \exists \vec{X}_c . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$

Inverted Encoding

The problem is Weakly Controllable if and only if the formula

$$\neg \exists \vec{X}_c . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$$

is unsatisfiable.

Decision Encodings

Naïve Rewritten Encoding: $\forall \vec{Y}_u . \exists \vec{X}_c . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$

Inverted Encoding

The problem is Weakly Controllable if and only if the formula

$$\neg \exists \vec{X}_c . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$$

is unsatisfiable.

Gamma Extraction Encoding

The problem is Weakly Controllable if and only if the formula

$$\Gamma(\vec{Y}_u) \wedge \neg(\exists \vec{X}_c . \Psi(\vec{X}_c, \vec{Y}_u)).$$

is unsatisfiable.

Decision Encodings

Naïve Rewritten Encoding: $\forall \vec{Y}_u . \exists \vec{X}_c . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$

Inverted Encoding

The problem is Weakly Controllable if and only if the formula

$$\neg \exists \vec{X}_c . (\Gamma(\vec{Y}_u) \rightarrow \Psi(\vec{X}_c, \vec{Y}_u))$$

is unsatisfiable.

Gamma Extraction Encoding

The problem is Weakly Controllable if and only if the formula

$$\Gamma(\vec{Y}_u) \wedge \neg(\exists \vec{X}_c . \Psi(\vec{X}_c, \vec{Y}_u)).$$

is unsatisfiable.

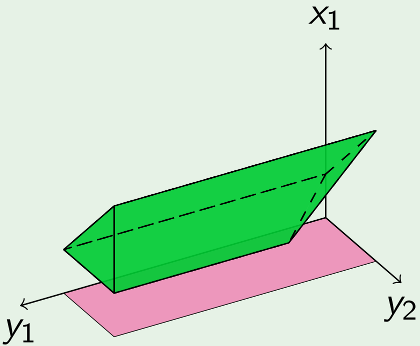
The Inverted Encodings can be solved by any SMT($\mathcal{LRA}$) solver.

Outline

Strategies: Intuition

Given the solution space **P**, in the space of $\vec{X}_c$ and $\vec{Y}_u$, a strategy is a (possibly non-continuous) surface S, such that $P \cap S$ projected in the space of $\vec{Y}_u$ only, contains the polyhedron $\Gamma(\vec{Y}_u)$.

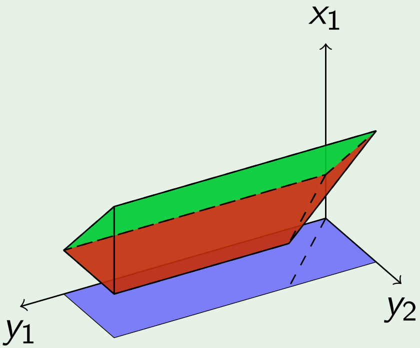
Example



Strategies: Intuition

Given the solution space **P**, in the space of $\vec{X}_c$ and $\vec{Y}_u$, a strategy is a (possibly non-continuous) surface **S**, such that $P \cap S$ **projected in the space of** $\vec{Y}_u$ only, contains the polyhedron $\Gamma(\vec{Y}_u)$.

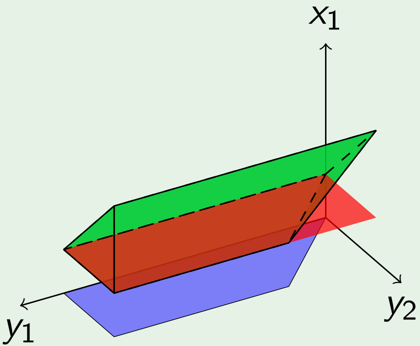
Example



Strategies: Intuition

Given the solution space **P**, in the space of $\vec{X}_c$ and $\vec{Y}_u$, a strategy is a (possibly non-continuous) surface **S**, such that $P \cap S$ **projected in the space of $\vec{Y}_u$** only, contains the polyhedron $\Gamma(\vec{Y}_u)$.

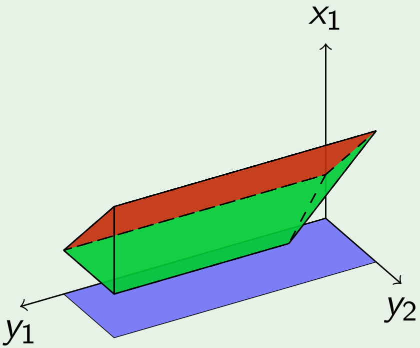
Example



Strategies: Intuition

Given the solution space **P**, in the space of $\vec{X}_c$ and $\vec{Y}_u$, a strategy is a (possibly non-continuous) surface **S**, such that $P \cap S$ **projected in the space of** $\vec{Y}_u$ only, contains the polyhedron $\Gamma(\vec{Y}_u)$.

Example



If S is a (hyper-)plane, the strategy is linear, i.e.

$$f(\vec{Y}_u) \doteq A \cdot \vec{Y}_u + \vec{b}$$

Outline

Linearity is not enough

Theorem

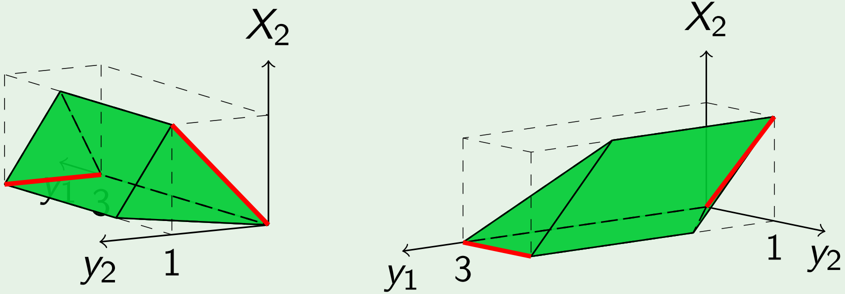
Not every weakly controllable STPU admits a linear strategy.

Linearity is not enough

Theorem

Not every weakly controllable STPU admits a linear strategy.

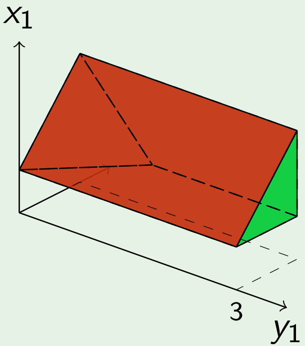
Example



Linear Strategy Encoding in *SMT* ($\mathcal{QF\_LRA}$) for *STPU*

Idea: exploit the convexity of the *STPU*

Example

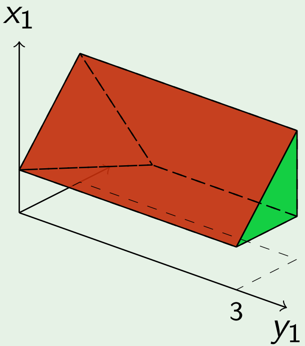


$$\Gamma(y_1, y_2) \doteq y_1 \geq 0 \wedge y_1 \geq 0 \wedge$$
$$y_1 \leq 3 \wedge y_2 \leq 1$$

Linear Strategy Encoding in *SMT* ($\mathcal{QF\_LRA}$) for *STPU*

Idea: exploit the convexity of the *STPU*

Example



$$\Gamma(y_1, y_2) \doteq y_1 \geq 0 \wedge y_1 \geq 0 \wedge$$
$$y_1 \leq 3 \wedge y_2 \leq 1$$

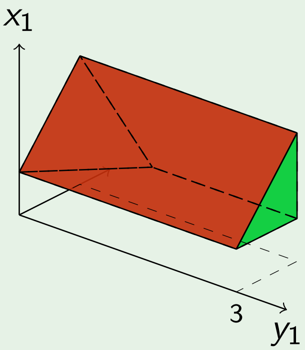
$$f(\vec{Y}_u) \doteq (a_1\ a_2) \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + b$$

$$\exists a_1, a_2, b.\ \forall y_1, y_2.$$
$$\Gamma(y_1, y_2) \rightarrow \Psi(f(a_1, a_2, b), y_1, y_2)$$

Linear Strategy Encoding in *SMT* ($\mathcal{QF\_LRA}$) for *STPU*

Idea: exploit the convexity of the *STPU*

Example



$$\Gamma(y_1, y_2) \doteq y_1 \geq 0 \wedge y_1 \geq 0 \wedge$$
$$y_1 \leq 3 \wedge y_2 \leq 1$$

$$f(\vec{Y}_u) \doteq (a_1 \ a_2) \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + b$$

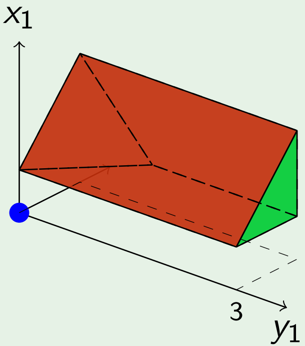
$$\exists a_1, a_2, b. \forall y_1, y_2.$$
$$\Gamma(y_1, y_2) \rightarrow \Psi(f(a_1, a_2, b), y_1, y_2)$$

$$Enc(a_1, a_2, c) \doteq$$

Linear Strategy Encoding in *SMT* ($\mathcal{QF\_LRA}$) for *STPU*

Idea: exploit the convexity of the *STPU*

Example



$$\Gamma(y_1, y_2) \doteq y_1 \geq 0 \wedge y_1 \geq 0 \wedge \\ y_1 \leq 3 \wedge y_2 \leq 1$$

$$f(\vec{Y}_u) \doteq (a_1\ a_2) \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + b$$

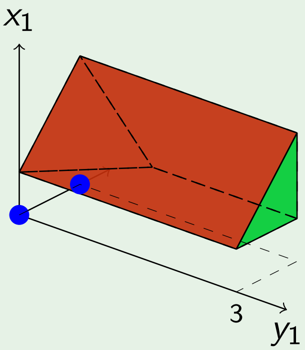
$$\exists a_1, a_2, b. \forall y_1, y_2. \\ \Gamma(y_1, y_2) \rightarrow \Psi(f(a_1, a_2, b), y_1, y_2)$$

$$Enc(a_1, a_2, c) \doteq \Psi(0a_1 + 0a_2 + b,\ 0,\ 0) \wedge$$

Linear Strategy Encoding in *SMT* ($\mathcal{QF\_LRA}$) for *STPU*

Idea: exploit the convexity of the *STPU*

Example



$$\Gamma(y_1, y_2) \doteq y_1 \geq 0 \wedge y_1 \geq 0 \wedge \\ y_1 \leq 3 \wedge y_2 \leq 1$$

$$f(\vec{Y}_u) \doteq (a_1\ a_2) \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + b$$

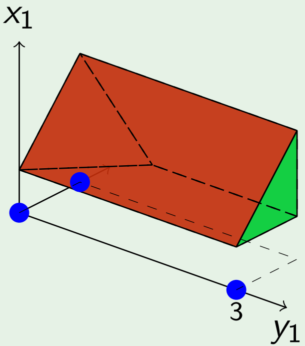
$$\exists a_1, a_2, b. \forall y_1, y_2. \\ \Gamma(y_1, y_2) \rightarrow \Psi(f(a_1, a_2, b), y_1, y_2)$$

$$Enc(a_1, a_2, c) \doteq \Psi(0a_1 + 0a_2 + b,\ 0,\ 0) \wedge \\ \Psi(0a_1 + 1a_2 + b,\ 0,\ 1) \wedge$$

Linear Strategy Encoding in *SMT* ($\mathcal{QF\_LRA}$) for *STPU*

Idea: exploit the convexity of the *STPU*

Example



$$\Gamma(y_1, y_2) \doteq y_1 \geq 0 \wedge y_1 \geq 0 \wedge$$
$$y_1 \leq 3 \wedge y_2 \leq 1$$

$$f(\vec{Y}_u) \doteq (a_1 \; a_2) \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + b$$

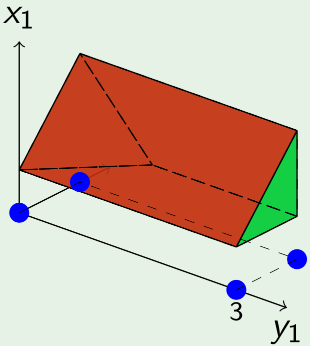
$$\exists a_1, a_2, b. \forall y_1, y_2.$$
$$\Gamma(y_1, y_2) \rightarrow \Psi(f(a_1, a_2, b), y_1, y_2)$$

$$\begin{aligned} Enc(a_1, a_2, c) \doteq\ & \Psi(0a_1 + 0a_2 + b,\ 0,\ 0) \wedge \\ & \Psi(0a_1 + 1a_2 + b,\ 0,\ 1) \wedge \\ & \Psi(3a_1 + 0a_2 + b,\ 3,\ 0) \wedge \end{aligned}$$

Linear Strategy Encoding in *SMT* ($\mathcal{QF\_LRA}$) for *STPU*

Idea: exploit the convexity of the *STPU*

Example



$$\Gamma(y_1, y_2) \doteq y_1 \geq 0 \wedge y_1 \geq 0 \wedge$$
$$y_1 \leq 3 \wedge y_2 \leq 1$$

$$f(\vec{Y}_u) \doteq (a_1\ a_2) \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + b$$

$$\exists a_1, a_2, b. \forall y_1, y_2.$$
$$\Gamma(y_1, y_2) \rightarrow \Psi(f(a_1, a_2, b), y_1, y_2)$$

$$\begin{aligned} Enc(a_1, a_2, c) \doteq\ & \Psi(0a_1 + 0a_2 + b,\ 0,\ 0) \wedge \\ & \Psi(0a_1 + 1a_2 + b,\ 0,\ 1) \wedge \\ & \Psi(3a_1 + 0a_2 + b,\ 3,\ 0) \wedge \\ & \Psi(3a_1 + 1a_2 + b,\ 3,\ 1) \end{aligned}$$

Encoding in *SMT* ($\mathcal{QF\_LRA}$)

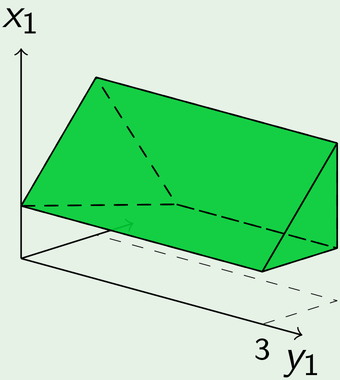
Idea

Search for the coefficients fulfilling a linear strategy in all the extreme assignments (bounds) of $\Gamma(\vec{Y}_u)$.

procedure $\textsc{LinearStrategy}(\Gamma(\vec{Y}_u), \Psi(\vec{X}_c, \vec{Y}_u))$
  $p \leftarrow \textsc{VariableMatrix}(|X_c|, |X_u| + 1)$
  $\phi(p) \leftarrow \top$
  for all $\vec{c} \in \textsc{ExtremeAssignments}(\Gamma(\vec{Y}_u))$ **do**
    $\phi(p) \leftarrow \phi(p) \wedge \Psi(p \cdot \vec{c}, \vec{c})$
  if $\mathrm{SMT}(\phi(p))$ **then**
    return $\textsc{GetModel}()$
  return $\perp$

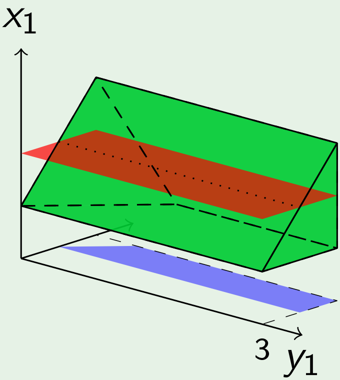
Incremental Weakening for *STPU*

Example



Incremental Weakening for *STPU*

Example

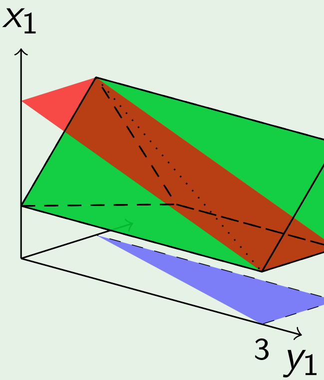


Observed uncontrollable offsets:

- $\emptyset$
- $\{y_1\}$
- $\{y_2\}$
- $\{y_1, y_2\}$

Incremental Weakening for *STPU*

Example

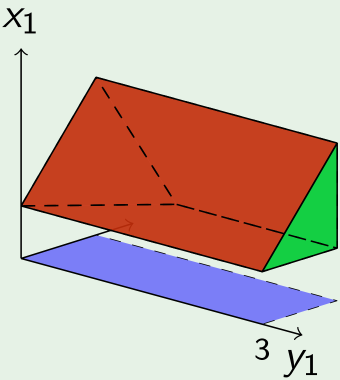


Observed uncontrollable offsets:

- $\emptyset$
- $\{y_1\}$
- $\{y_2\}$
- $\{y_1, y_2\}$

Incremental Weakening for *STPU*

Example

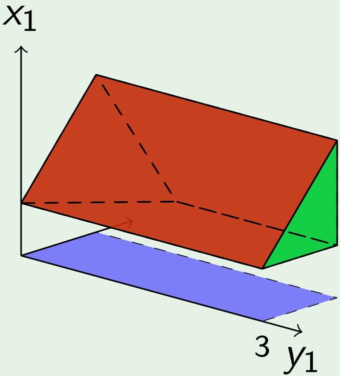


Observed uncontrollable offsets:

- $\emptyset$
- $\{y_1\}$
- $\{y_2\}$
- $\{y_1, y_2\}$

Incremental Weakening for *STPU*

Example



Observed uncontrollable offsets:

- $\emptyset$
- $\{y_1\}$
- $\{y_2\}$
- $\{y_1, y_2\}$

Intuition

If we do not observe the *i*-th variable, the *i*-th column in the matrix is filled with 0.

Incremental Weakening

Idea

Start from strong controllability check (polynomial), and search for a strategy depending on a subset of $\vec{Y}_u$.

procedure $\textsc{IWLinearStrategy}(\Gamma(\vec{Y}_u), \Psi(\vec{X}_c, \vec{Y}_u))$
    while $\vec{p} \leftarrow \textsc{GetHeuristicPivots}(\Gamma(\vec{Y}_u))$ **do**
        $\vec{n} \leftarrow \{y \in \vec{Y}_u \mid y \notin \vec{p}\}$
        $\eta(\vec{X}_c, \vec{p}) \leftarrow \textsc{SC\_Enc}(\Gamma(\vec{n}), \Psi(\vec{X}_c, \vec{n}))$
        $res \leftarrow \textsc{LinearStrategy}(\Gamma(\vec{p}), \eta(\vec{X}_c, \vec{p}))$
        if $res \neq$ None **then**
            return res
    return $\perp$

Outline

Definitions

Piecewise linear strategies

f is a piecewise linear strategy if it has the form

$$\begin{aligned} f(\vec{Y}_u) \doteq\ & \text{If } \phi_1(\vec{Y}_u) \text{ then } A_1 \cdot \vec{Y}_u + \vec{b}_1; \\ & \text{If } \phi_2(\vec{Y}_u) \text{ then } A_2 \cdot \vec{Y}_u + \vec{b}_2; \\ & \ldots \\ & \text{If } \phi_n(\vec{Y}_u) \text{ then } A_n \cdot \vec{Y}_u + \vec{b}_n; \end{aligned}$$

Definitions

Piecewise linear strategies

f is a piecewise linear strategy if it has the form

$$\begin{aligned} f(\vec{Y}_u) \doteq \ & \text{If } \phi_1(\vec{Y}_u) \text{ then } A_1 \cdot \vec{Y}_u + \vec{b}_1; \\ & \text{If } \phi_2(\vec{Y}_u) \text{ then } A_2 \cdot \vec{Y}_u + \vec{b}_2; \\ & \ldots \\ & \text{If } \phi_n(\vec{Y}_u) \text{ then } A_n \cdot \vec{Y}_u + \vec{b}_n; \end{aligned}$$

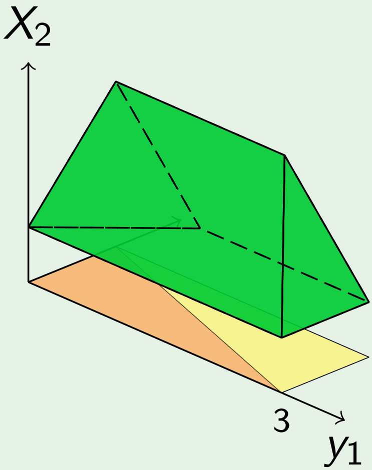
Simplexes

An n-simplex is an n-dimensional polytope which is the convex hull of its $n+1$ vertexes. E.g.

- 2-d $\rightarrow$ Triangle
- 3-d $\rightarrow$ Tetrahedron

Enumerating all the maximal simplexes for *STPU*

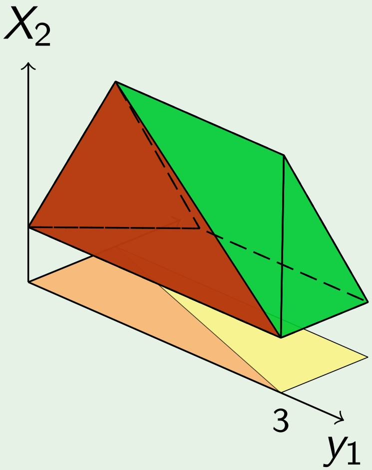
Example



Strategy:

Enumerating all the maximal simplexes for *STPU*

Example

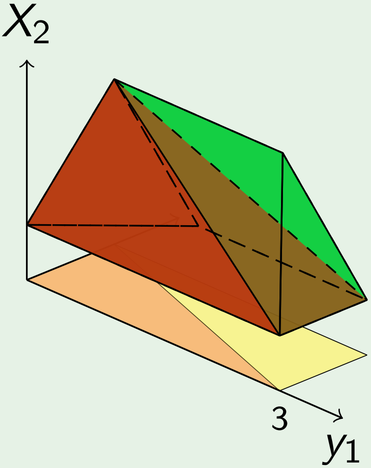


Strategy:

$$\text{If } y_2 \leq -\frac{1}{3}y_1 + 1 \text{ then } S_1;$$

Enumerating all the maximal simplexes for *STPU*

Example



Strategy:

If $y_2 \leq -\frac{1}{3}y_1 + 1$ then S_1;

If $y_2 > -\frac{1}{3}y_1 + 1$ then S_2;

All simplexes

Idea

Decompose the solution space in simplexes, and search a linear strategy for each of them.

procedure $\textsc{GetStrategy}(\Gamma(\vec{Y}_u), \Psi(\vec{X}_c, \vec{Y}_u))$

  $p \leftarrow \emptyset$

  for all $s(\vec{Y}_u) \in \textsc{ExtremalSimplexes}(\Gamma(\vec{Y}_u))$ **do**

    $l \leftarrow \textsc{GetLinearStrategy}(s(\vec{Y}_u)\ \Psi(\vec{X}_c, \vec{Y}_u))$

    $p \leftarrow p \cup \{(\text{ ``}\textsc{if}\ s(\vec{Y}_u)\ \textsc{then}\ l\text{ '' })\}$

  return p

Lazy extraction for *DTPU*

Idea

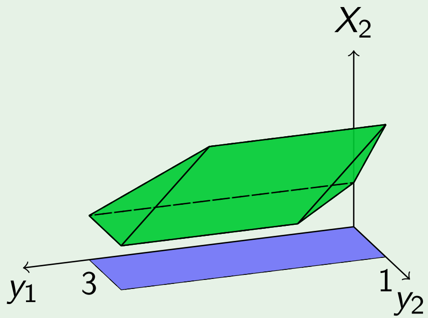
Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Lazy extraction for *DTPU*

Idea

Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Example



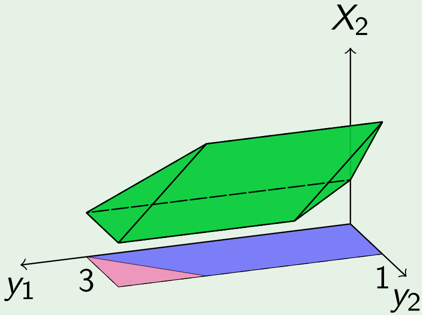
Strategy:

Lazy extraction for *DTPU*

Idea

Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Example



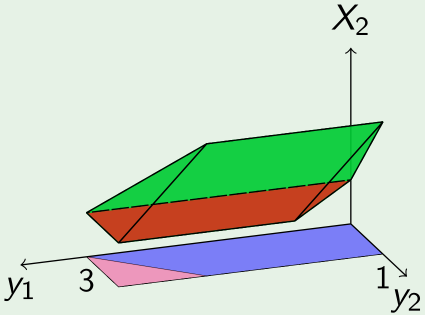
Strategy:

Lazy extraction for *DTPU*

Idea

Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Example



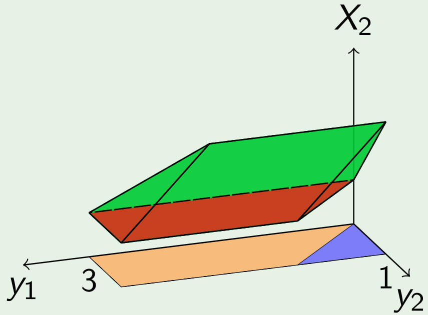
Strategy:

Lazy extraction for *DTPU*

Idea

Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Example



Strategy:

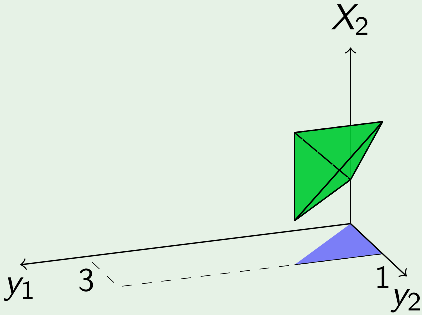
If $y_1 \geq y_2$ then S_1;

Lazy extraction for *DTPU*

Idea

Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Example



Strategy:

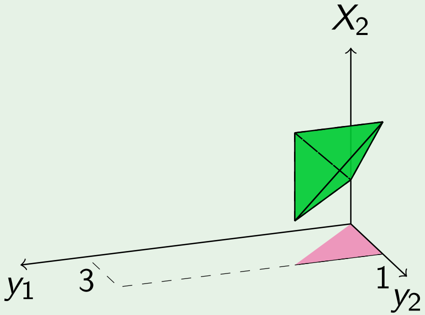
If $y_1 \geq y_2$ then S_1;

Lazy extraction for *DTPU*

Idea

Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Example



Strategy:

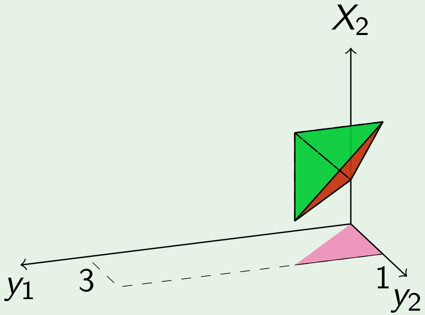
If $y_1 \geq y_2$ then S_1;

Lazy extraction for *DTPU*

Idea

Pick a simplex $R(\vec{Y}_u)$ in $\Gamma(\vec{Y}_u)$, find a linear strategy $S(\vec{Y}_u)$ for $R(\vec{Y}_u)$, and remove the region where $S(\vec{Y}_u)$ is applicable from $\Gamma(\vec{Y}_u)$. Iterate until $\Gamma(\vec{Y}_u)$ is empty.

Example



Strategy:

If $y_1 \geq y_2$ then S_1;

If $y_1 < y_2$ then S_2;

Lazy extraction

Idea

Find a strategy S for a simplex, and remove from $\Gamma(\vec{Y}_u)$ all the region satisfied by S.

procedure $\textsc{GetStrategy}(\Gamma(\vec{Y}_u), \Psi(\vec{X}_c, \vec{Y}_u))$

  $p \leftarrow \emptyset$

  $\eta(\vec{Y}_u) \leftarrow \Gamma(\vec{Y}_u)$

  while $\mathrm{SMT}(\eta(\vec{Y}_u))$ **do**

    $s(\vec{Y}_u) \leftarrow \textsc{Simplex}(\eta(\vec{Y}_u))$

    $strategy(\vec{Y}_u) \leftarrow \textsc{GetLinearStrategy}(s(\vec{Y}_u), \Psi(\vec{X}_c, \vec{Y}_u))$

    $covered(\vec{Y}_u) \leftarrow \Psi(strategy(\vec{Y}_u), \vec{Y}_u)$

    $p \leftarrow p \cup \{(\text{“}\textsc{if}\ \eta(\vec{Y}_u) \wedge covered(\vec{Y}_u)\ \textsc{then}\ strategy\text{”})\}$

    $\eta(\vec{Y}_u) \leftarrow \eta(\vec{Y}_u) \wedge \neg\Psi(s(\vec{Y}_u), \vec{Y}_u)$

  return p

Skin Crawler for *DTPU*

Idea

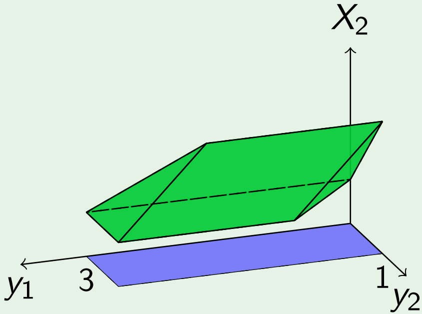
Explore the faces of the solution space projecting them until the entire uncontrollable space is covered.
This builds a piecewise-linear strategy by combining hyperplanes containing at least one facets of the solution space.

Skin Crawler for *DTPU*

Idea

Explore the faces of the solution space projecting them until the entire uncontrollable space is covered.
This builds a piecewise-linear strategy by combining hyperplanes containing at least one facets of the solution space.

Example



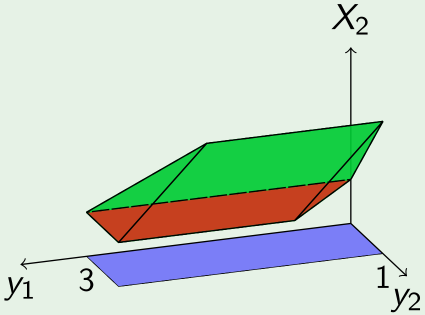
Strategy:

Skin Crawler for *DTPU*

Idea

Explore the faces of the solution space projecting them until the entire uncontrollable space is covered.
This builds a piecewise-linear strategy by combining hyperplanes containing at least one facets of the solution space.

Example



Strategy:

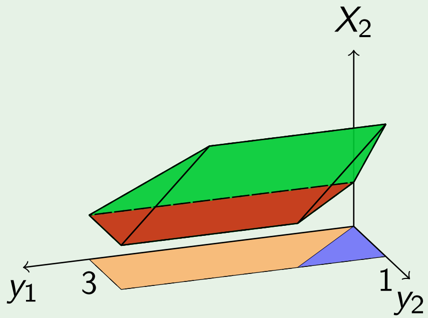
If $y_1 \geq y_2$ then S_1;

Skin Crawler for *DTPU*

Idea

Explore the faces of the solution space projecting them until the entire uncontrollable space is covered.
This builds a piecewise-linear strategy by combining hyperplanes containing at least one facets of the solution space.

Example



Strategy:

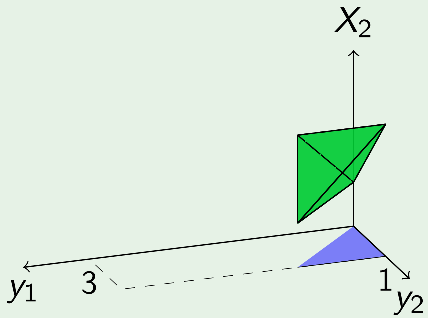
If $y_1 \geq y_2$ then S_1;

Skin Crawler for *DTPU*

Idea

Explore the faces of the solution space projecting them until the entire uncontrollable space is covered.
This builds a piecewise-linear strategy by combining hyperplanes containing at least one facets of the solution space.

Example



Strategy:

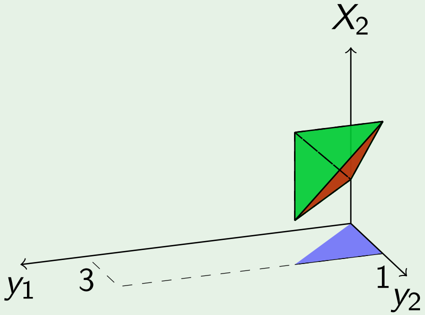
If $y_1 \geq y_2$ then S_1;

Skin Crawler for *DTPU*

Idea

Explore the faces of the solution space projecting them until the entire uncontrollable space is covered.
This builds a piecewise-linear strategy by combining hyperplanes containing at least one facets of the solution space.

Example



Strategy:

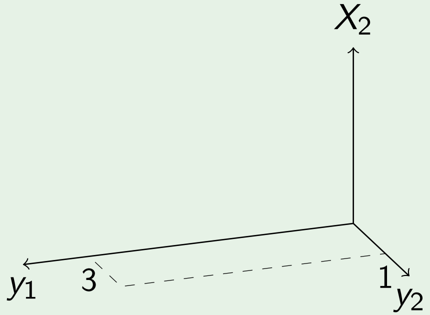
If $y_1 \geq y_2$ then S_1;
If $y_1 < y_2$ then S_2;

Skin Crawler for *DTPU*

Idea

Explore the faces of the solution space projecting them until the entire uncontrollable space is covered.
This builds a piecewise-linear strategy by combining hyperplanes containing at least one facets of the solution space.

Example



Strategy:

If $y_1 \geq y_2$ then S_1;
If $y_1 < y_2$ then S_2;

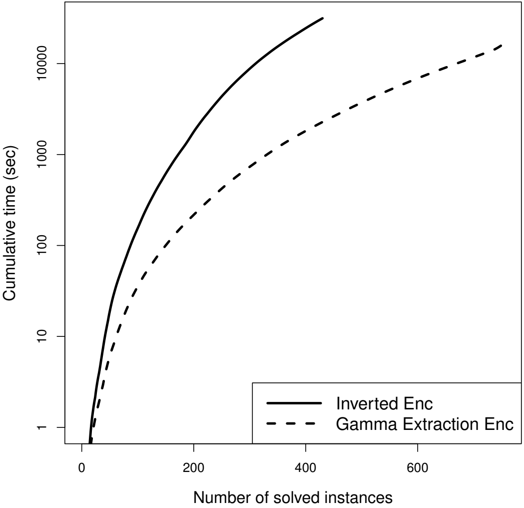
Skin Crawler

procedure $\textsc{SkinCrawlingSE}(\Gamma(\vec{Y_u}), \Psi(\vec{X_c}, \vec{Y_u}))$
 $\textsc{Assert}(\Psi(\vec{X_c}, \vec{Y_u}) \wedge \Gamma(\vec{Y_u}))$
 for all $eq_i(\vec{X_c}, \vec{Y_u}) \in \mathit{Equalities}(\Psi(\vec{X_c}, \vec{Y_u}))$ **do**
  $\textsc{Assert}(eq_i(\vec{X_c}, \vec{Y_u}) \rightarrow (eqv_i = 1))$
  $\textsc{Assert}(eq_i(\vec{X_c}, \vec{Y_u}) \vee (eqv_i = 0))$
  $\mathit{EQVs} \leftarrow \mathit{EQVs} \cup \{eqv_i\}$
 $\mathit{strategy} \leftarrow \emptyset$; $\mathit{currentCost} \leftarrow |\mathit{EQVs}|$
 while $\mathit{currentCost} > 0$ **do**
  if $\mathrm{SMT}() = \mathit{UNSAT}$ **then return** $\mathit{strategy}$
  $\textsc{Push}()$
  $\textsc{Assert}(\mathit{cost} = \mathit{currentCost})$
  if $\mathrm{SMT}() = \mathit{SAT}$ **then**
   $\mathit{model} \leftarrow \textsc{GetModel}()$
   for all $eq_i(\vec{X_c}, \vec{Y_u}) \in \mathit{Equalities}(\Psi(\vec{X_c}, \vec{Y_u}))$ **do**
    if $\mathit{model} \models eq_i(\vec{X_c}, \vec{Y_u})$ **then** $\mathit{system} \leftarrow \mathit{system} \cup \{eq_i(\vec{X_c}, \vec{Y_u})\}$
   $\textsc{Pop}()$
   $(\mathit{covered}(\vec{Y_u}), \mathit{subStartegy}) \leftarrow \textsc{GetFaceStrategy}(\mathit{strategy}, \mathit{system})$
   $\mathit{strategy} \leftarrow \mathit{strategy} \cup \{$ "if $\mathit{covered}(\vec{Y_u})$ then $\mathit{subStartegy}$ "$\}$
   $\textsc{Assert}((\bigvee_{s(\vec{X_c}, \vec{Y_u}) \in \mathit{system}} \neg s(\vec{X_c}, \vec{Y_u})) \wedge (\neg \mathit{covered}(\vec{Y_u})))$
  else
   $\mathit{currentCost} \leftarrow \mathit{currentCost} - 1$
   $\textsc{Pop}()$
 return $\perp$

Outline

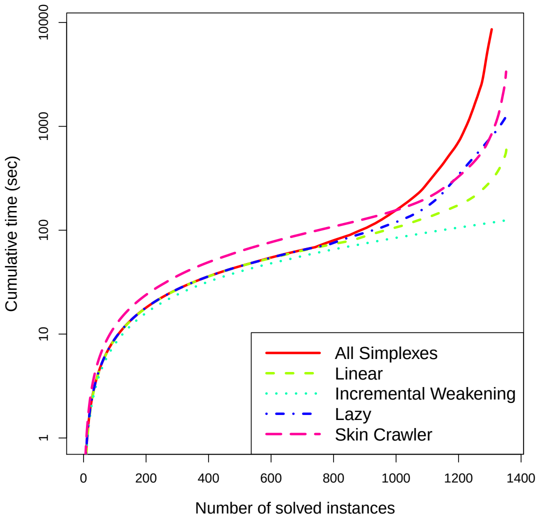
Decision problem

- Random instances
- Z3 SMT Solver

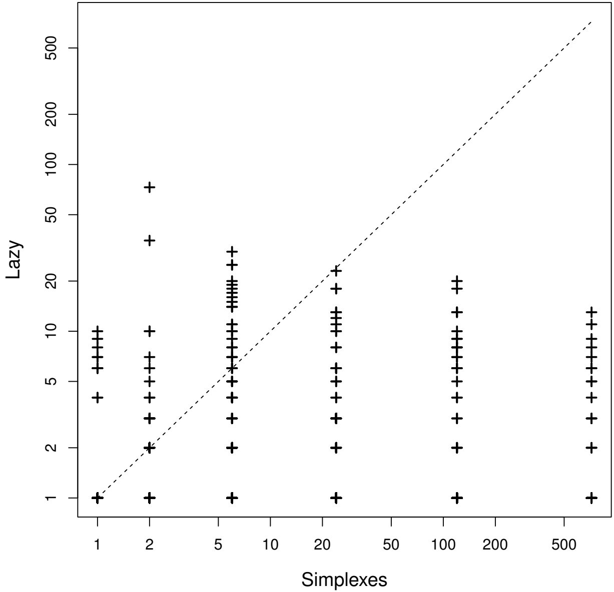


Scalability of strategy extraction algorithms

- Random instance generator derived from TSAT++ experiments
- Implementation
  - Python implementation
  - MathSAT5 API
- 4 algorithms
  - Linear
  - Incremental Weakening
  - All simplexes
  - Lazy
- 1354 weakly controllable instances admitting a linear strategy



Extracted strategy size



Outline

Conclusions

Summary

- SMT-based techniques for temporal reasoning in presence of uncertainty
- Strong Controllability optimization for TCSPU
- Weak Controllability decision procedures
- Weak Strategies synthesis

Conclusions

Summary

- SMT-based techniques for temporal reasoning in presence of uncertainty
- Strong Controllability optimization for TCSPU
- Weak Controllability decision procedures
- Weak Strategies synthesis

Future works

- Dynamic Controllability
- Cost function optimization
- Incrementality

Thanks

Thanks for your attention!

Bibliography

Bart Peintner, Kristen Brent Venable, and Neil Yorke-Smith. Strong controllability of disjunctive temporal problems with uncertainty. In *Principles and Practice of Constraint Programming - CP*, pages 856–863, 2007.

Thierry Vidal and Hélène Fargier. Handling contingency in temporal constraint networks: from consistency to controllabilities. *Journal of Experimental Theoretical Artificial Intelligence*, 11(1):23–45, 1999.

Backup Slides

Related work

- The strong controllability problem has been introduced in Vidal and Fargier [1999].

Related work

- The strong controllability problem has been introduced in Vidal and Fargier [1999].
- Strong Controllability of DTPUs has been theoretically tackled in Peintner et al. [2007] using Meta-CSP techniques.

Related work

- The strong controllability problem has been introduced in Vidal and Fargier [1999].
- Strong Controllability of DTPUs has been theoretically tackled in Peintner et al. [2007] using Meta-CSP techniques.
- We developed SMT-based encodings also for Weak Controllability decision problem, and a portfolio of SMT-based algorithms for strategy extraction.