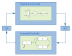
# Generalized Craig Interpolants for Stochastic Satisfiability modulo theory problems

**Ahmed Mahdi**
Martin Fränzle

Research Group Hybrid Systems
Carl von Ossietzky Universität Oldenburg, Germany
Transregional Collaborative Research Center "AVACS",
German Research Council, SFB/TR 14

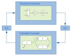
IPRA 2014: Interpolation: From Proofs to Applications
Vienna, Austria, 17.07.2014



# Motivation(1)

**Classical Case:**

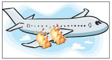
**Real-world system**



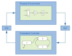
# Motivation(1)

**Classical Case:**

**Real-world system**



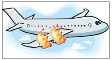
**Safety property; no crash**



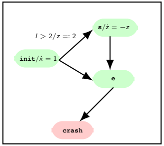
# Motivation(1)

**Classical Case:**

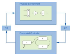
**Real-world system**



**Safety property; no crash**



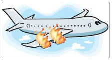

**Formal Model**



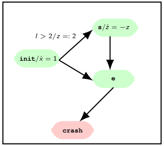
# Motivation(1)

**Classical Case:**

**Real-world system**



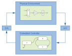
**Safety property; no crash**




**Formal Model**

$\texttt{init}(\vec{s})$
$\texttt{Trans}(\vec{s}, \vec{s}')$
$\texttt{Bad}(\vec{s}')$

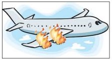
**Mathematical representation**



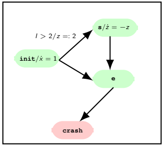
# Motivation(1)

**Classical Case:**

**Real-world system**



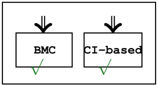
**Safety property; no crash**



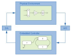

**Formal Model**

init($\vec{s}$)
Trans($\vec{s}, \vec{s}'$)
Bad($\vec{s}'$)

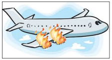
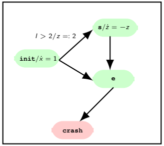
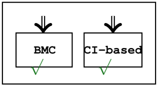
**Mathematical representation**



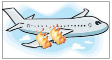
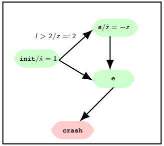
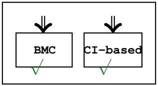

**Verification**



# Motivation(1)

| | Classical Case: | Probabilistic Case: |
|---|---|---|
|  | Real–world system | Real–world system |
|  | Safety property; no crash | |
|  | Formal Model | |
| $\texttt{init}(\vec{s})$<br>$\texttt{Trans}(\vec{s}, \vec{s}')$<br>$\texttt{Bad}(\vec{s}')$ | Mathematical representation | |
|  | Verification | |

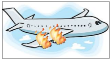
# Motivation(1)

| | Classical Case: | Probabilistic Case: |
|---|---|---|
|  | Real–world system | Real–world system |
|  | Safety property; no crash | Safety property; Pr(crash) $\leq 10^{-9}$ |
|  | Formal Model | |
| init($\vec{s}$)<br>Trans($\vec{s}, \vec{s}'$)<br>Bad($\vec{s}'$) | Mathematical representation | |
|  | Verification | |

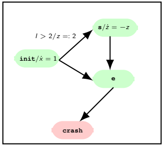
# Motivation(1)

**Classical Case:** | **Probabilistic Case:**

**Real-world system** | **Real-world system**



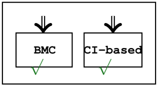
**Safety property; no crash** | **Safety property;** Pr(crash) $\leq 10^{-9}$



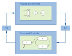

**Formal Model** | **Plus:** nondeterministic **and** probabilistic **choices**

$\texttt{init}(\vec{s})$
$\texttt{Trans}(\vec{s}, \vec{s}')$
$\texttt{Bad}(\vec{s}')$

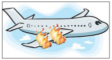
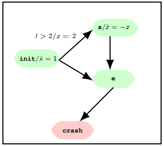
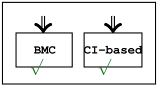
**Mathematical representation**

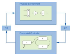



**Verification**

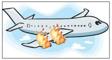
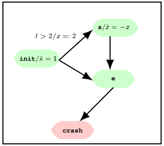
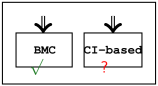


# Motivation(1)

| | Classical Case: | Probabilistic Case: |
|---|---|---|
|  | Real-world system | Real-world system |
|  | Safety property; no crash | Safety property; Pr(crash) $\leq 10^{-9}$ |
|   | Formal Model | Plus: nondeterministic and probabilistic choices |
| init($\vec{s}$) Trans($\vec{s}, \vec{s}'$) Bad($\vec{s}'$) | Mathematical representation | Plus: nondeterministic and probabilistic choices |
|   | Verification | |

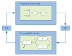


# Motivation(1)

| | Classical Case: | Probabilistic Case: |
|---|---|---|
|  | Real-world system | Real-world system |
|  | Safety property; no crash | Safety property; Pr(crash) $\leq 10^{-9}$ |
|  | Formal Model | Plus: nondeterministic and probabilistic choices |
| init($\vec{s}$) Trans($\vec{s}, \vec{s}'$) Bad($\vec{s}'$) | Mathematical representation | Plus: nondeterministic and probabilistic choices |
|  | Verification | Verification |







# Motivation(2)

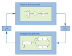
Classical (non-Probabilistic) Case: $TS \models AG\neg p$.


System modelled by a transition system TS.

# Motivation(2)

Classical (non-Probabilistic) Case: $TS \models AG\neg p$.


Our Task: verify that the system does not reach unsafe states.



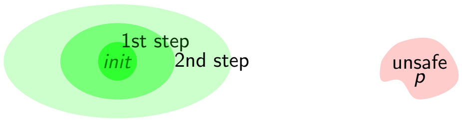
# Motivation(2)

Classical (non-Probabilistic) Case: $TS \models AG\neg p$.


Explore one step (interpolant $\mathcal{I}_1$) further. $\mathcal{I}_1 \models^{?} AG\neg p$

# Motivation(2)

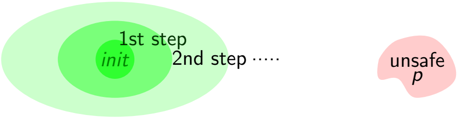
Classical (non-Probabilistic) Case: $TS \models AG\neg p$.




Explore one step ($\mathcal{I}_2$) further. $\mathcal{I}_2 \models^? AG\neg p$.

# Motivation(2)

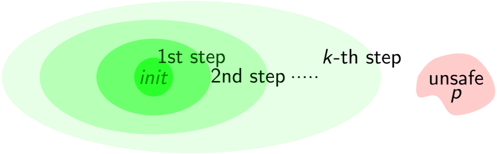
Classical (non-Probabilistic) Case: $TS \models AG\neg p$.




Continue exploring

# Motivation(2)

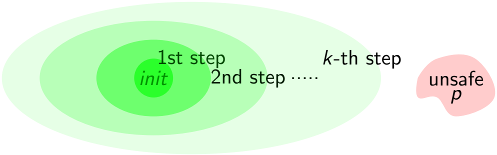
Classical (non-Probabilistic) Case: $TS \models AG\neg p$.




until $\mathcal{I}_k$ stabilizes. $\mathcal{I}_k \models^? AG\neg p$ or $\not\models \mathcal{I}_k \wedge$ *unsafe*.

# Motivation(2)

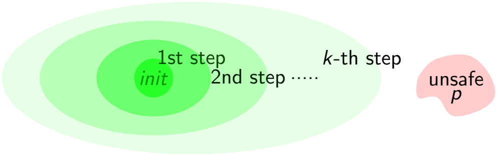
Classical (non-Probabilistic) Case: $TS \models AG\neg p$.




Reachability analysis of *non-probabilistic* finite-state systems based on CI.

# Motivation(2)

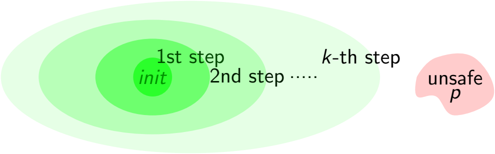
Classical (non-Probabilistic) Case: $TS \models AG\neg p$.



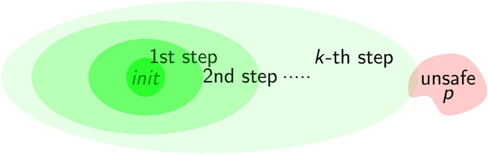

How about to verify that $\mathtt{Pr}(TS \wedge p) \leq \theta$!!

# Motivation(2)

Classical (non-Probabilistic) Case: $TS \models AG\neg p$.




Probabilistic Case: $\mathtt{Pr}(\mathcal{I}_k \wedge p) \leq \theta$



# Outline

- Craig interpolation
- SSMT problems
- Resolution Calculus for SSMT problems
- Generalized Craig interpolation for SSAT and SSMT problems
- Conclusion and future work.

# Craig Interpolation

# Craig interpolation

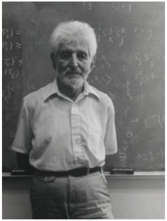


Figure: William Craig, 1957

**Theorem 1 (Craig Interpolation [Cra57])**

*Let $A$ and $B$ be closed FOF. If $A \rightarrow B$ is valid, there exists a formula $\mathcal{I}$ such that:*

- $A \rightarrow \mathcal{I}$
- $\mathcal{I} \rightarrow B$
- $Var(\mathcal{I}) \subseteq Var(A) \cap Var(B)$.

# Craig interpolation

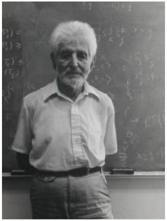
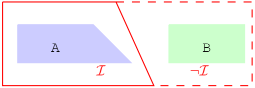


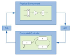
Figure: William Craig, 1957

**Theorem 1 (Craig Interpolation [Cra57])**

*Let $A$ and $B$ be closed FOF. If $A \rightarrow B$ is valid, there exists a formula $\mathcal{I}$ such that:*

- $A \rightarrow \mathcal{I}$
- $\mathcal{I} \rightarrow B$
- $Var(\mathcal{I}) \subseteq Var(A) \cap Var(B)$.

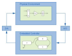


**Example** 1 (Simple Craig interpolant)

*Let* $A = P \wedge Q$, *and* $B = R \rightarrow Q$, *then:*

- $A \rightarrow B$,
- $A \rightarrow Q$,
- $Q \rightarrow B$,
- $Q \subseteq Var(A) \cap Var(B)$, *and*
- $\mathcal{I} = Q$ *(valid Craig interpolant).*



**Example** 1 (Simple Craig interpolant)

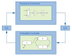
*Let* $A = P \wedge Q$, *and* $B = R \rightarrow Q$, *then:*

- $A \rightarrow B$,
- $A \rightarrow Q$,
- $Q \rightarrow B$,
- $Q \subseteq Var(A) \cap Var(B)$, *and*
- $\mathcal{I} = Q$ *(valid Craig interpolant).*

## Example 1 (Simple Craig interpolant)

*Let* $A = P \wedge Q$, *and* $B = R \rightarrow Q$, *then:*

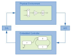
- $A \rightarrow B$,
- $A \rightarrow Q$,
- $Q \rightarrow B$,
- $Q \subseteq Var(A) \cap Var(B)$, *and*
- $\mathcal{I} = Q$ *(valid Craig interpolant).*



## Example 1 (Simple Craig interpolant)

*Let* $A = P \wedge Q$, *and* $B = R \rightarrow Q$, *then:*

- $A \rightarrow B$,
- $A \rightarrow Q$,
- $Q \rightarrow B$,
- $Q \subseteq Var(A) \cap Var(B)$, *and*
- $\mathcal{I} = Q$ *(valid Craig interpolant).*



## Example 1 (Simple Craig interpolant)

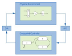
*Let* $A = P \wedge Q$*, and* $B = R \rightarrow Q$*, then:*

- $A \rightarrow B$,
- $A \rightarrow Q$,
- $Q \rightarrow B$,
- $Q \subseteq Var(A) \cap Var(B)$, *and*
- $\mathcal{I} = Q$ *(valid Craig interpolant).*

**Example** 1 (Simple Craig interpolant)

*Let* $A = P \wedge Q$*, and* $B = R \rightarrow Q$*, then:*

- $A \rightarrow B$,
- $A \rightarrow Q$,
- $Q \rightarrow B$,
- $Q \subseteq Var(A) \cap Var(B)$, *and*
- $\mathcal{I} = Q$ *(valid Craig interpolant).*



# Appliation of Craig Interpolation

Craig Interpolations is used as generalizations in:

- consistency proofs,

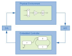
CARL VON OSSIETZKY universität OLDENBURG

# Appliation of Craig Interpolation

Craig Interpolations is used as generalizations in:

- consistency proofs,
- model checking in particular from BMC, to unbounded model checking [McM03]

$A$ $\neg B$



# Appliation of Craig Interpolation

Craig Interpolations is used as generalizations in:

- consistency proofs,
- model checking in particular from BMC, to unbounded model checking [McM03]

$$A \longrightarrow t_0 \longrightarrow t_1 \longrightarrow t_2 \longrightarrow t_3 \longrightarrow \neg B$$

$$\vdots$$

$$\mathcal{I}$$

# Appliation of Craig Interpolation

Craig Interpolations is used as generalizations in:

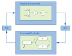
- consistency proofs,
- model checking in particular from BMC, to unbounded model checking [McM03]

$$A \longrightarrow t_0 \longrightarrow t_1 \longrightarrow t_2 \longrightarrow t_3 \longrightarrow \neg B$$

$$\vdots$$

$$\mathcal{I}$$

- Theorem provers [BGKK13].



# Appliation of Craig Interpolation

Craig Interpolations is used as generalizations in:

- consistency proofs,
- model checking in particular from BMC, to unbounded model checking [McM03]

$$A \longrightarrow t_0 \longrightarrow t_1 \longrightarrow t_2 \longrightarrow t_3 \longrightarrow \neg B$$

$$\mathcal{I}$$

- Theorem provers [BGKK13].
- Compositional SMT [AM13].

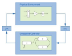
# Stochastic Satisfiability Modulo Theories

# Stochastic Satisfiability Modulo Theories

Stochastic Boolean Satisfiability SSAT [Pap94]

# Stochastic Satisfiability Modulo Theories

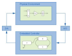
Stochastic Boolean Satisfiability SSAT [Pap94]
= Boolean Satisfiability + Randomized quantifiers



# Stochastic Satisfiability Modulo Theories

Stochastic Boolean Satisfiability SSAT [Pap94]
= Boolean Satisfiability + Randomized quantifiers

Stochastic Satisfiability Modulo Theories SSMT [FHT08]



# Stochastic Satisfiability Modulo Theories

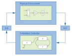
Stochastic Boolean Satisfiability SSAT [Pap94]
= Boolean Satisfiability + Randomized quantifiers

Stochastic Satisfiability Modulo Theories SSMT [FHT08]
= Satisfiability Modulo Theories + Randomized quantifiers

# Stochastic Satisfiability Modulo Theories: Syntax

SSMT formula $\mathcal{Q} : \varphi$

1. prefix $\mathcal{Q}$ of quantified variables
   - $\exists x \in \mathcal{D}_x : \mathcal{D}_x$ is finite. E.g. $\{1, 2, 5, 6\}$
   - $\text{Я} y_{[v_1 \mapsto p_1, \ldots, v_n \mapsto p_n]} : \sum_{i=1}^{n} p_i = 1$. E.g. $\{1 \mapsto 0.5, 2.5 \mapsto 0.21, 7 \mapsto 0.11, 10 \mapsto 0.18\}$
2. SMT formula $\varphi$ (matrix), e.g. $\varphi = (x < 2 \vee \sin(y)) \wedge (a = \texttt{true})\ldots.$



# Stochastic Satisfiability Modulo Theories: Syntax

SSMT formula $\mathcal{Q} : \varphi$

1. prefix $\mathcal{Q}$ of quantified variables
   - $\exists x \in \mathcal{D}_x : \mathcal{D}_x$ is finite. E.g. $\{1, 2, 5, 6\}$
   - $\text{Я} y_{[v_1 \mapsto p_1, \ldots, v_n \mapsto p_n]} : \sum_{i=1}^{n} p_i = 1$. E.g. $\{1 \mapsto 0.5, 2.5 \mapsto 0.21, 7 \mapsto 0.11, 10 \mapsto 0.18\}$
2. SMT formula $\varphi$ (matrix), e.g. $\varphi = (x < 2 \vee \sin(y)) \wedge (a = \texttt{true})\ldots.$

# Stochastic Satisfiability Modulo Theories: Syntax

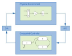
SSMT formula $\mathcal{Q} : \varphi$

1. prefix $\mathcal{Q}$ of quantified variables
   - $\exists x \in \mathcal{D}_x : \mathcal{D}_x$ is finite. E.g. $\{1, 2, 5, 6\}$
   - $\text{Я} y_{[v_1 \mapsto p_1, \ldots, v_n \mapsto p_n]} : \sum_{i=1}^{n} p_i = 1$. E.g. $\{1 \mapsto 0.5, 2.5 \mapsto 0.21, 7 \mapsto 0.11, 10 \mapsto 0.18\}$
2. SMT formula $\varphi$ (matrix), e.g. $\varphi = (x < 2 \vee \sin(y)) \wedge (a = \texttt{true})$....

# Stochastic Satisfiability Modulo Theories: Syntax

SSMT formula $\mathcal{Q} : \varphi$

1. prefix $\mathcal{Q}$ of quantified variables
   - $\exists x \in \mathcal{D}_x : \mathcal{D}_x$ is finite. E.g. $\{1, 2, 5, 6\}$
   - $\text{Я} y_{[v_1 \mapsto p_1, \ldots, v_n \mapsto p_n]} : \sum_{i=1}^{n} p_i = 1$. E.g. $\{1 \mapsto 0.5, 2.5 \mapsto 0.21, 7 \mapsto 0.11, 10 \mapsto 0.18\}$
2. SMT formula $\varphi$ (matrix), e.g. $\varphi = (x < 2 \vee \sin(y)) \wedge (a = \texttt{true})\ldots.$



# SSMT: Quantification

$\exists x : \varphi$ I.e., for some value $\varphi$ holds.
ᴚ$x : \varphi$ I.e., for random values $\varphi$ holds.

Randomized quantification to describe probabilistic events:

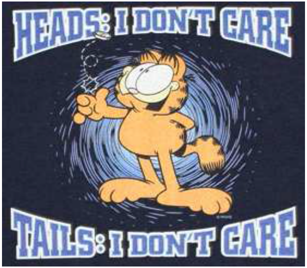



Figure: ᴚ$x_{[head \mapsto 0.5, tail \mapsto 0.5]}$

# SSMT: Quantification

$\exists x : \varphi$ I.e., for some value $\varphi$ holds.
$\text{Я}x : \varphi$ I.e., for random values $\varphi$ holds.

Randomized quantification to describe probabilistic events:

Figure: $\text{Я}x_{[2 \mapsto 0.5, 1 \mapsto 0.5, \ldots]} \cdots$

CARL VON OSSIETZKY universität OLDENBURG

# Stochastic Satisfiability Modulo Theories: Symantic

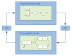
The semantics of an SSMT formula $\Phi$ is given by its maximum probability of satisfaction $Pr(\Phi)$ defined as follows:

$$Pr(\varepsilon : \varphi) = \begin{cases} 0 \text{ } if \text{ } \varphi \text{ } is \text{ } unsatisfiable, \\ 1 \text{ } if \text{ } \varphi \text{ } is \text{ } satisfiable, \end{cases}$$

$$Pr(\exists x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = max_{v \in D_x} Pr(\mathcal{Q} : \varphi[v/x]),$$

$$Pr(\text{Я}^{d_x} x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = \sum_{v \in \mathcal{D}_x} d_x(v) \cdot Pr(\mathcal{Q} : \varphi[v/x])$$

The quantifier-free SMT formula $\Phi$ is called the matrix of $\Phi$



# Stochastic Satisfiability Modulo Theories: Symantic

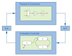
The semantics of an SSMT formula $\Phi$ is given by its maximum probability of satisfaction $Pr(\Phi)$ defined as follows:

$$Pr(\varepsilon : \varphi) = \begin{cases} 0 \ \textit{if } \varphi \textit{ is unsatisfiable,} \\ 1 \ \textit{if } \varphi \textit{ is satisfiable,} \end{cases}$$

$$Pr(\exists x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = max_{v \in D_x} Pr(\mathcal{Q} : \varphi[v/x]),$$

$$Pr(\text{Я}^{d_x} x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = \sum_{v \in \mathcal{D}_x} d_x(v) \cdot Pr(\mathcal{Q} : \varphi[v/x])$$

The quantifier-free SMT formula $\Phi$ is called the matrix of $\Phi$



# Stochastic Satisfiability Modulo Theories: Symantic

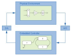
The semantics of an SSMT formula $\Phi$ is given by its maximum probability of satisfaction $Pr(\Phi)$ defined as follows:

$$Pr(\varepsilon : \varphi) = \begin{cases} 0 \text{ if } \varphi \text{ is unsatisfiable,} \\ 1 \text{ if } \varphi \text{ is satisfiable,} \end{cases}$$

$$Pr(\exists x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = max_{v \in D_x} Pr(\mathcal{Q} : \varphi[v/x]),$$

$$Pr(\text{Я}^{d_x} x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = \sum_{v \in \mathcal{D}_x} d_x(v) \cdot Pr(\mathcal{Q} : \varphi[v/x])$$

The quantifier-free SMT formula $\Phi$ is called the matrix of $\Phi$



# Stochastic Satisfiability Modulo Theories: Symantic

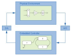
The semantics of an SSMT formula $\Phi$ is given by its maximum probability of satisfaction $Pr(\Phi)$ defined as follows:

$$Pr(\varepsilon : \varphi) = \begin{cases} 0 \textit{ if } \varphi \textit{ is unsatisfiable,} \\ 1 \textit{ if } \varphi \textit{ is satisfiable,} \end{cases}$$

$$Pr(\exists x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = max_{v \in D_x} Pr(\mathcal{Q} : \varphi[v/x]),$$

$$Pr(\text{Я}^{d_x} x \in \mathcal{D}_x \odot \mathcal{Q} : \varphi) = \sum_{v \in \mathcal{D}_x} d_x(v) \cdot Pr(\mathcal{Q} : \varphi[v/x])$$

The quantifier-free SMT formula $\Phi$ is called the matrix of $\Phi$



# Example:

$$\Phi = \exists x \in \{2,3,4\}, \text{Я}_{[1 \mapsto 0.2, 2 \mapsto 0.4, 3 \mapsto 0.4]} y \in \{1,2,3\} : (x + y > 3 \vee 2 \cdot y - x > 3) \wedge (x < 4)$$

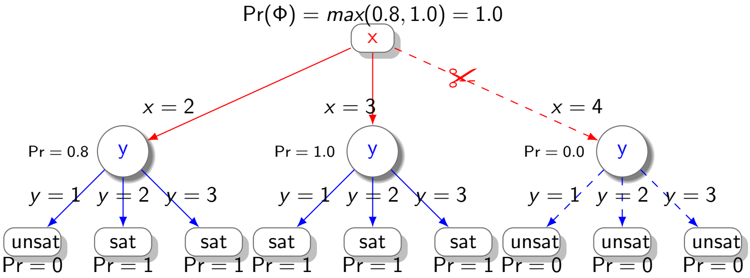


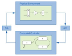

Figure: An example of SSMT formula, the selected part will be traversed and the other part will be pruned from the search space.

# SSMT resolution Calculus

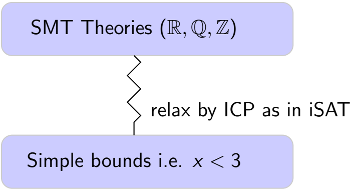
# Resolution Calculus for SAT and SMT

**(Sound and Complete SAT resolution calculus)**

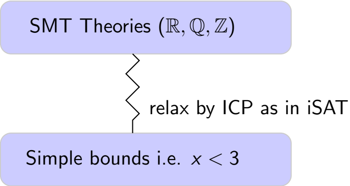
$$\frac{((C_1 \vee x) \wedge (C_2 \vee \neg x))}{(C_1 \vee C_2)} x, \neg x \notin (C_1 \vee C_2) \qquad \text{(SAT-Resolution [Rob65])}$$



# Resolution Calculus for SAT and SMT



# Resolution Calculus for SAT and SMT




## (Sound and Complete SMT resolution calculus)

$$\frac{(\mathcal{Q} : (C_1 \vee x \sim a) \wedge (C_2 \vee x \sim' b))}{(C_1 \vee C_2)} \mathcal{Q}_x : (x \sim a) \wedge (x \sim' b) \vdash \mathit{false}$$

(SMT-Resolution)

*where* $\sim, \sim' \in \{\leq, <, \geq, >\}$.

# Resolution of SSMT

## (Base case)

$$\frac{(c \in \varphi)}{c^0} \qquad \text{(RR.1)}$$

## Example 1

$$\frac{\exists x \in \{1,5,6\}, \text{Я}_{[4 \mapsto 0.3, 17 \mapsto 0.7]} y : (x \leq 3 \vee y > 10 \vee z > 12) \wedge (x > 5)}{(x \leq 3 \vee y > 10 \vee z > 12)^0, (x > 5)^0} \qquad (1)$$

# Resolution of SSMT

## (Falsification rule)

$$\frac{\begin{pmatrix} c \subseteq \{x \sim a \mid x \in Var(c)\}, \not\models c, \mathcal{Q}(c) = \mathcal{Q}_1 x_1 \ldots \mathcal{Q}_i x_i, \\ \textit{for each } \tau : Var(\varphi) \downarrow_i \rightarrow \mathbb{SB} \textit{ with } \forall x \in Var(\varphi) : \tau(x) \textit{ in } ff_c(x \sim a) : \\ \models \varphi[\tau(x_1)/x_1] \ldots [\tau(x_i)/x_i] \end{pmatrix}}{c^1} \tag{RR.2}$$

## Example 1

$$\frac{\exists x \in \{1, 5, 6\}, \text{Я}_{[4 \mapsto 0.3, 17 \mapsto 0.7]} y : (x \leq 3 \vee y > 10 \vee z > 12) \wedge (x > 5)}{(y \leq 10)^1 \wedge (z \leq 12)^1} \tag{1}$$

# Resolution of SSMT

## (Resolution in case of free variable)

$$\frac{\begin{pmatrix} (x \sim a \vee c_1)^{p_1}, (x \sim' b \vee c_2)^{p_2}, \mathcal{Q}_x \notin \mathcal{Q}, \\ (\exists x : x \sim a \wedge x \sim' b) \vdash False, \not\models (c_1 \vee c_2) \\ p = max(p_1, p_2) \end{pmatrix}}{(c_1 \vee c_2)^p} \quad \text{(RR.3e)}$$

## Example 1

$$\frac{\exists x \in \{1, 5, 6\}, \text{Ⅎ}_{[4 \mapsto 0.3, 17 \mapsto 0.7]} y : (x \leq 3 \vee y > 10 \vee z > 12)^0 \wedge (z \leq 12)^1}{(y > 10 \vee x \leq 3)^1} \quad (1)$$

# Resolution of SSMT

## (Resolution rule between clauses)

$$\frac{\left(\begin{array}{c}(x \sim a \vee c_1)^{p_1}, (x \sim' b \vee c_2)^{p_2}, (\mathcal{Q}_x : x \sim a \wedge x \sim' b \vdash False) \\ \mathcal{Q}_x \in \mathcal{Q}, \not\models (c_1 \vee c_2) \\ p = \begin{cases} max(p_1, p_2) & \text{if } \mathcal{Q} = \exists \\ p_1 \cdot Pr(x \sim' b) + p_2 \cdot Pr(x \sim a) & \text{if } \mathcal{Q} = \reflectbox{R}^{p_x} \end{cases}\end{array}\right)}{(c_1 \vee c_2)^p} \quad \text{(RR.3)}$$

## Example 1

$$\frac{\exists x \in \{1, 5, 6\}, \reflectbox{R}_{[4 \mapsto 0.3, 17 \mapsto 0.7]} y : (y > 10 \vee x \leq 3)^1 \wedge (y \leq 10)^1}{(x \leq 3)^1} \quad (1)$$

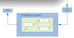
# Resolution of SSMT

## (Resolution rule between clauses)

$$\frac{\begin{pmatrix} (x \sim a \vee c_1)^{p_1}, (x \sim' b \vee c_2)^{p_2}, (\mathcal{Q}_x : x \sim a \wedge x \sim' b \vdash False) \\ \mathcal{Q}_x \in \mathcal{Q}, \not\models (c_1 \vee c_2) \\ p = \begin{cases} max(p_1, p_2) & if\ \mathcal{Q} = \exists \\ p_1 \cdot Pr(x \sim' b) + p_2 \cdot Pr(x \sim a) & if\ \mathcal{Q} = \text{Я}^{p_x} \end{cases} \end{pmatrix}}{(c_1 \vee c_2)^p} \quad \text{(RR.3)}$$

## Example 1

$$\frac{\exists x \in \{1, 5, 6\}, \text{Я}_{[4 \mapsto 0.3, 17 \mapsto 0.7]} y : (x \leq 3)^1 \wedge (x > 5)^0}{\emptyset^1} \quad (1)$$



# Generalized Craig Interpolants *for SSMT*

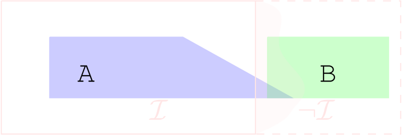
Idea: The same as GCI for SSAT [TF12]; namely:




- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).

# Generalized Craig Interpolants *for SSMT*

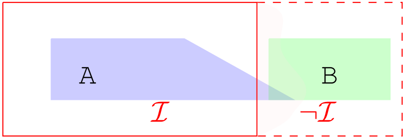
Idea: The same as GCI for SSAT [TF12]; namely:




- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).

# Generalized Craig Interpolants *for SSMT*

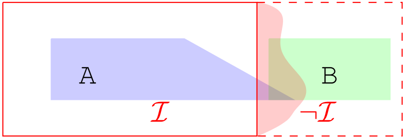
Idea: The same as GCI for SSAT [TF12]; namely:




- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).

# Generalized Craig Interpolants *for SSMT*

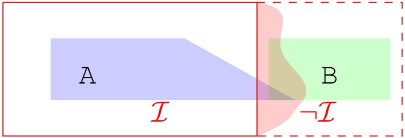
Idea: The same as GCI for SSAT [TF12]; namely:




- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).

# Generalized Craig Interpolants *for SSMT*

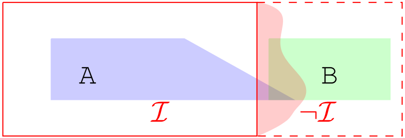
Idea: The same as GCI for SSAT [TF12]; namely:



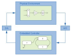

- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).

# Generalized Craig Interpolants *for SSMT*

Idea: The same as GCI for SSAT [TF12]; namely:

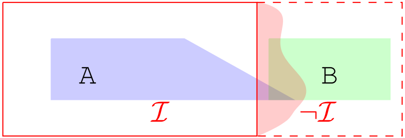



- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).



# Generalized Craig Interpolants *for SSMT*

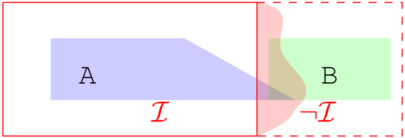
Idea: The same as GCI for SSAT [TF12]; namely:




- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).

# Generalized Craig Interpolants *for SSMT*

Idea: The same as GCI for SSAT [TF12]; namely:




- we apply the same procedure as in SSAT i.e. adding $\neg S_{A,B}$
- we combine the previous procedure with iSAT reasoning technique i.e. simple bounds.
- we can use either Pudlák or McMillan mechanisms.
- this interpolant is the generalized one (SAT, SMT, and SSAT).

# Generalized Craig interpolation: Idea

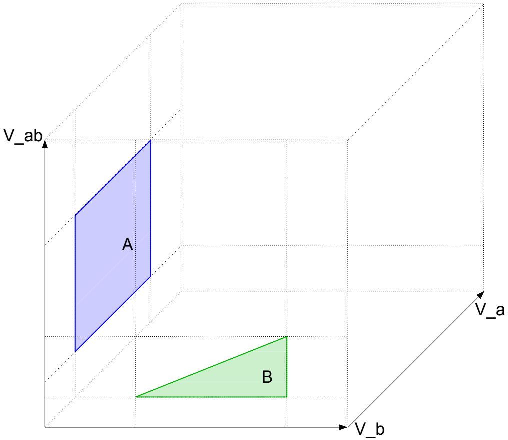



Figure: Generalized Craig interpolation [TF12]

# Generalized Craig interpolation: Idea

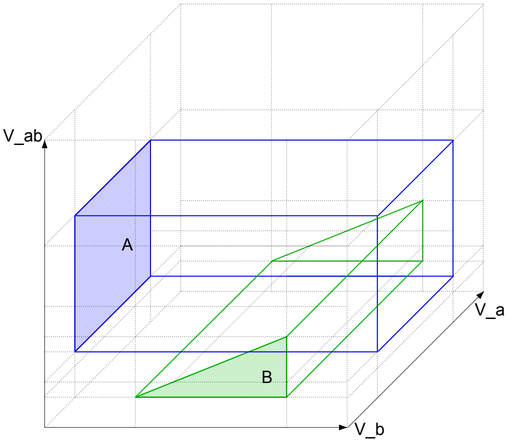



Figure: Generalized Craig interpolation [TF12]

# Generalized Craig interpolation: Idea

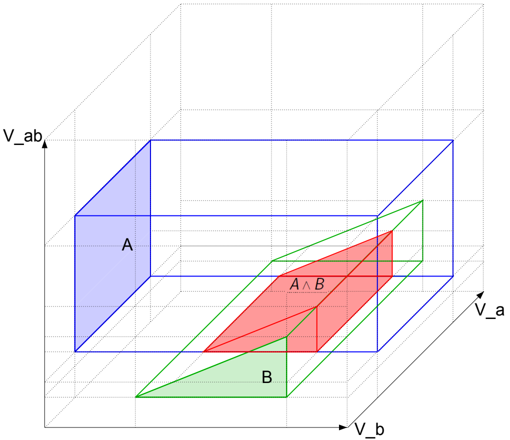



Figure: Generalized Craig interpolation [TF12]

# Generalized Craig interpolation: Idea

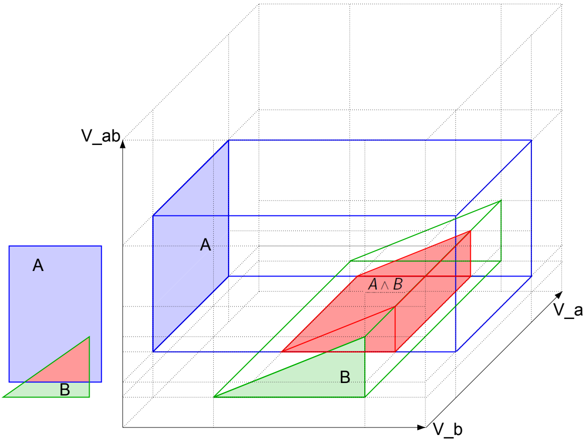



Figure: Generalized Craig interpolation [TF12]

# Generalized Craig interpolation: Idea

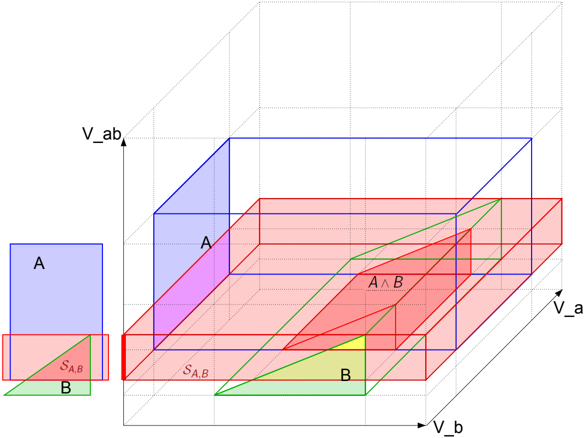



Figure: Generalized Craig interpolation [TF12]

# Generalized Craig interpolation: Idea

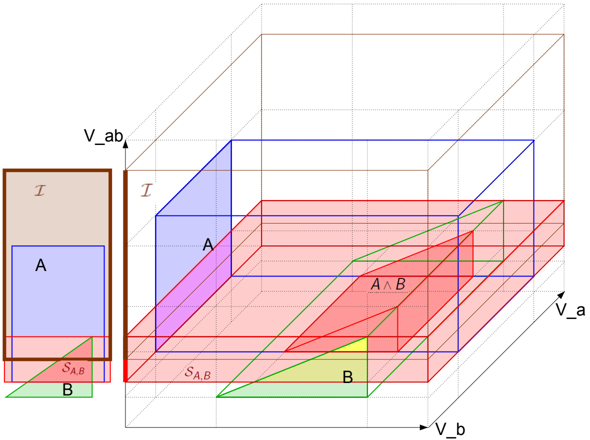



Figure: Generalized Craig interpolation [TF12]

## Definition 1 (Generalized Craig Interpolation–Pudlák extension)

- *Let $A$ and $B$ be some SMT formulae where $V_A := Var(A) \setminus Var(B) = \{a_1, \ldots, a_\alpha\}$, $V_B := Var(B) \setminus Var(A) = \{b_1, \ldots, b_\beta\}$, $V_{A,B} := Var(A) \cap Var(B)$,*
- $A^\exists = \exists a_1, \ldots, a_\alpha : A$, *and*
- $\overline{B}^\forall = \neg \exists b_1, \ldots, b_\beta : B$.

*An SMT formula $\mathcal{I}$ is called a generalized Craig interpolant for $(A, B)$ if and only if the following properties are satisfied:*

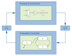
- $Var(\mathcal{I}) \subseteq V_{A,B}$,
- $\models_{\mathcal{L}} (A^\exists \wedge \overline{B}^\forall) \rightarrow \mathcal{I}$,
- $\models_{\mathcal{L}} \mathcal{I} \rightarrow (A^\exists \vee \overline{B}^\forall)$

## Definition cont.

*GCI is computed according to the following rules:*

$$\frac{c \vdash_{R.1} c^p, \quad \mathcal{I} = \begin{cases} False, & c \in A \\ True, & c \in B \end{cases}.}{(c^p, \mathcal{I})} \tag{GR.1}$$

$$\frac{\vdash_{R.2} c^p \quad \mathcal{I} \text{ is any formula over } V_{A,B}}{(c^p, \mathcal{I})} \tag{GR.2}$$



# Definition cont.

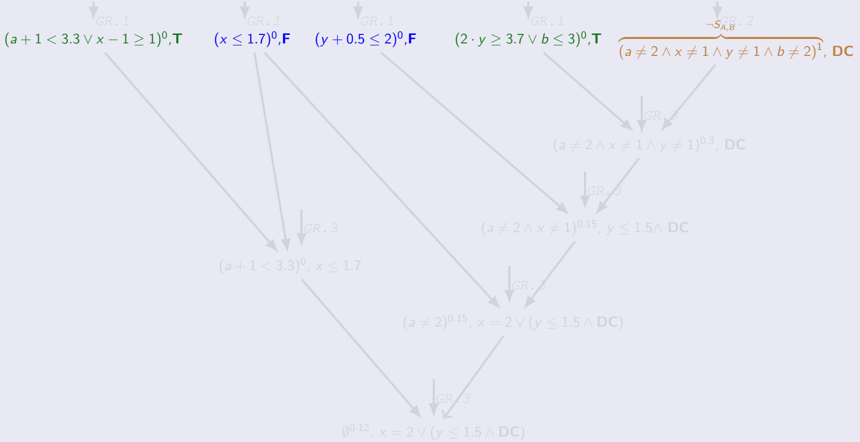
$$
\frac{((x \sim a \vee c_1)^{p_1}, \mathcal{I}_1), ((x \sim b \vee c_2)^{p_2}, \mathcal{I}_2), (x \sim a \wedge x \sim b \vdash false) \quad (x \sim a \vee c_1)^{p_1}, (x \sim b \vee c_2)^{p_2} \vdash_{R.3} (c_1 \vee c_2)^{p}, \quad \mathcal{I} = \begin{cases} \mathcal{I}_1 \vee \mathcal{I}_2 & \text{if } x \in V_A \\ \mathcal{I}_1 \wedge \mathcal{I}_2 & \text{if } x \in V_B \\ (x \sim a \vee \mathcal{I}_1) \wedge (x \sim b \vee \mathcal{I}_2) & \text{if } x \in V_{A,B} \end{cases}}{((c_1 \vee c_2)^{p}, \mathcal{I})} \quad (GR.3)
$$

$$
p = \begin{cases} max(p_1, p_2) & \text{if } \mathcal{Q} = \exists \\ p_1 \cdot Pr(x \sim b) + p_2 \cdot Pr(x \sim a) & \text{if } \mathcal{Q} = \text{Я}^{p_*} \end{cases} \quad (GR.4)
$$

# Example 2

$$\text{Я}.a \in \{2 \mapsto 0.8, 3 \mapsto 0.2\}, \exists.x \in \{1,2\}, \text{Я}.y \in \{1 \mapsto 0.5, 2 \mapsto 0.5\}, \text{Я}.b \in \{2 \mapsto 0.3, 3.2 \mapsto 0.7\}:$$

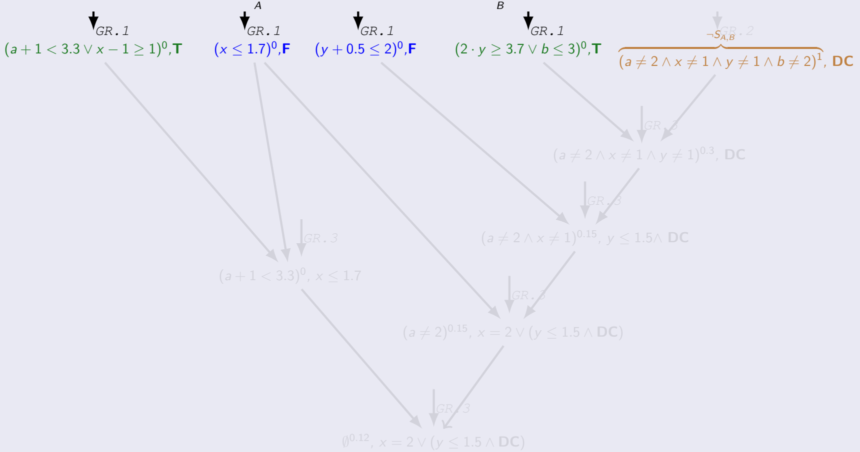
$$\underbrace{(y + 0.5 \leq 2) \wedge (x \leq 1.7)}_{A} \wedge \underbrace{(a + 1 < 3.3 \vee x - 1 \geq 1) \wedge (2 \cdot y \geq 3.7 \vee b \leq 3)}_{B}$$



# Example 2

$$ᴙ.a \in \{2 \mapsto 0.8, 3 \mapsto 0.2\}, \exists.x \in \{1,2\}, ᴙ.y \in \{1 \mapsto 0.5, 2 \mapsto 0.5\}, ᴙ.b \in \{2 \mapsto 0.3, 3.2 \mapsto 0.7\} :$$

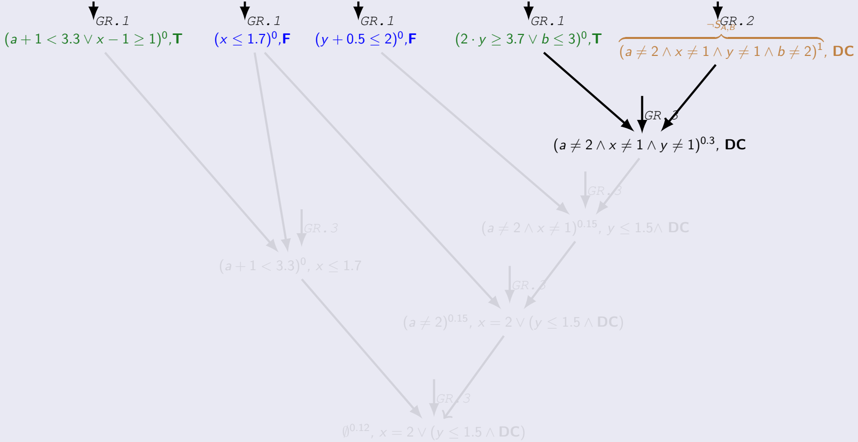
$$\underbrace{(y + 0.5 \leq 2) \wedge (x \leq 1.7)}_{A} \wedge \underbrace{(a + 1 < 3.3 \vee x - 1 \geq 1) \wedge (2 \cdot y \geq 3.7 \vee b \leq 3)}_{B}$$



# Example 2

$$\text{Ɐ}.a \in \{2 \mapsto 0.8, 3 \mapsto 0.2\}, \exists.x \in \{1,2\}, \text{Ɐ}.y \in \{1 \mapsto 0.5, 2 \mapsto 0.5\}, \text{Ɐ}.b \in \{2 \mapsto 0.3, 3.2 \mapsto 0.7\}:$$

$$\underbrace{(y + 0.5 \leq 2) \wedge (x \leq 1.7)}_{A} \wedge \underbrace{(a + 1 < 3.3 \vee x - 1 \geq 1) \wedge (2 \cdot y \geq 3.7 \vee b \leq 3)}_{B}$$

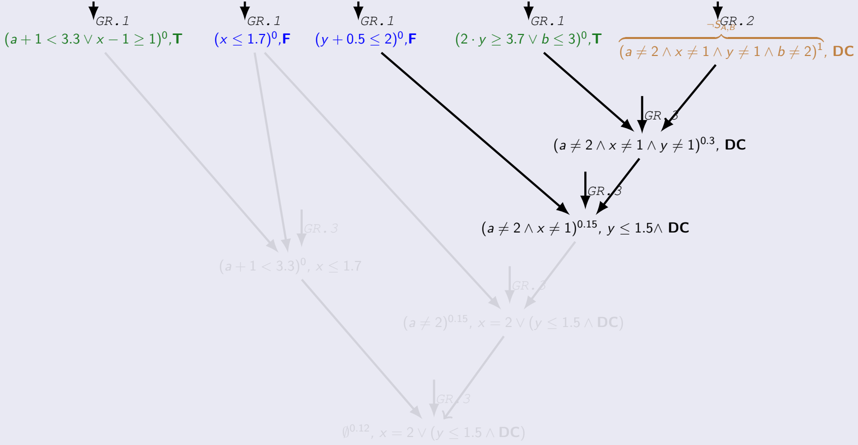


CARL VON OSSIETZKY universität OLDENBURG

# Example 2

$$\text{Я}.a \in \{2 \mapsto 0.8, 3 \mapsto 0.2\}, \exists.x \in \{1,2\}, \text{Я}.y \in \{1 \mapsto 0.5, 2 \mapsto 0.5\}, \text{Я}.b \in \{2 \mapsto 0.3, 3.2 \mapsto 0.7\}:$$

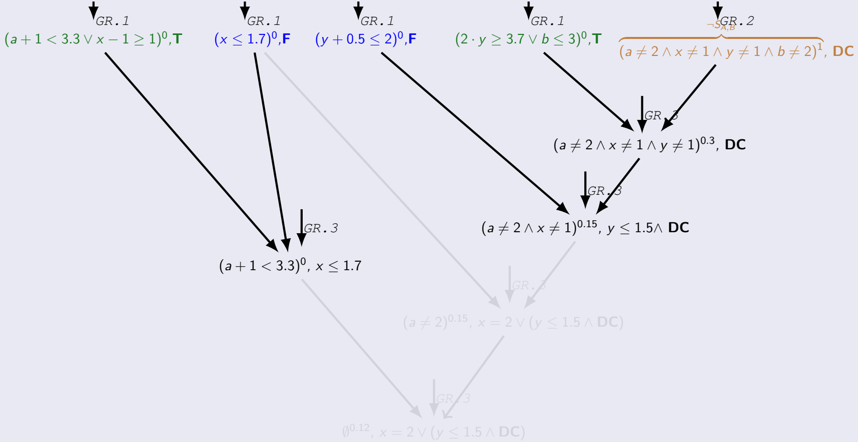
$$\underbrace{(y+0.5 \le 2) \wedge (x \le 1.7)}_{A} \wedge \underbrace{(a+1 < 3.3 \vee x-1 \ge 1) \wedge (2 \cdot y \ge 3.7 \vee b \le 3)}_{B}$$



# Example 2

$$\text{Я}.a \in \{2 \mapsto 0.8, 3 \mapsto 0.2\}, \exists.x \in \{1,2\}, \text{Я}.y \in \{1 \mapsto 0.5, 2 \mapsto 0.5\}, \text{Я}.b \in \{2 \mapsto 0.3, 3.2 \mapsto 0.7\}:$$

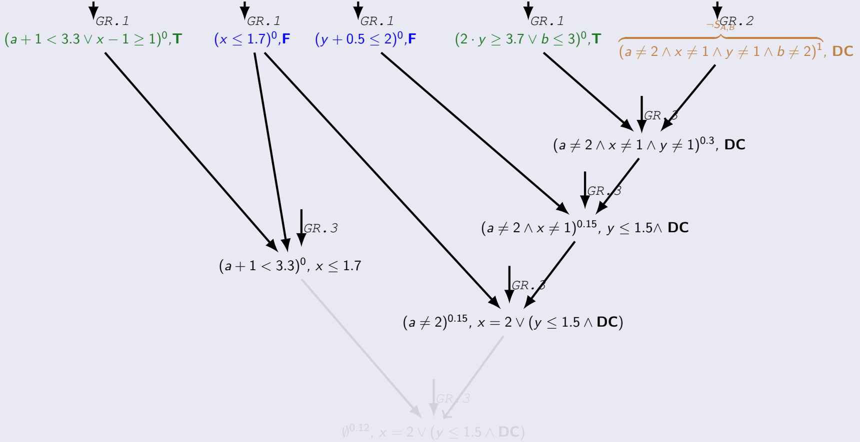
$$\underbrace{(y + 0.5 \leq 2) \wedge (x \leq 1.7)}_{A} \wedge \underbrace{(a + 1 < 3.3 \vee x - 1 \geq 1) \wedge (2 \cdot y \geq 3.7 \vee b \leq 3)}_{B}$$



# Example 2

$$\text{Я}.a \in \{2 \mapsto 0.8, 3 \mapsto 0.2\}, \exists.x \in \{1,2\}, \text{Я}.y \in \{1 \mapsto 0.5, 2 \mapsto 0.5\}, \text{Я}.b \in \{2 \mapsto 0.3, 3.2 \mapsto 0.7\} :$$

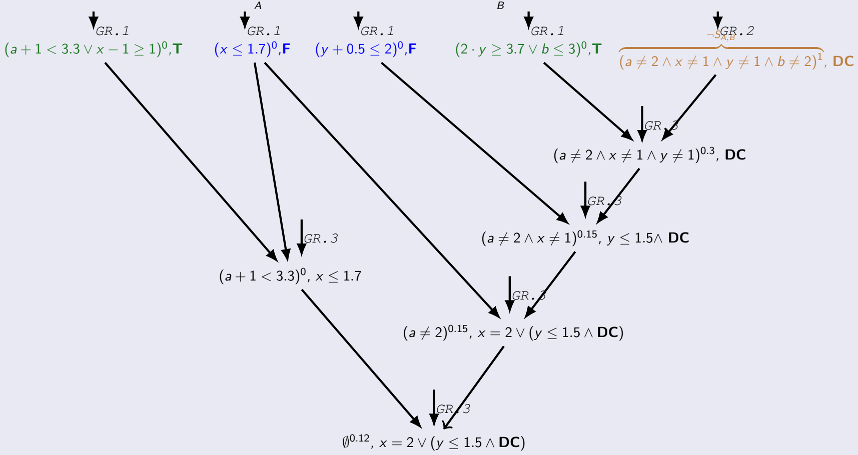
$$\underbrace{(y + 0.5 \leq 2) \wedge (x \leq 1.7)}_{A} \wedge \underbrace{(a + 1 < 3.3 \vee x - 1 \geq 1) \wedge (2 \cdot y \geq 3.7 \vee b \leq 3)}_{B}$$



## Example 2

$$\text{Ʉ}.a \in \{2 \mapsto 0.8, 3 \mapsto 0.2\}, \exists.x \in \{1,2\}, \text{Ʉ}.y \in \{1 \mapsto 0.5, 2 \mapsto 0.5\}, \text{Ʉ}.b \in \{2 \mapsto 0.3, 3.2 \mapsto 0.7\} :$$

$$\underbrace{(y + 0.5 \leq 2) \wedge (x \leq 1.7)}_{A} \wedge \underbrace{(a + 1 < 3.3 \vee x - 1 \geq 1) \wedge (2 \cdot y \geq 3.7 \vee b \leq 3)}_{B}$$



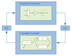
# Conclusion and Future work

# Conclusion

- An approach to compute Craig Interpolant for SSMT problems
- CI is computed regardless of the linearity of a formula.
- All SAT, SSAT, SMT (linear, non-linear, integer and rational) problems are also covered by this approach.
- iSAT interpolants are not simple ones, due to non-linear constraints and ICP ☹.

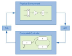
# Conclusion

- An approach to compute Craig Interpolant for SSMT problems
- CI is computed regardless of the linearity of a formula.
- All SAT, SSAT, SMT (linear, non-linear, integer and rational) problems are also covered by this approach.
- iSAT interpolants are not simple ones, due to non-linear constraints and ICP ☹.



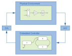
# Conclusion

- An approach to compute Craig Interpolant for SSMT problems
- CI is computed regardless of the linearity of a formula.
- All SAT, SSAT, SMT (linear, non-linear, integer and rational) problems are also covered by this approach.
- iSAT interpolants are not simple ones, due to non-linear constraints and ICP ☹.



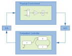
# Conclusion

- An approach to compute Craig Interpolant for SSMT problems
- CI is computed regardless of the linearity of a formula.
- All SAT, SSAT, SMT (linear, non-linear, integer and rational) problems are also covered by this approach.
- iSAT interpolants are not simple ones, due to non-linear constraints and ICP ☹.



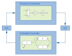
# Conclusion

- An approach to compute Craig Interpolant for SSMT problems
- CI is computed regardless of the linearity of a formula.
- All SAT, SSAT, SMT (linear, non-linear, integer and rational) problems are also covered by this approach.
- iSAT interpolants are not simple ones, due to non-linear constraints and ICP ☹.



# Future Work

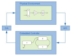
- proper approach to compute $S_{A,B}$ ☹☹.
- slackness of interpolants ☹☺.
- integrate GCI with stochastic CEGAR loop.



# References I

Aws Albarghouthi and Kenneth L. McMillan.
Beautiful interpolants.
In Natasha Sharygina and Helmut Veith, editors, *CAV*, volume 8044 of *Lecture Notes in Computer Science*, pages 313–329. Springer, 2013.

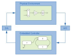
Fahiem Bacchus, Shannon Dalmao, and Toniann Pitassi.
Dpll with caching: A new algorithm for #sat and bayesian inference.
*Electronic Colloquium on Computational Complexity (ECCC)*, 10(003), 2003.



# References II

Régis Blanc, Ashutosh Gupta, Laura Kovács, and Bernhard Kragl.
Tree interpolation in vampire.
In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *LPAR*, volume 8312 of *Lecture Notes in Computer Science*, pages 173–181. Springer, 2013.

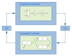
William Craig.
Three uses of the herbrand-gentzen theorem in relating model theory and proof theory.
*J. Symb. Log.*, 22(3):269–285, 1957.



# References III

Martin Fränzle, Ernst Moritz Hahn, Holger Hermanns, Nicolás Wolovick, and Lijun Zhang.
Measurability and safety verification for stochastic hybrid systems.
In Marco Caccamo, Emilio Frazzoli, and Radu Grosu, editors, *HSCC*, pages 43–52. ACM, 2011.

Martin Fraenzle, Holger Hermanns, and Tino Teige.
Stochastic satisfiability modulo theory: A novel technique for the analysis of probabilistic hybrid systems.
In *HSCC*, pages 172–186, 2008.

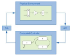


# References IV

Eric Freudenthal and Vijay Karamcheti.
Qtm: Trust management with quantified stochastic attributes.
Technical Report NYU Computer Science Technical Report TR2003-848, Courant Institute of Mathematical Sciences , New York University, 2003.

Kenneth L. McMillan.
Interpolation and sat-based model checking.
In *CAV*, pages 1–13, 2003.

Ahmed Mahdi and Martin Fraenzle.
Resolution for stochastic satisfiability modulo theories.
2014.



# References V


- Stephen M. Majercik and Michael L. Littman.
  Maxplan: A new approach to probabilistic planning.
  In *AIPS*, pages 86–93, 1998.

- Stephen M. Majercik and Michael L. Littman.
  Contingent planning under uncertainty via stochastic satisfiability.
  *Artif. Intell.*, 147(1-2):119–162, 2003.

- Christos H. Papadimitriou.
  *Computational complexity*.
  Addison-Wesley, 1994.

- John Alan Robinson.
  A machine-oriented logic based on the resolution principle.
  *J. ACM*, 12(1):23–41, 1965.

# References VI

Tino Teige and Martin Fränzle.
Resolution for stochastic boolean satisfiability.
In *LPAR (Yogyakarta)*, pages 625–639, 2010.

Tino Teige and Martin Fränzle.
Generalized craig interpolation.
*Logical Methods in Computer Science*, 8(2), 2012.

Toby Walsh.
Stochastic constraint programming.
In *ECAI*, pages 111–115, 2002.

Thank you for Listening!

Any questions!