RWTH AACHEN UNIVERSITY

On Variable Orderings in MCSAT for Non-linear Real Arithmetic

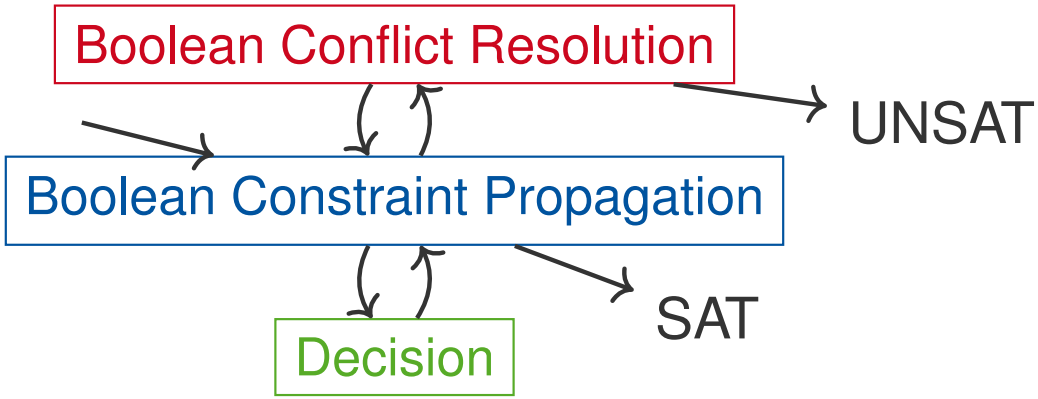
SC^2 2019

Jasper Nalbach Gereon Kremer Erika Ábrahám

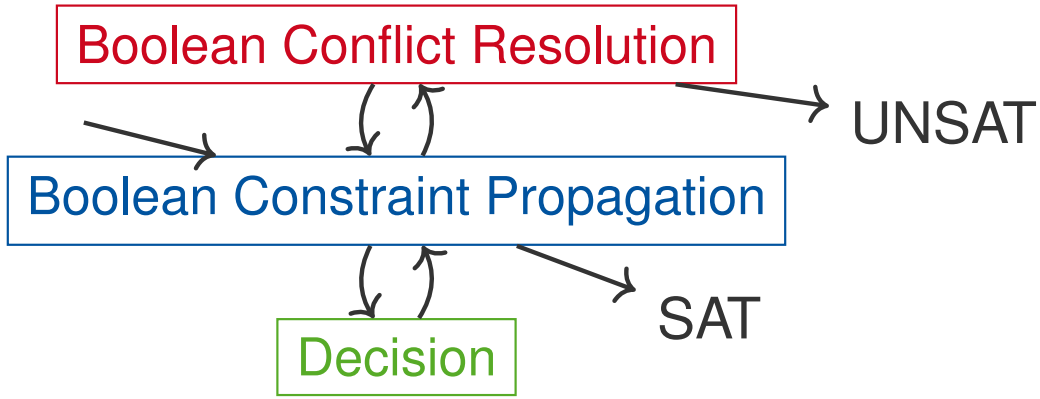
July 10, 2019

hybr2d Theory of Hybrid Systems Informatik 2 | RWTH AACHEN UNIVERSITY

Our MCSAT Approach



Our MCSAT Approach

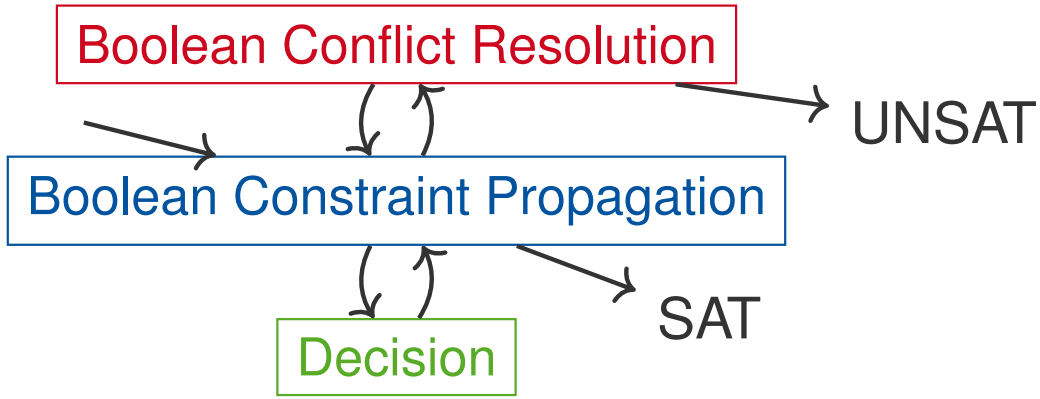


$$[\![]\!]$$

$$c_1 : a \vee b \vee d \qquad c_3 : \neg d \vee f$$

$$c_2 : \neg d \vee e \qquad c_4 : \neg e \vee \neg f$$

Our MCSAT Approach

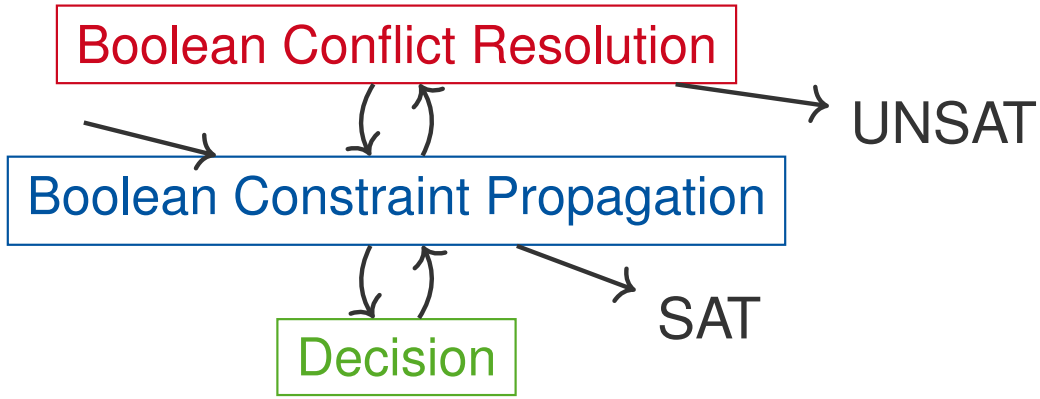


$$[\![\neg a]\!]$$

$$c_1 : a \vee b \vee d \qquad c_3 : \neg d \vee f$$

$$c_2 : \neg d \vee e \qquad c_4 : \neg e \vee \neg f$$

Our MCSAT Approach

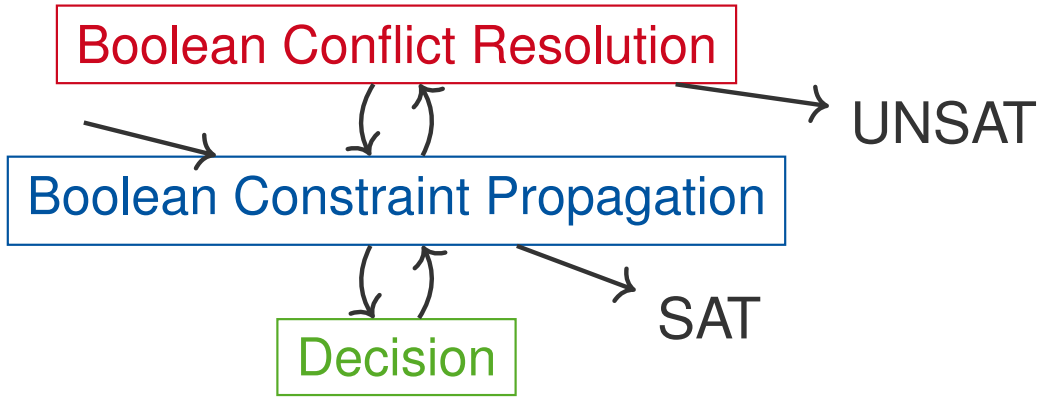


$$[\![\neg a, \neg b]\!]$$

$$c_1 : a \vee b \vee d \qquad c_3 : \neg d \vee f$$

$$c_2 : \neg d \vee e \qquad c_4 : \neg e \vee \neg f$$

Our MCSAT Approach

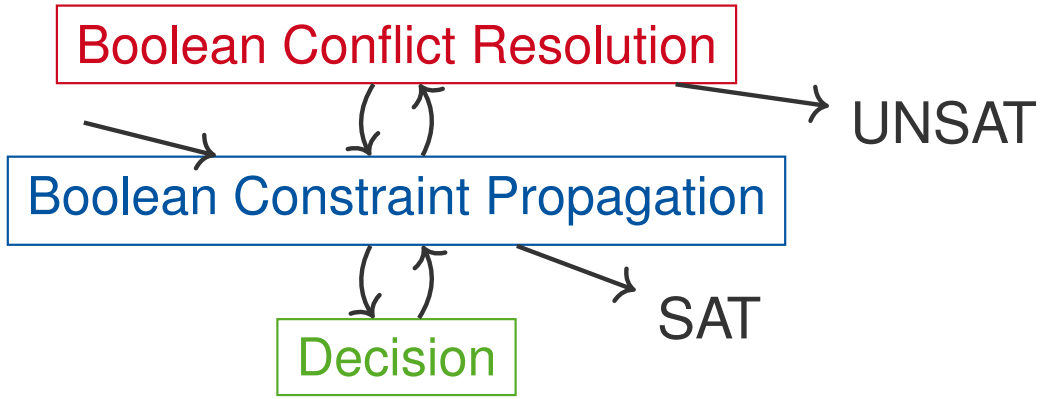


$$[\![\neg a, \neg b, c_1 \rightarrow d]\!]$$

$c_1 : a \vee b \vee d$ $\qquad$ $c_3 : \neg d \vee f$

$c_2 : \neg d \vee e$ $\qquad$ $c_4 : \neg e \vee \neg f$

Our MCSAT Approach

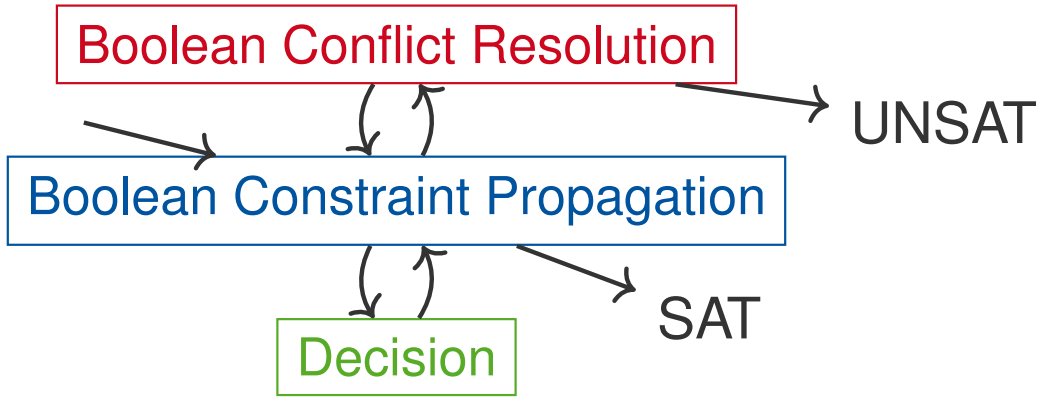


$$[\![\neg a, \neg b, c_1 \to d, c_2 \to e, c_3 \to f]\!]$$

$c_1 : a \vee b \vee d$ $\qquad$ $c_3 : \neg d \vee f$

$c_2 : \neg d \vee e$ $\qquad$ $c_4 : \neg e \vee \neg f$

Our MCSAT Approach

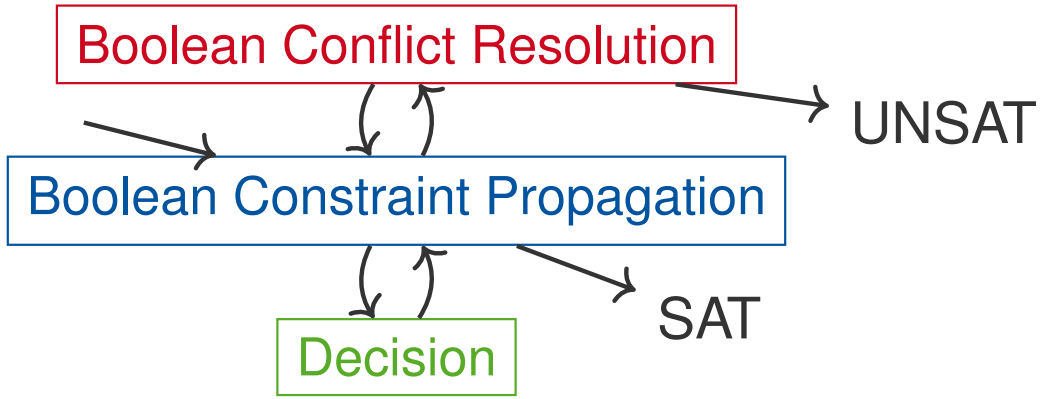


$$[\![]\!]$$

$$c_1 : x^2 y > 0 \vee a$$

$$c_2 : x > 0 \vee z = 0$$

Our MCSAT Approach

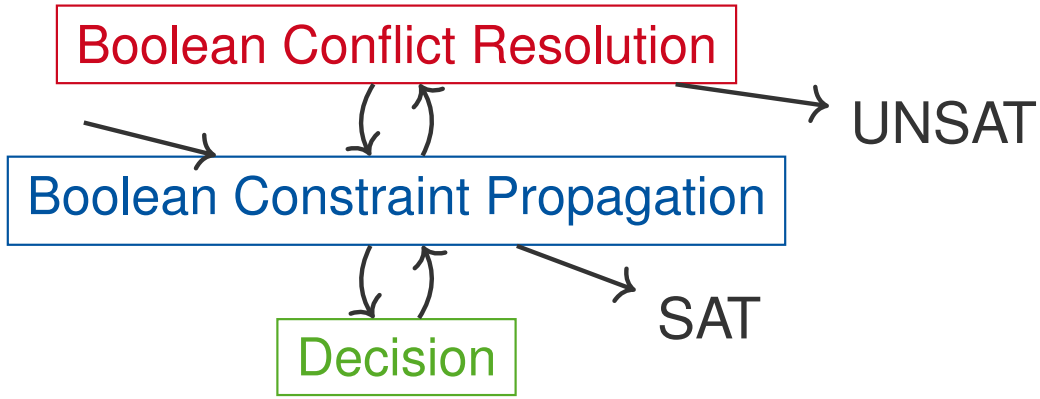


$$[\![x^2y > 0]\!]$$

$$c_1 : x^2y > 0 \vee a$$

$$c_2 : x > 0 \vee z = 0$$

Our MCSAT Approach

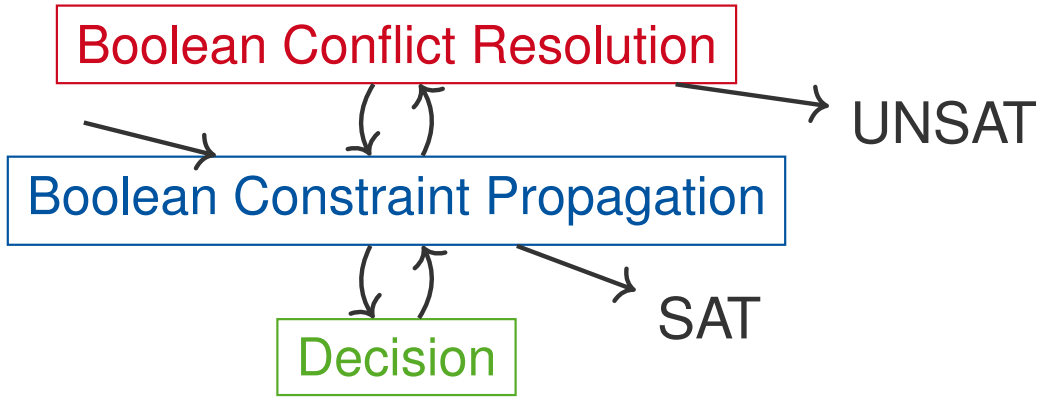


$$[\![x^2y > 0, x \mapsto -1]\!]$$

$$c_1 : x^2y > 0 \vee a$$

$$c_2 : x > 0 \vee z = 0$$

Our MCSAT Approach

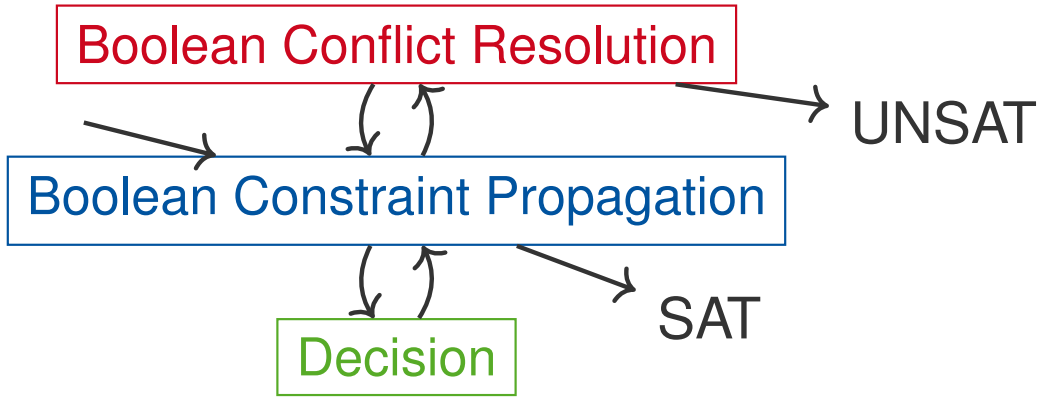


$$[\![x^2y > 0, x \mapsto -1]\!]$$

$$c_1 : x^2y > 0 \vee a$$

$$c_2 : x > 0 \vee z = 0$$

Our MCSAT Approach

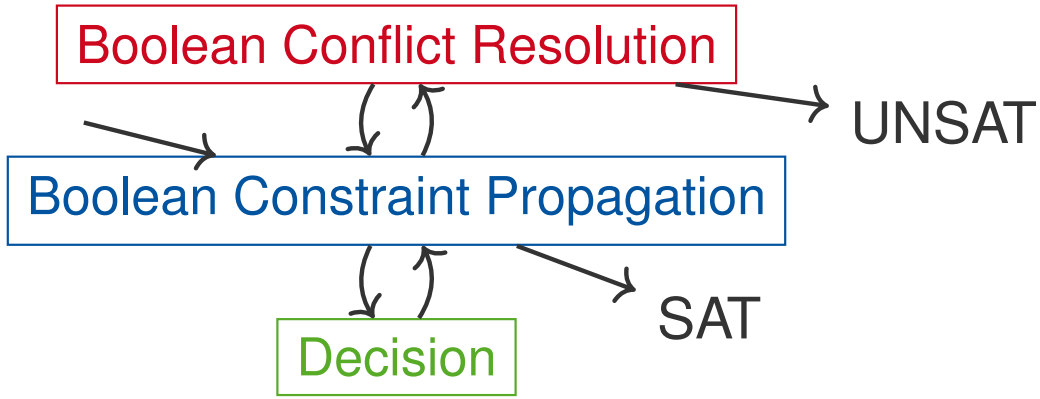


$$[\![x^2y > 0, x \mapsto -1, z = 0]\!]$$

$$c_1 : x^2y > 0 \vee a$$

$$c_2 : x > 0 \vee z = 0$$

Our MCSAT Approach

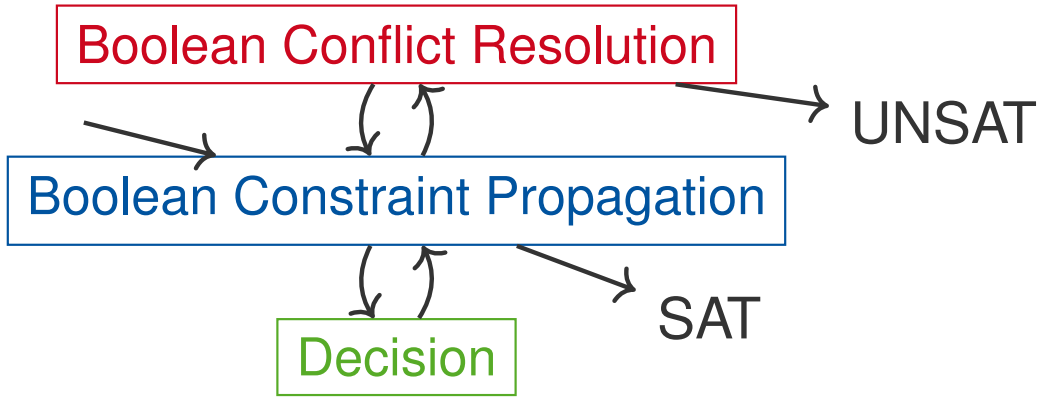


$$[\![x^2y > 0, x \mapsto -1, z = 0, y \mapsto (0, \infty)]\!]$$

$$c_1 : x^2y > 0 \vee a$$

$$c_2 : x > 0 \vee z = 0$$

Our MCSAT Approach

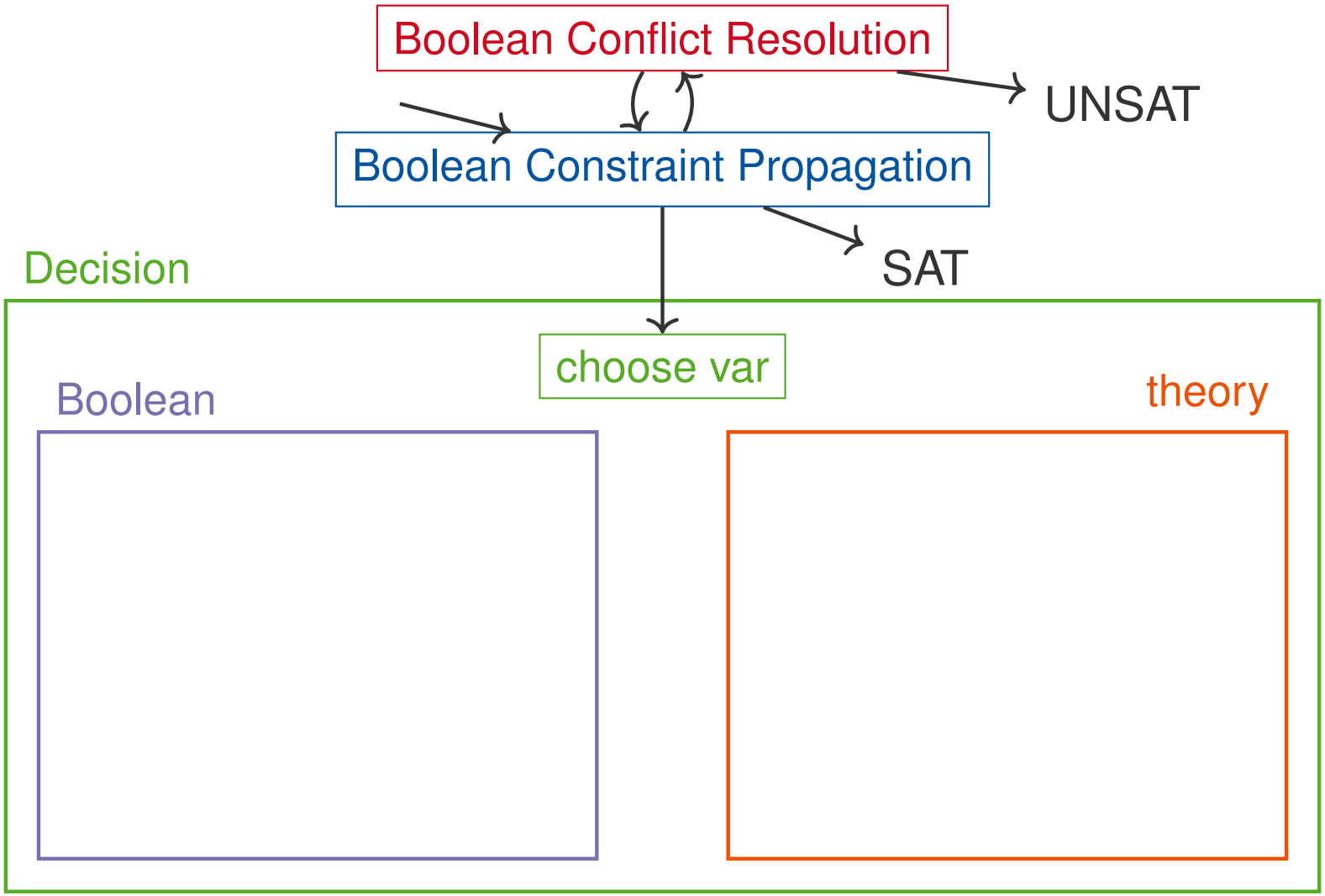


$$[\![x^2y > 0, x \mapsto -1, z = 0, y \mapsto 1]\!]$$

$$c_1 : x^2y > 0 \vee a$$

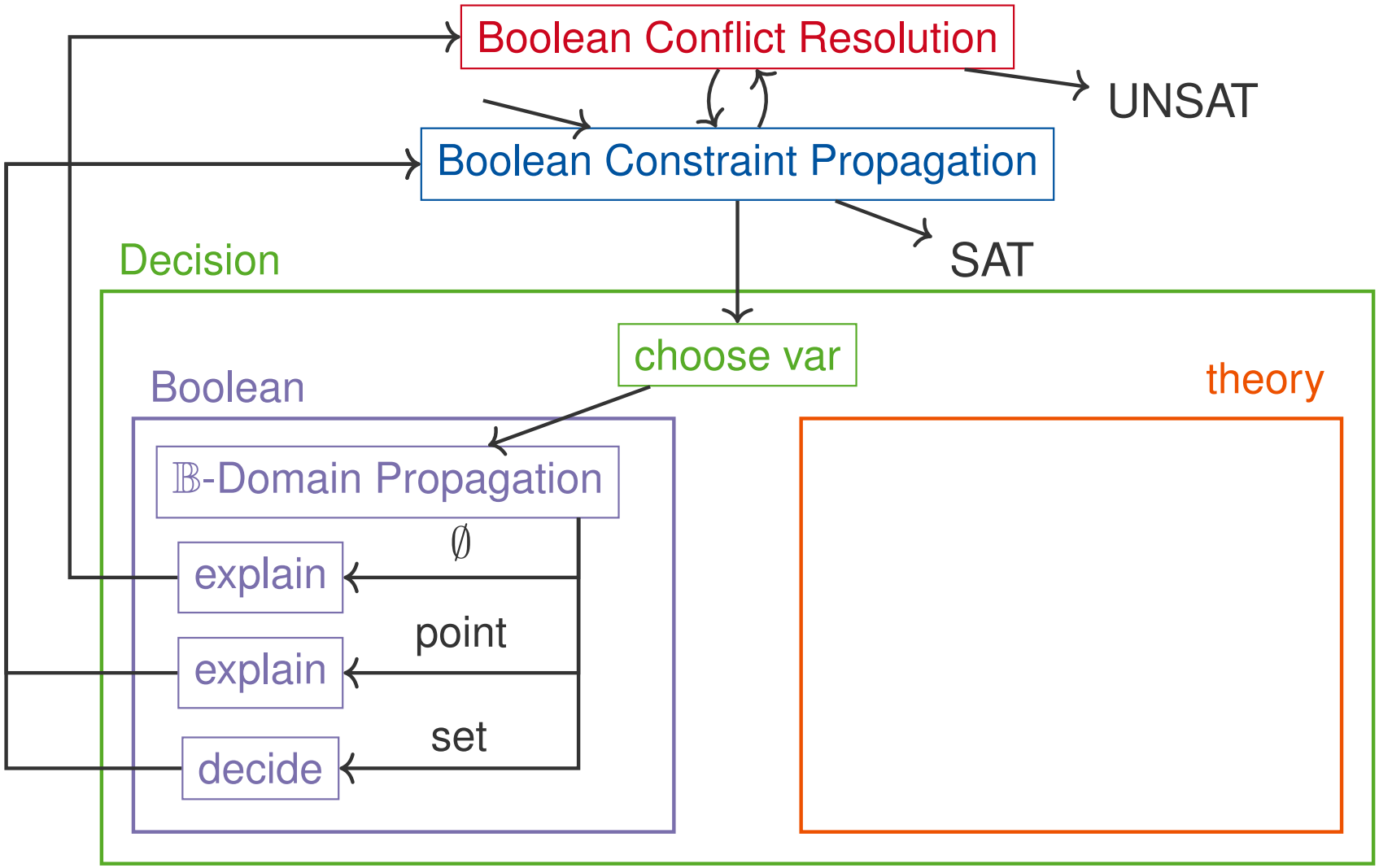
$$c_2 : x > 0 \vee z = 0$$

Our MCSAT Approach

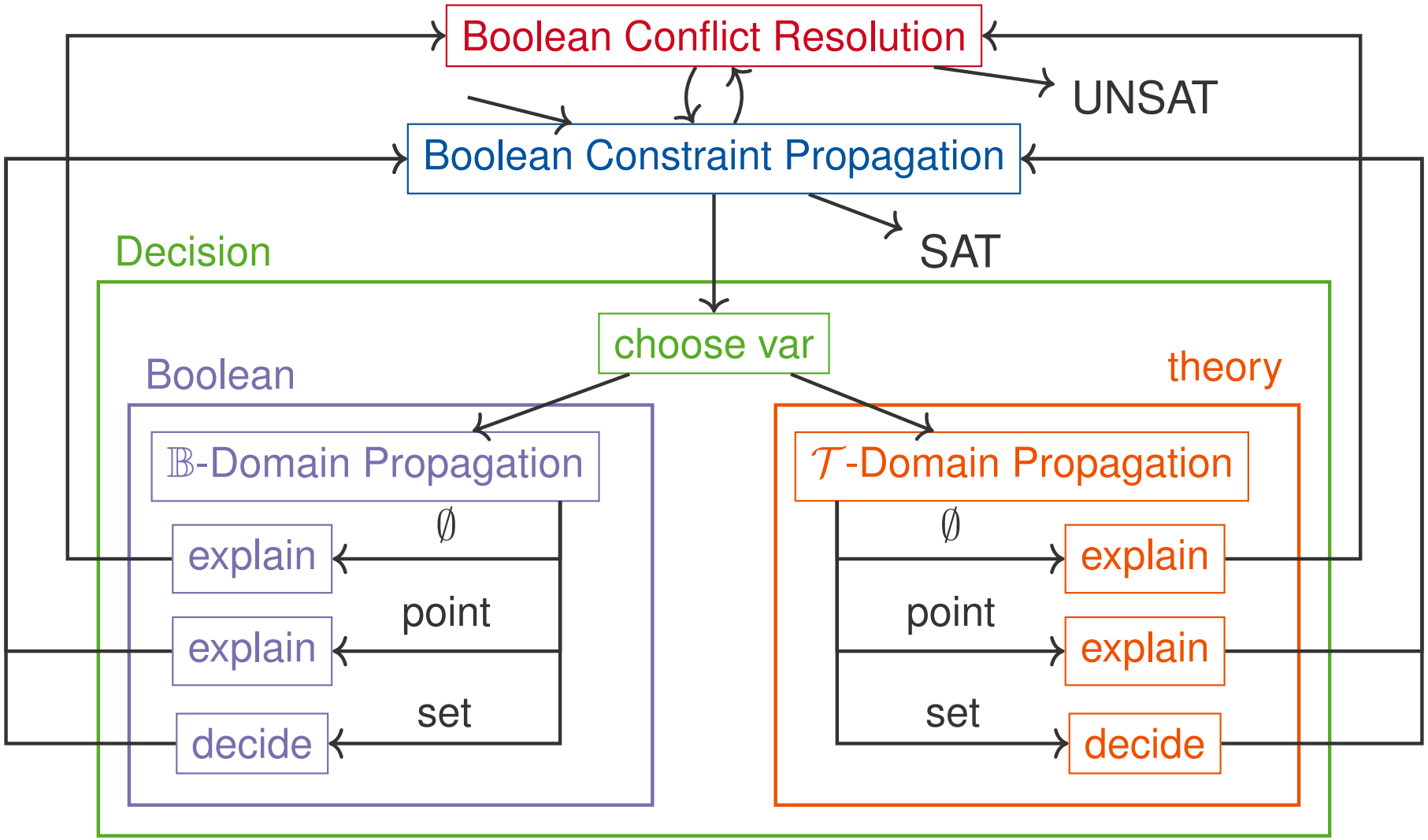


RWTH AACHEN UNIVERSITY

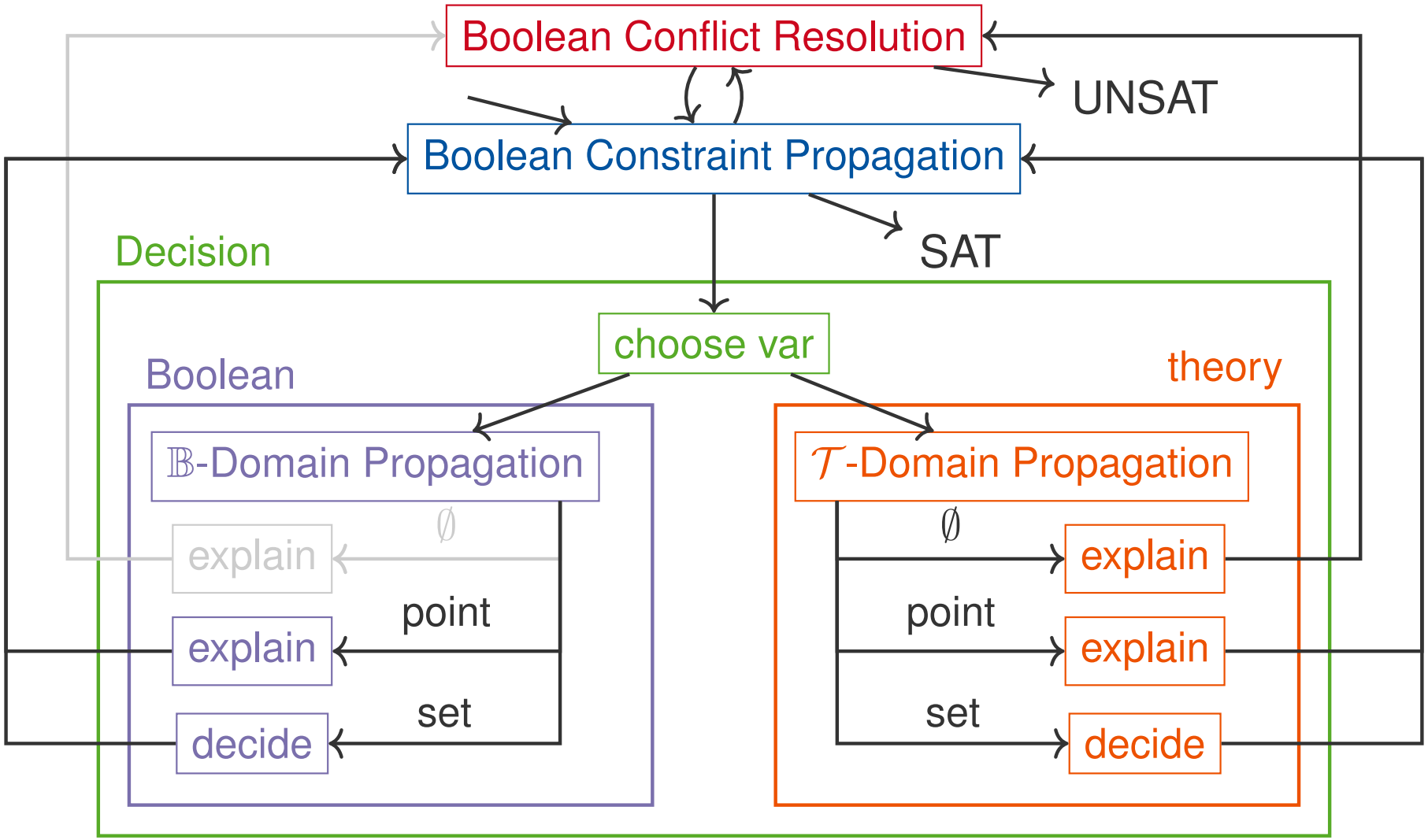
Our MCSAT Approach



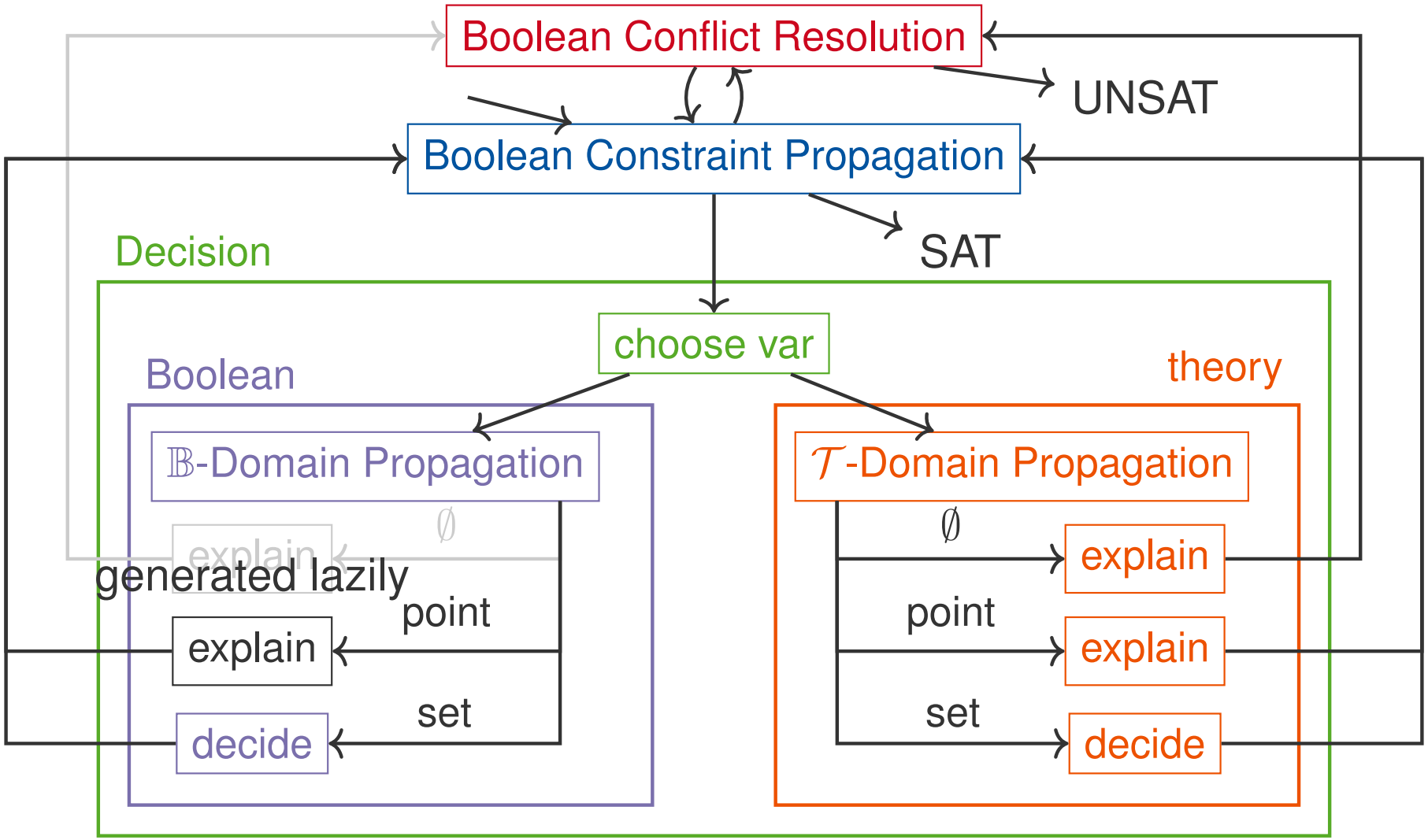
Our MCSAT Approach



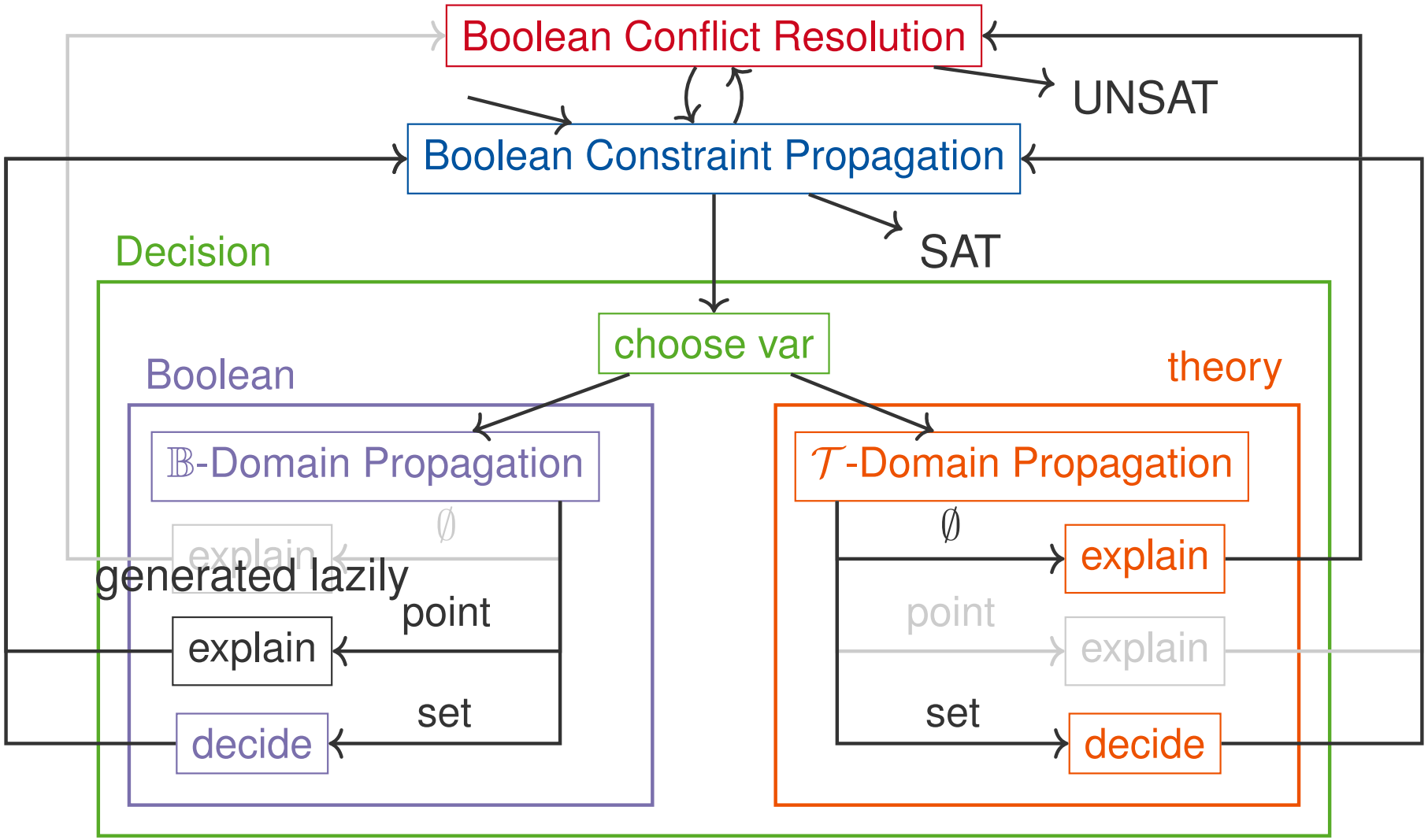
Our MCSAT Approach



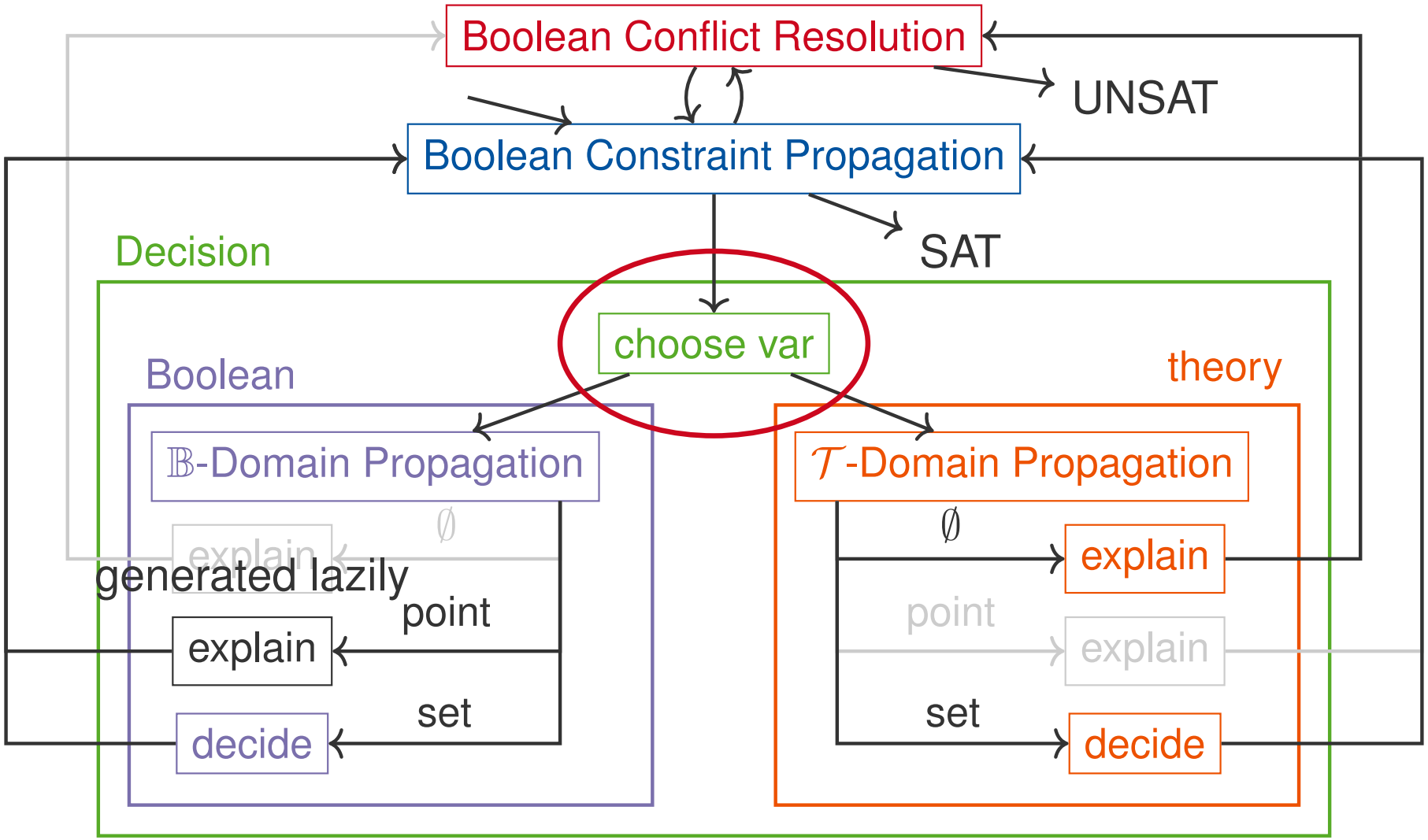
Our MCSAT Approach



Our MCSAT Approach



Our MCSAT Approach



NLSAT's strategy [Jovanović, De Moura 2012]

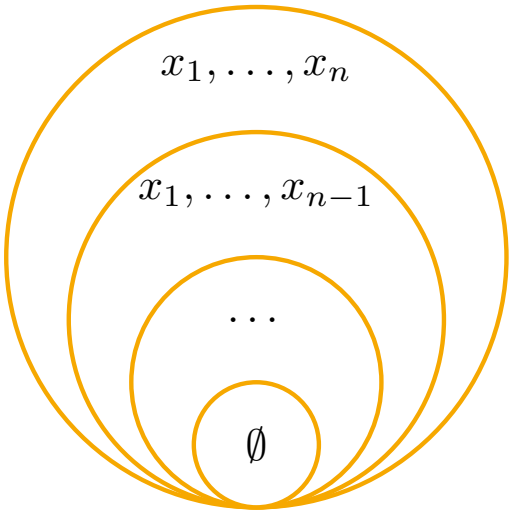
- static ordering of theory variables
- Boolean decisions: literal of a not-yet satisfied clause that is univariate in next theory variable
- theory decisions: only if no Boolean decision possible

RWTH AACHEN UNIVERSITY

Static vs dynamic orderings

Static orderings

- induce partial ordering on constraints/clauses

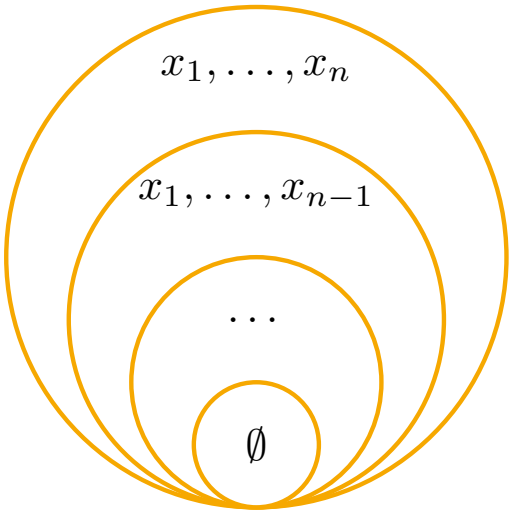


RWTH AACHEN UNIVERSITY

Static vs dynamic orderings

Static orderings

- induce partial ordering on constraints/clauses



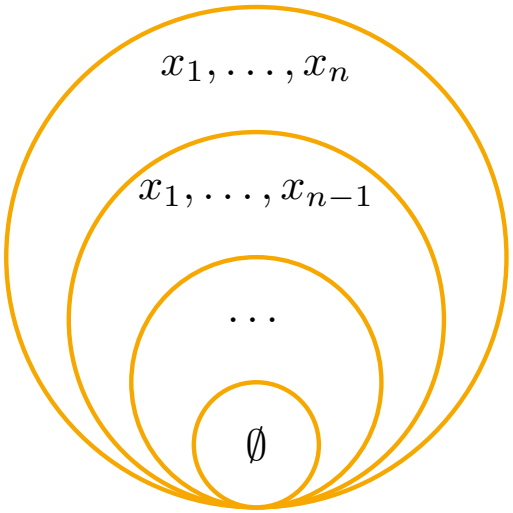
But: How to guess a good ordering in advance?

RWTH AACHEN UNIVERSITY

Static vs dynamic orderings

Static orderings

- induce partial ordering on constraints/clauses



But: How to guess a good ordering in advance?

Dynamic orderings

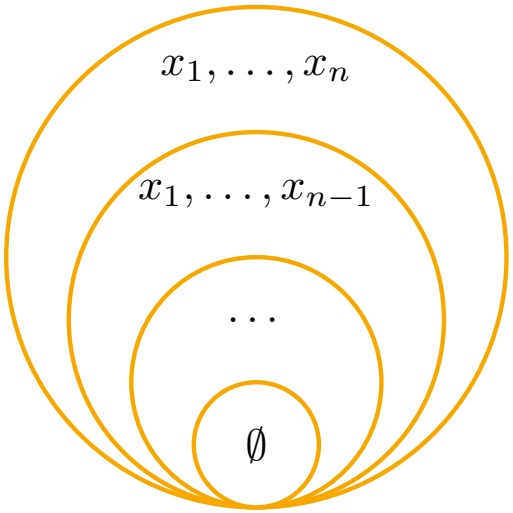
- assign any undecided theory variable
- SAT-like heuristics for theory

RWTH AACHEN UNIVERSITY

Static vs dynamic orderings

Static orderings

- induce partial ordering on constraints/clauses



But: How to guess a good ordering in advance?

Dynamic orderings

- assign any undecided theory variable
- SAT-like heuristics for theory

However: not straight-forward!

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \mathsf{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \mathsf{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Semantics: $\mathsf{root}_i(p(z; x_1, \ldots, x_n)) \mathrel{\hat{=}} i$th root of z in $p(z; x_1, \ldots, x_n)$

RWTH AACHEN UNIVERSITY

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \mathsf{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Assume variable ordering $x_1 < \ldots < x_n < y$

$[\![$ $]\!]$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \text{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Assume variable ordering $x_1 < \ldots < x_n < y$

$[\![\, x_1 \mapsto \alpha_1 \qquad \qquad \qquad \qquad \,]\!]$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \text{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Assume variable ordering $x_1 < \ldots < x_n < y$

$[\![x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2 \qquad]\!]$

RWTH AACHEN UNIVERSITY

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \mathsf{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Constraint on y

Assume variable ordering $x_1 < \ldots < x_n < y$

$[\![x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2, x_{n-1} \mapsto \alpha_{n-1}, x_n \mapsto \alpha_n]\!]$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \text{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Change variable ordering $x_1 < \ldots < x_{n-1} < y < x_n$

$[\![$ $]\!]$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \mathrm{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Change variable ordering $x_1 < \ldots < x_{n-1} < y < x_n$

$[\![x_1 \mapsto \alpha_1]\!]$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \text{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Change variable ordering $x_1 < \ldots < x_{n-1} < y < x_n$

$[\![x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2 \quad]\!]$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \text{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Change variable ordering $x_1 < \ldots < x_{n-1} < y < x_n$

$$[\![\, x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2, x_{n-1} \mapsto \alpha_{n-1} \qquad]\!]$$

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \text{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Change variable ordering $x_1 < \ldots < x_{n-1} < y < x_n$

$$[\![x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2, x_{n-1} \mapsto \alpha_{n-1}, y \mapsto \alpha_y \quad]\!]$$

RWTH AACHEN UNIVERSITY

Extended Polynomial Constraints

CAD explanation backend generates

$$y \sim \text{root}_i(p(z; x_1, x_2, \ldots, x_{n-1}, x_n))$$

Constraint on x_n

Change variable ordering $x_1 < \ldots < x_{n-1} < y < x_n$

$$[\![x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2, x_{n-1} \mapsto \alpha_{n-1}, y \mapsto \alpha_y, x_n \mapsto \lightning]\!]$$

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

$[\![y \sim \mathsf{root}_i(p(z, x_1, x_2, x_3))]\!]$

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

$$[\![y \sim \mathsf{root}_i(p(z, x_1, x_2, x_3)), x_1 \mapsto a_1]\!]$$

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

$$[\![y \not\sim \mathsf{root}_i(p(z, x_1, x_2, x_3)), x_1 \mapsto a_1, y \mapsto a_y]\!]$$

Constraint is temporarily disabled.

Handling incompatible constraints

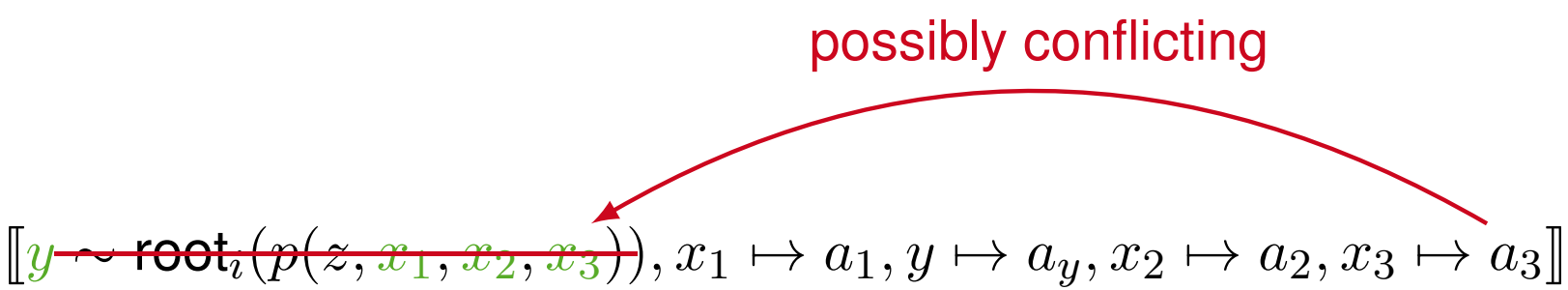
- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

$$[\![y \not\sim \mathsf{root}_i(p(z, x_1, x_2, x_3)), x_1 \mapsto a_1, y \mapsto a_y, x_2 \mapsto a_2]\!]$$

RWTH AACHEN UNIVERSITY

Handling incompatible constraints

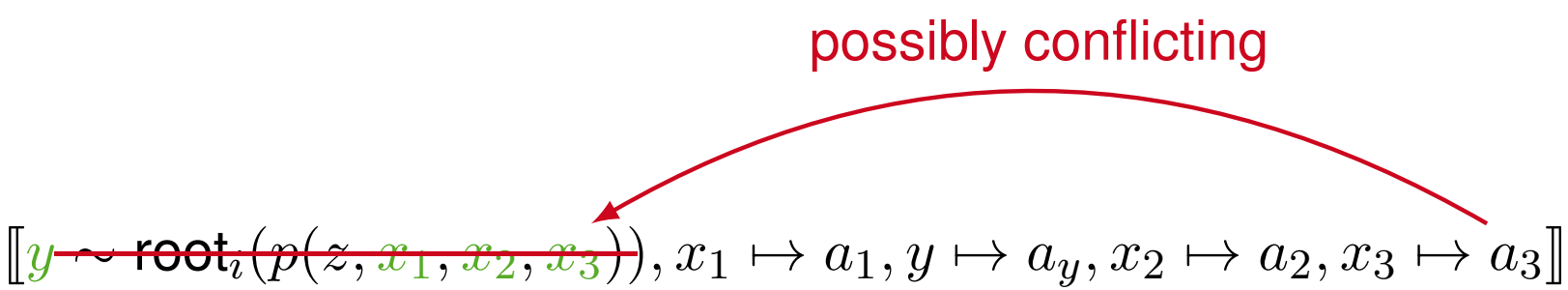
- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them



RWTH AACHEN UNIVERSITY

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them



Conflict reoccurs once per variable ordering!

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

$$[\![y \not\sim \mathsf{root}_i(p(z, x_1, x_2, x_3)), x_1 \mapsto a_1, y \mapsto a_y, x_2 \mapsto a_2]\!]$$

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

$$[\![y \not\sim \mathsf{root}_i(p(z, x_1, x_2, x_3)), x_1 \mapsto a_1, y \mapsto a_y]\!]$$

Handling incompatible constraints

- Boolean reasoning: treat them as any other constraint
- theory reasoning: simply ignore them

$$[\![y \sim \mathsf{root}_i(p(z, x_1, x_2, x_3)), x_1 \mapsto a_1]\!]$$

Constraint can safely be re-enabled!

Completeness of MCSAT

Definition (Finite Basis Property)

All literals occurring in the solving process stem from a finite set depending on the input formula and the explanation backend.

Completeness of MCSAT

Definition (Finite Basis Property)

All literals occurring in the solving process stem from a finite set depending on the input formula and the explanation backend.

- The completeness proof of MCSAT relies on the finite basis property!

Finite Basis for Quantifier Elimination Procedures

Assume $x_1 < x_2 < \ldots < x_n$

Finite Basis for Quantifier Elimination Procedures

Assume $x_1 < x_2 < \ldots < x_n$

Observe: Conflict consists of

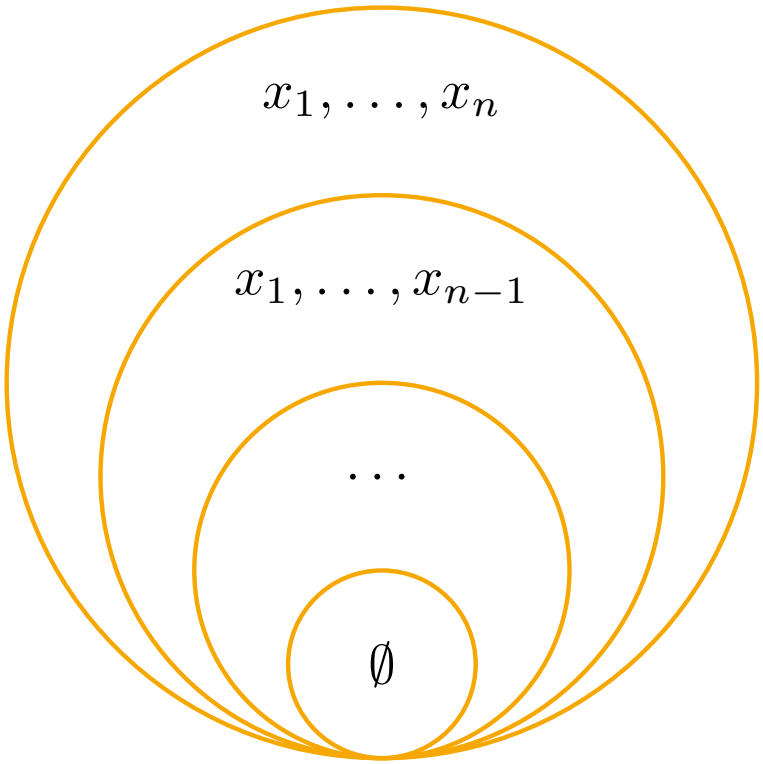
- assigned variables $x_1, \ldots, x_{k-1}$
- conflicting variable x_k
- constraints in $x_1, \ldots, x_k$

Finite Basis for Quantifier Elimination Procedures

Assume $x_1 < x_2 < \ldots < x_n$

Observe: Conflict consists of

- assigned variables $x_1, \ldots, x_{k-1}$
- conflicting variable x_k
- constraints in $x_1, \ldots, x_k$

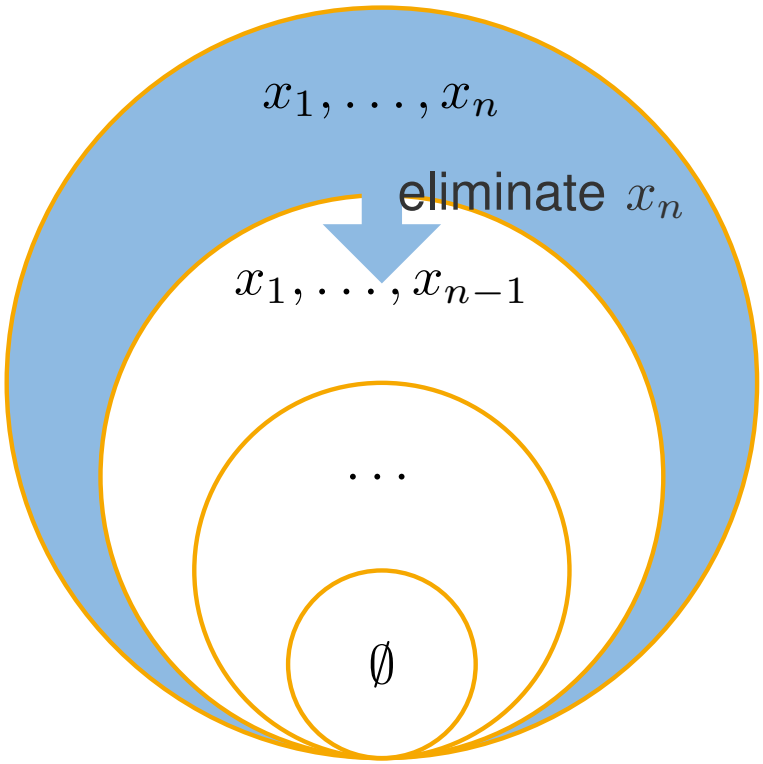


Finite Basis for Quantifier Elimination Procedures

Assume $x_1 < x_2 < \ldots < x_n$

Observe: Conflict consists of

- assigned variables $x_1, \ldots, x_{k-1}$
- conflicting variable x_k
- constraints in $x_1, \ldots, x_k$



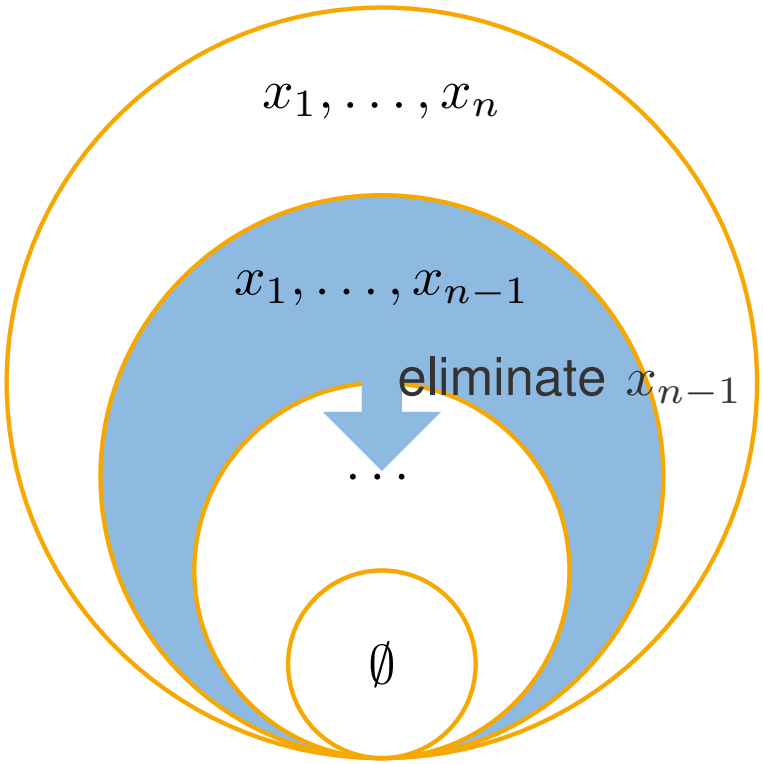
RWTH AACHEN UNIVERSITY

Finite Basis for Quantifier Elimination Procedures

Assume $x_1 < x_2 < \ldots < x_n$

Observe: Conflict consists of

- assigned variables $x_1, \ldots, x_{k-1}$
- conflicting variable x_k
- constraints in $x_1, \ldots, x_k$

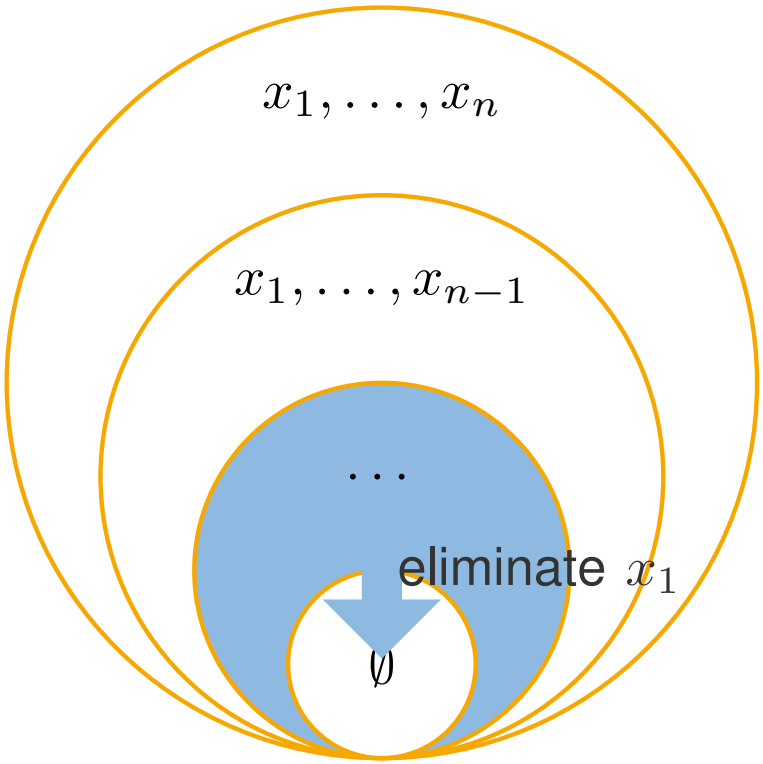


Finite Basis for Quantifier Elimination Procedures

Assume $x_1 < x_2 < \ldots < x_n$

Observe: Conflict consists of

- assigned variables $x_1, \ldots, x_{k-1}$
- conflicting variable x_k
- constraints in $x_1, \ldots, x_k$



Finite Basis vs Dynamic Theory Ordering

$x_1 = 2x_2$

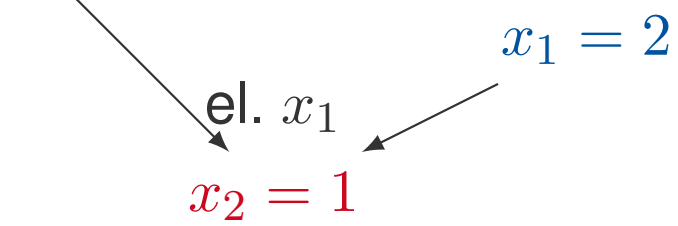
$x_2 = 2x_1$

$x_1 = 2$

Finite Basis vs Dynamic Theory Ordering

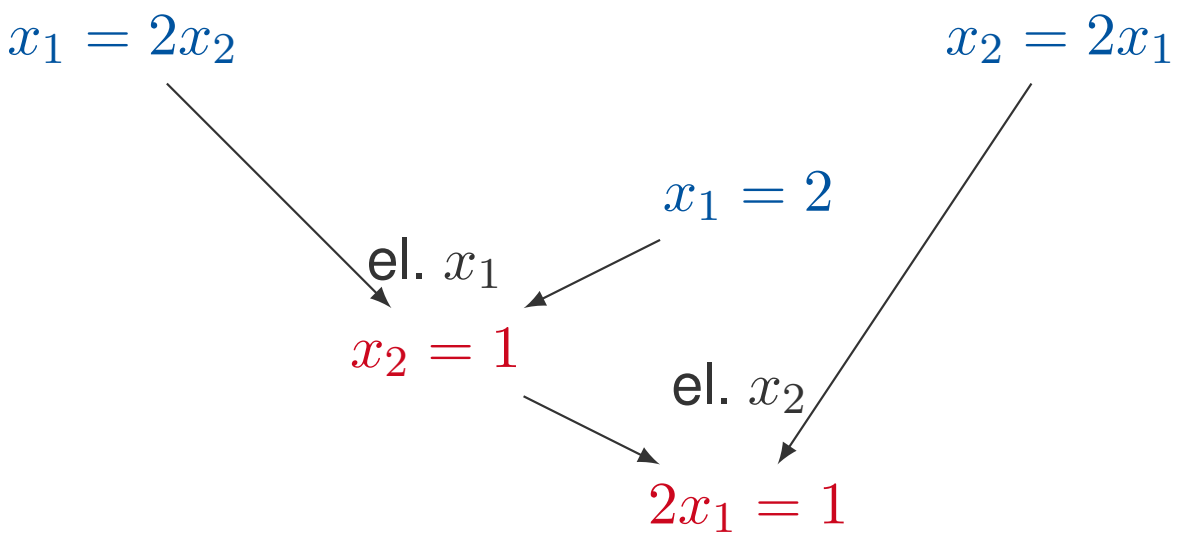
$x_1 = 2x_2$

$x_2 = 2x_1$



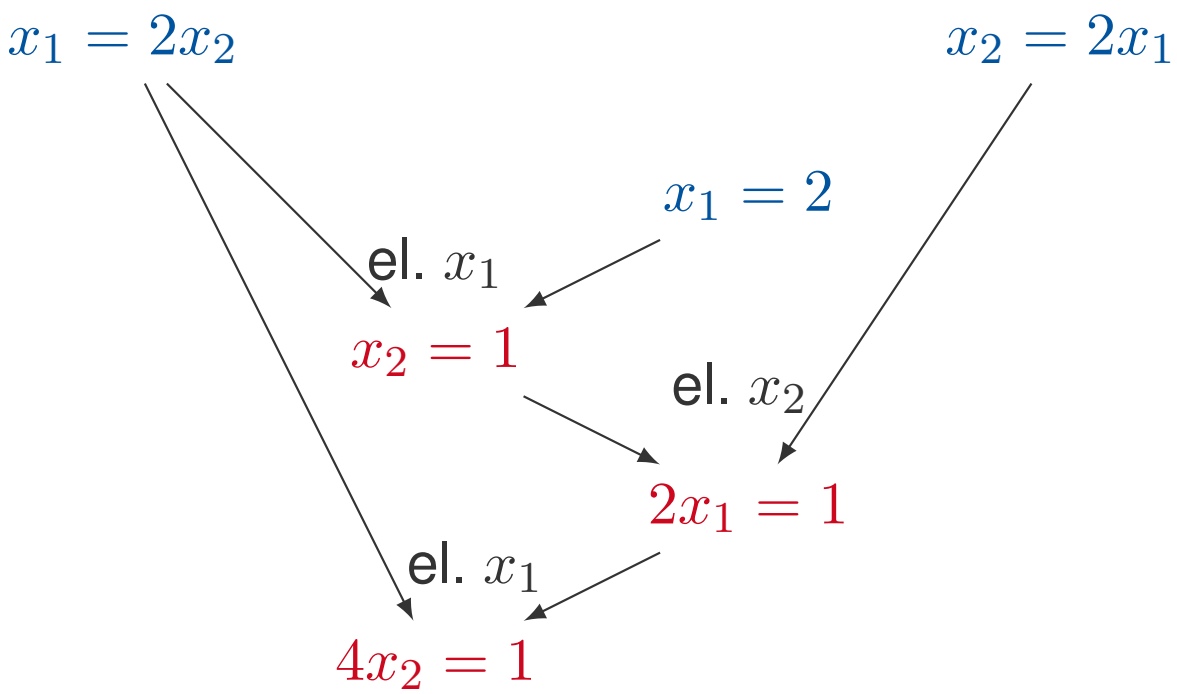
RWTH AACHEN UNIVERSITY

Finite Basis vs Dynamic Theory Ordering



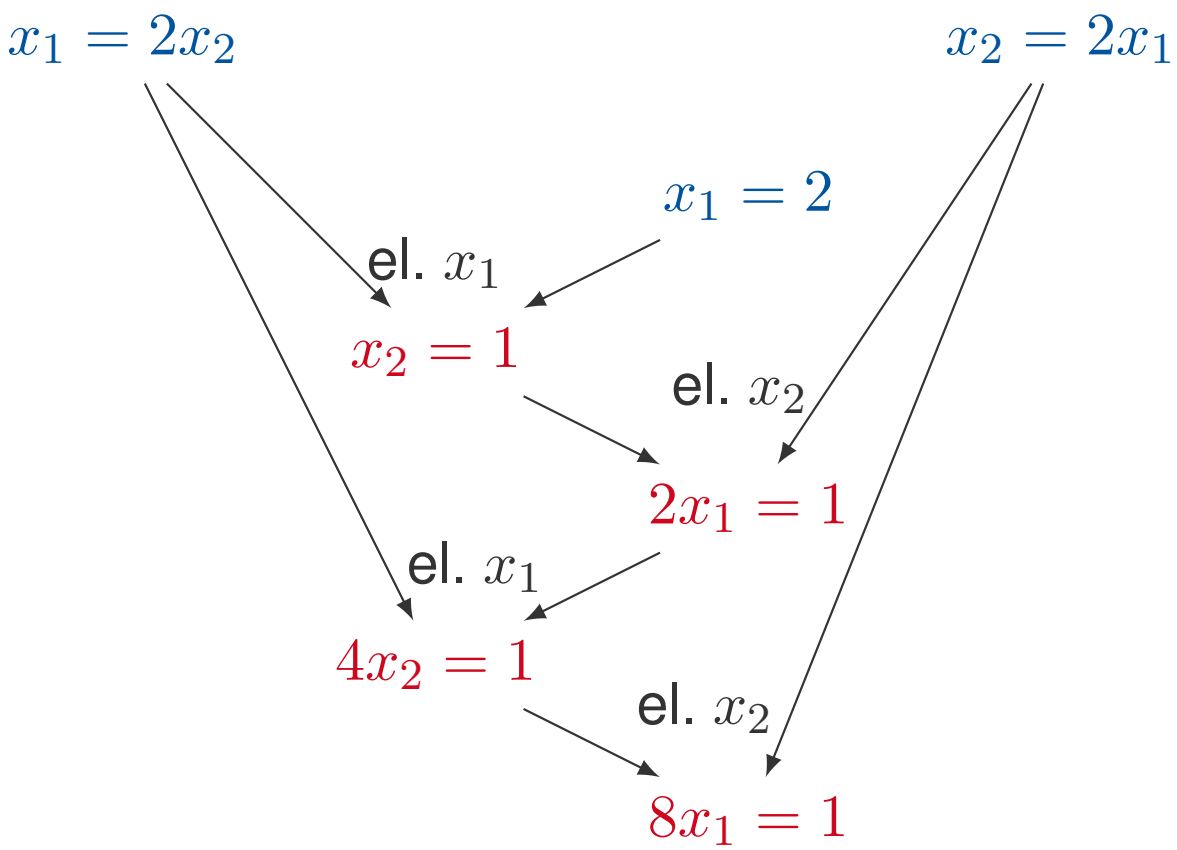
RWTH AACHEN UNIVERSITY

Finite Basis vs Dynamic Theory Ordering



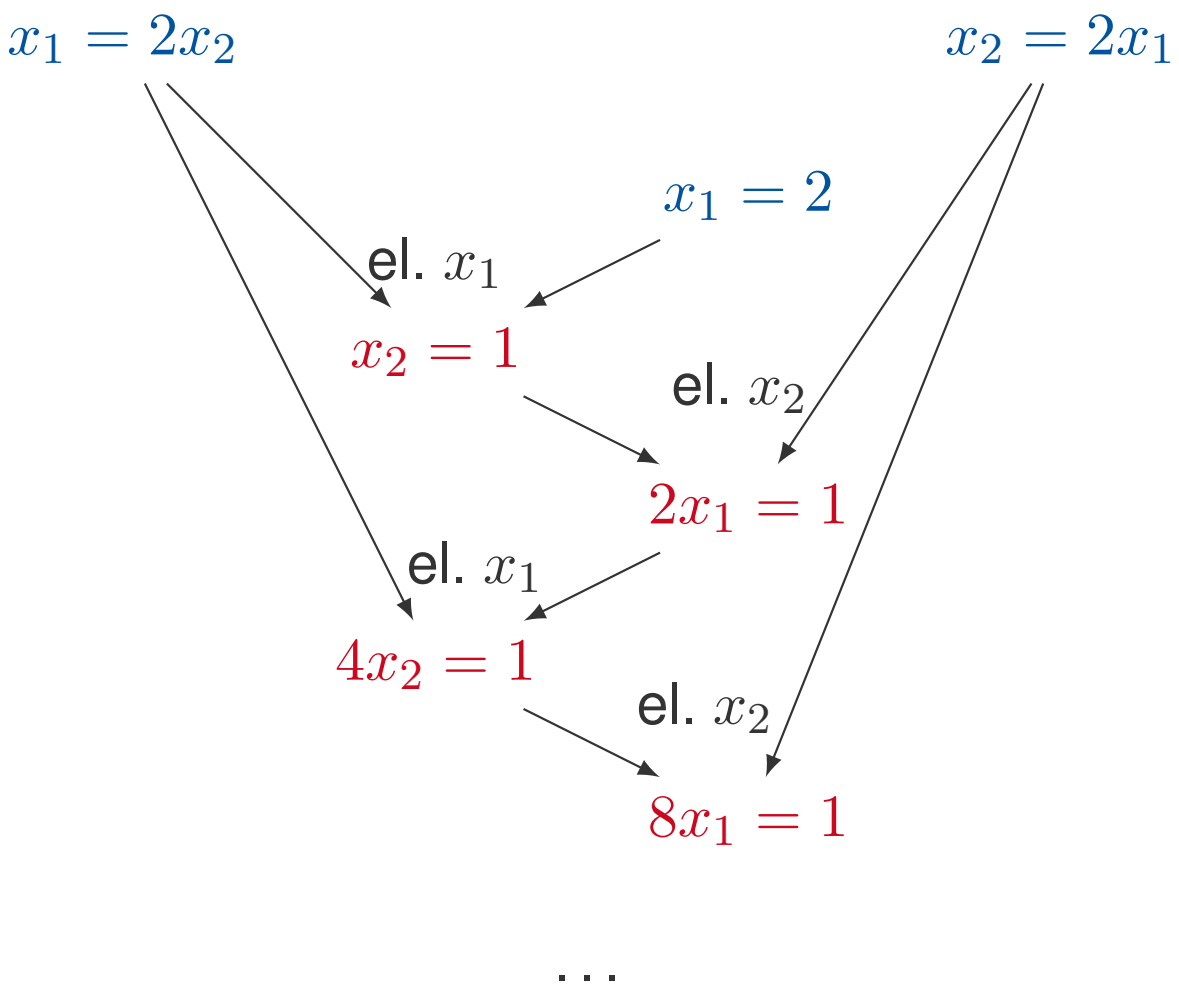
RWTH AACHEN UNIVERSITY

Finite Basis vs Dynamic Theory Ordering



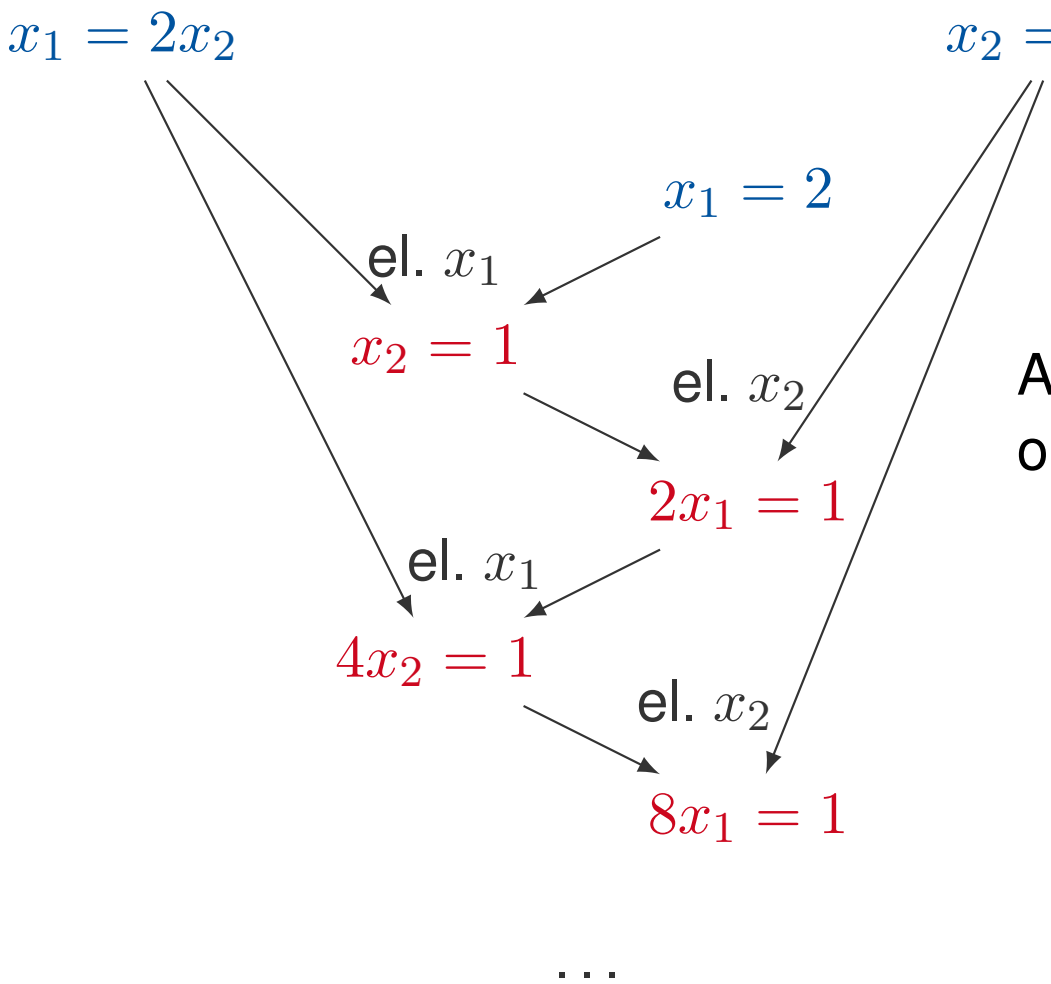
RWTH AACHEN UNIVERSITY

Finite Basis vs Dynamic Theory Ordering



RWTH AACHEN UNIVERSITY

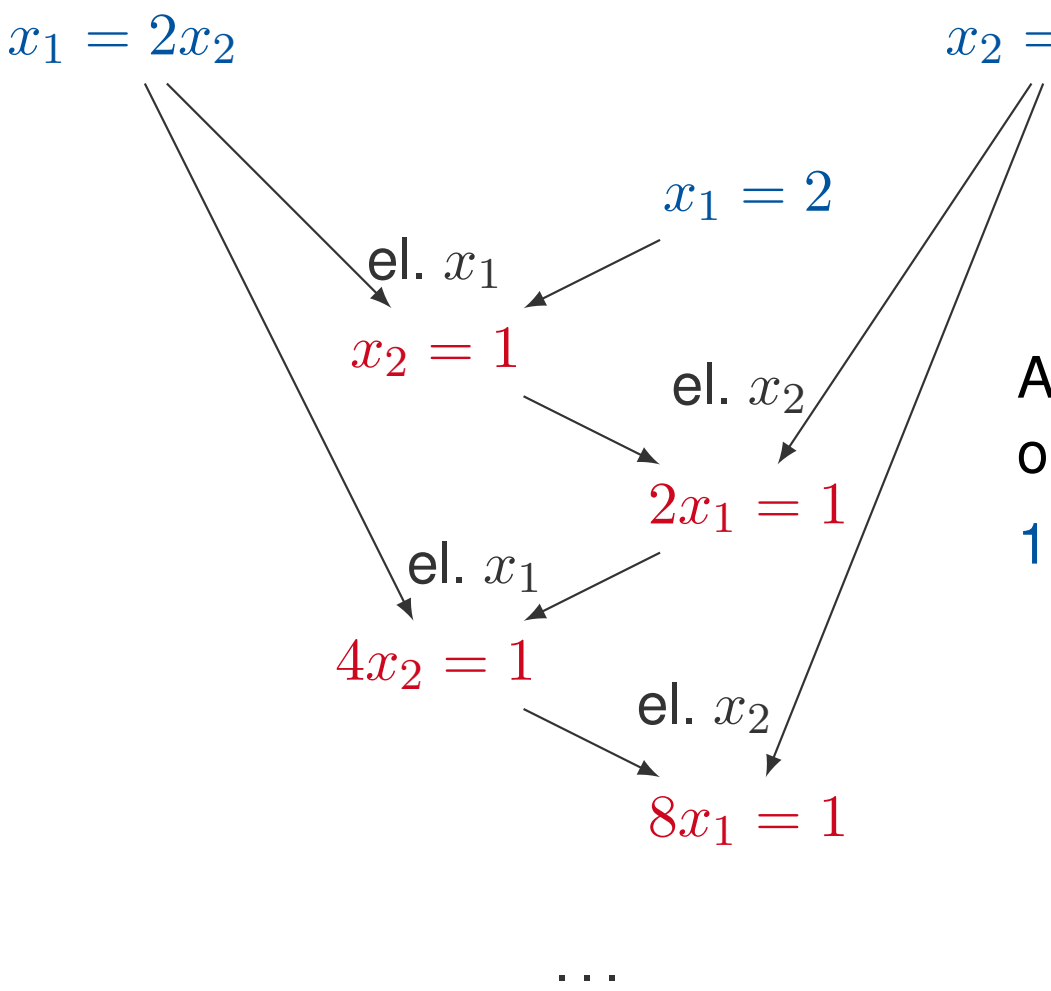
Finite Basis vs Dynamic Theory Ordering



Assume finite basis under fixed ordering. Two solutions:

RWTH AACHEN UNIVERSITY

Finite Basis vs Dynamic Theory Ordering

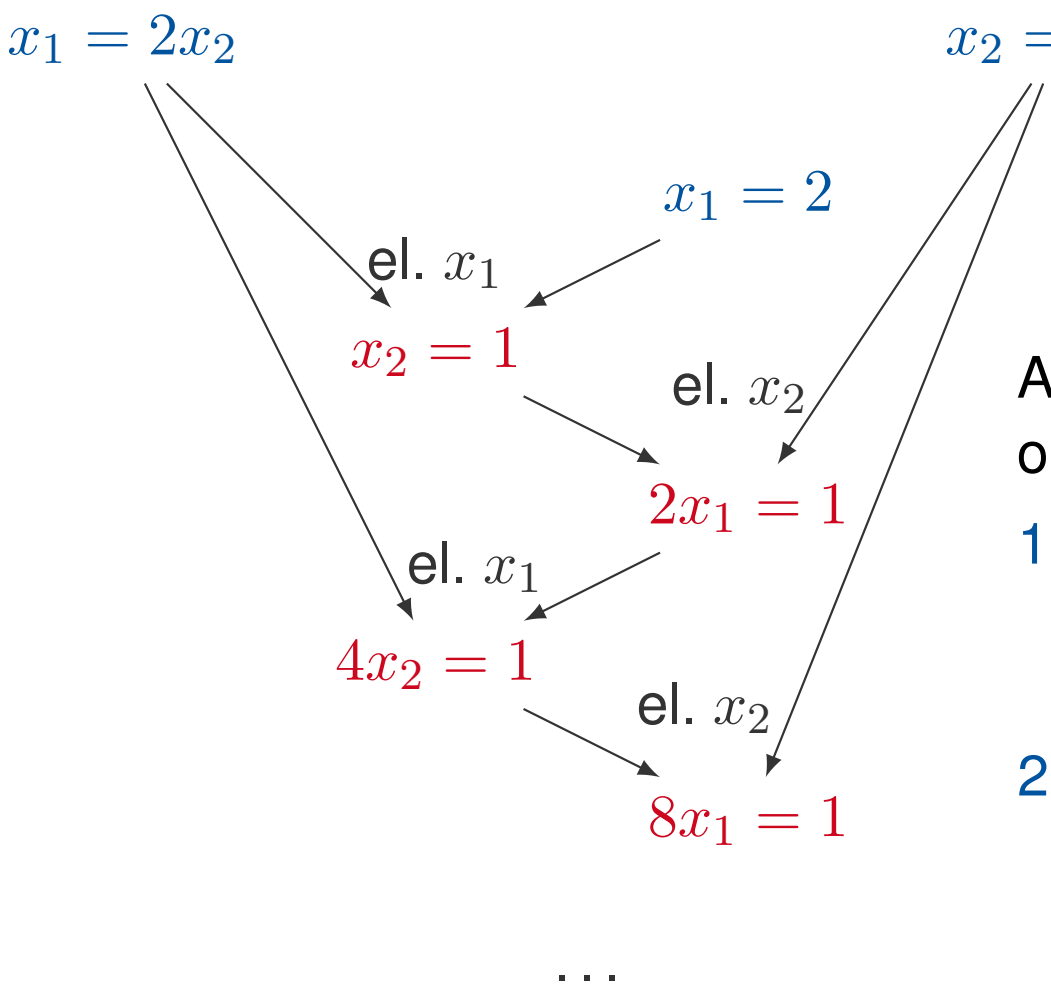


Assume finite basis under fixed ordering. Two solutions:

1. Fix an ordering from some point onwards
   (similar argument as for restarts)

RWTH AACHEN UNIVERSITY

Finite Basis vs Dynamic Theory Ordering



Assume finite basis under fixed ordering. Two solutions:

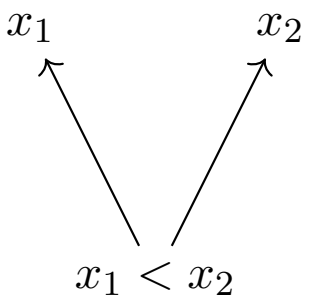
1. Fix an ordering from some point onwards
   (similar argument as for restarts)
2. Disable clauses/constraints generated under an incompatible partial ordering

Termination: Compatible Orderings

x_1

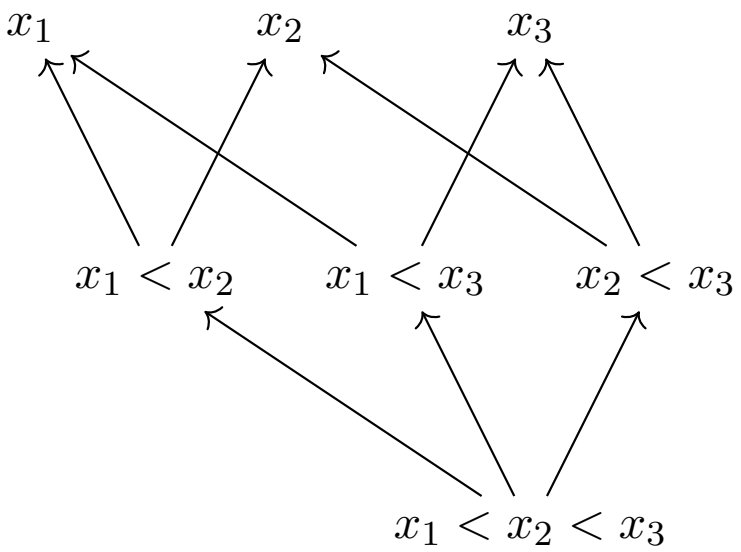
$[\![x_1 \mapsto ?]\!]$

Termination: Compatible Orderings



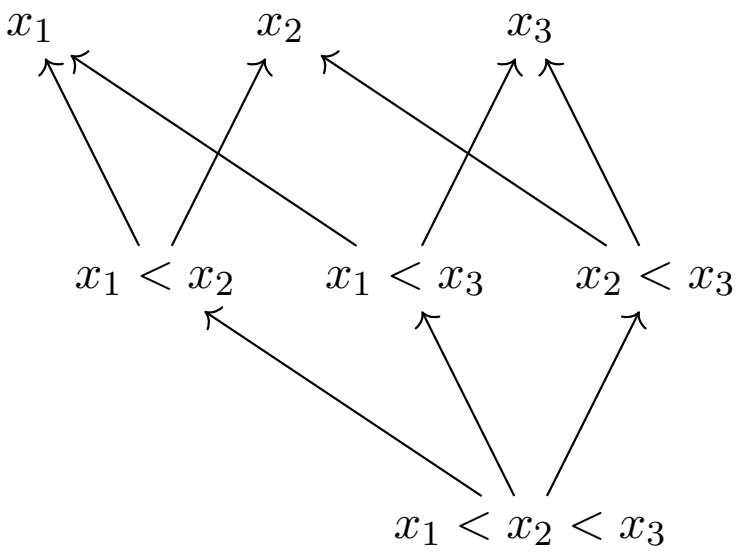
$[\![x_1 \mapsto \alpha_1, x_2 \mapsto ?]\!]$

Termination: Compatible Orderings



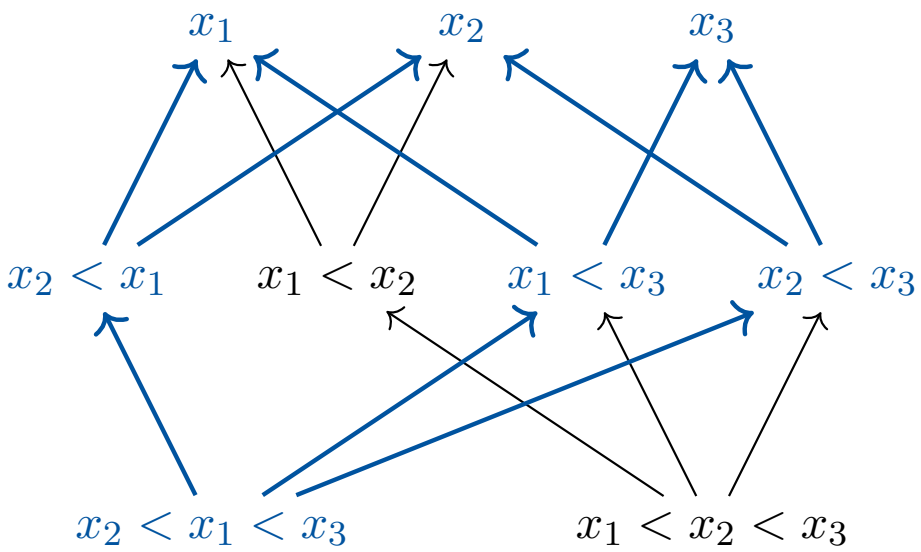
$[\![x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2, x_3 \mapsto ?]\!]$

Termination: Compatible Orderings



$[\![x_1 \mapsto \alpha_1, x_2 \mapsto \alpha_2, x_3 \mapsto ?]\!]$

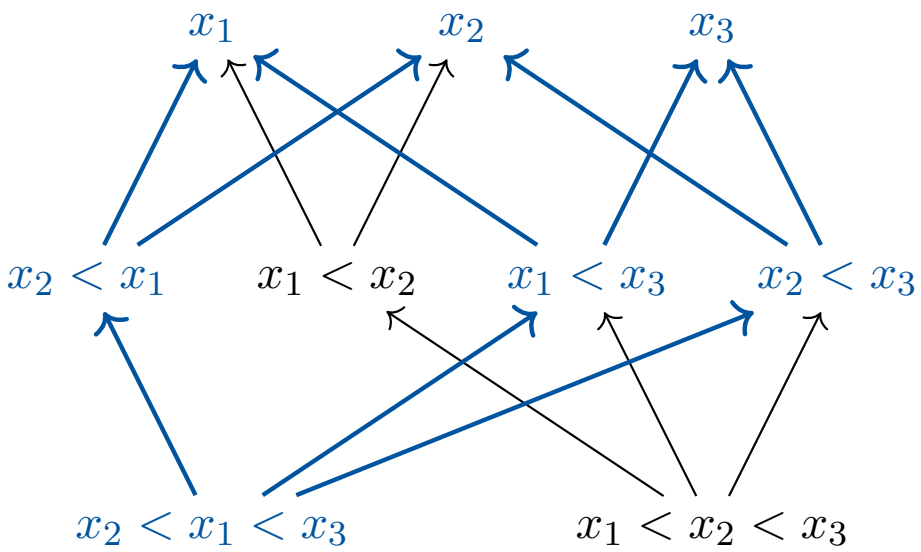
Termination: Compatible Orderings



$[\![x_2 \mapsto \alpha_2, x_1 \mapsto \alpha_1, x_3 \mapsto ?]\!]$

RWTH AACHEN UNIVERSITY

Termination: Compatible Orderings



$[\![x_2 \mapsto \alpha_2, x_1 \mapsto \alpha_1, x_3 \mapsto ?]\!]$

Our intuition: Dynamic orderings let us find a good ordering, but progress is not made by changing the ordering itself!

Experiments: Heuristics

$\mathbb{B}^{dynamic}\ \mathcal{T}^{static}$

first all Boolean (activities), then theory (static)

RWTH AACHEN UNIVERSITY

Experiments: Heuristics

$\mathbb{B}^{dynamic}\ \mathcal{T}^{static}$

first all Boolean (activities), then theory (static)

$\mathcal{T}^{static}\ \mathbb{B}^{dynamic}$

first all theory (static), then Boolean (activities)

Experiments: Heuristics

$\mathbb{B}^{dynamic}\ \mathcal{T}^{static}$

first all Boolean (activities), then theory (static)

$\mathcal{T}^{static}\ \mathbb{B}^{dynamic}$

first all theory (static), then Boolean (activities)

$\mathbb{B}^{dynamic}_{univariate}\ \mathcal{T}^{static}$

univariate constraints w.r.t. next theory var, then next theory variable (static)

Experiments: Heuristics

$\mathbb{B}^{dynamic}\ \mathcal{T}^{static}$

first all Boolean (activities), then theory (static)

$\mathcal{T}^{static}\ \mathbb{B}^{dynamic}$

first all theory (static), then Boolean (activities)

$\mathbb{B}^{dynamic}_{univariate}\ \mathcal{T}^{static}$

univariate constraints w.r.t. next theory var, then next theory variable (static)

$\mathbb{B}^{dynamic}_{active}\ \mathcal{T}^{static}$

univariate constraints occurring in not-yet-satisfied clauses, then next theory variable (static)

most similar to NLSAT strategy

RWTH AACHEN UNIVERSITY

Experiments: Heuristic

$(\mathbb{B} \cup \mathcal{T})^{dynamic}$

Boolean and theory variables treat equally, ordered by activities

applied for LRA [Jovanovic, Barrett, De Moura 2013].

RWTH AACHEN UNIVERSITY

Experiments: Heuristic

$(\mathbb{B} \cup \mathcal{T})^{dynamic}$

Boolean and theory variables treat equally, ordered by activities

applied for LRA [Jovanovic, Barrett, De Moura 2013].

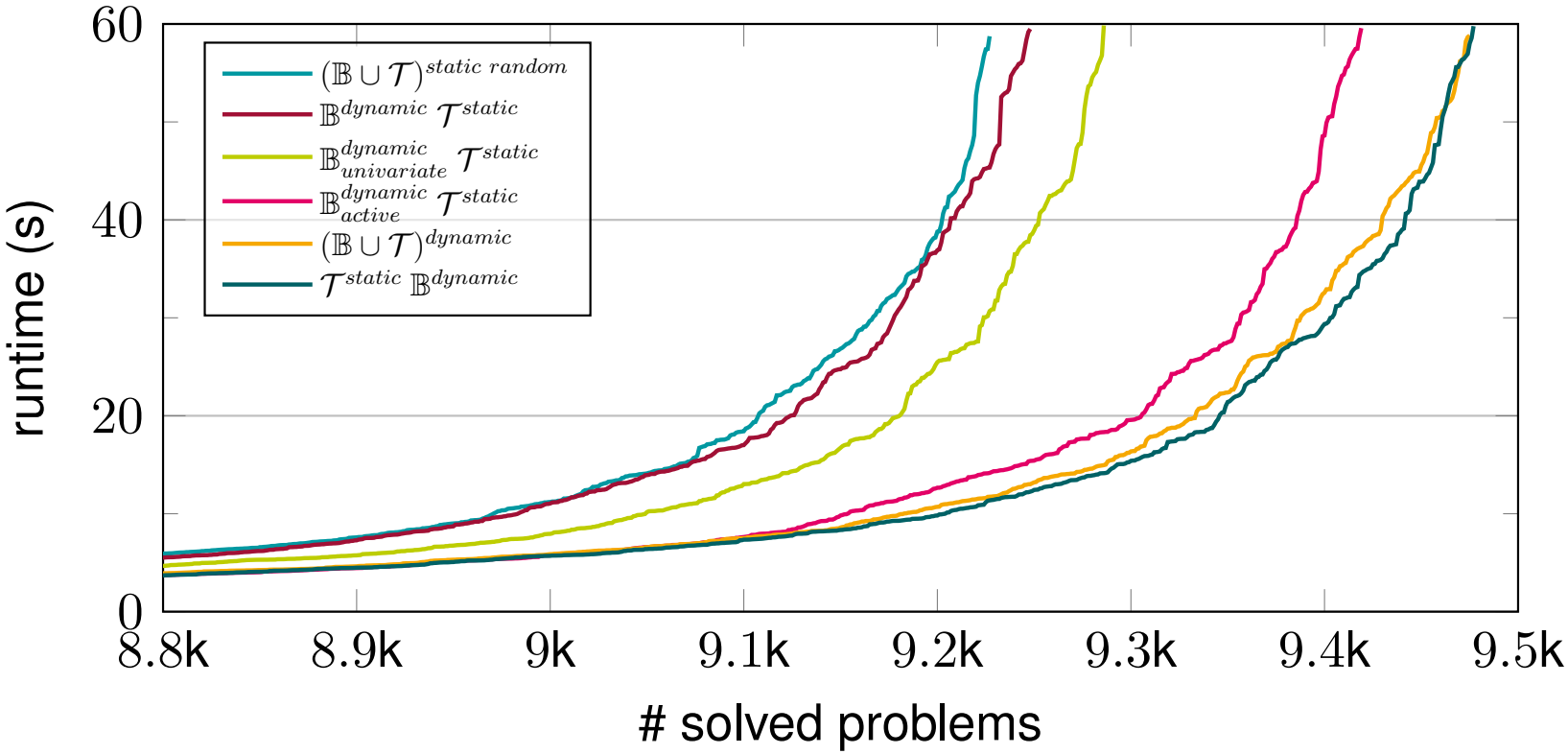
$(\mathbb{B} \cup \mathcal{T})^{static\ random}$

random ordering on all variables is fixed at beginning

for reference

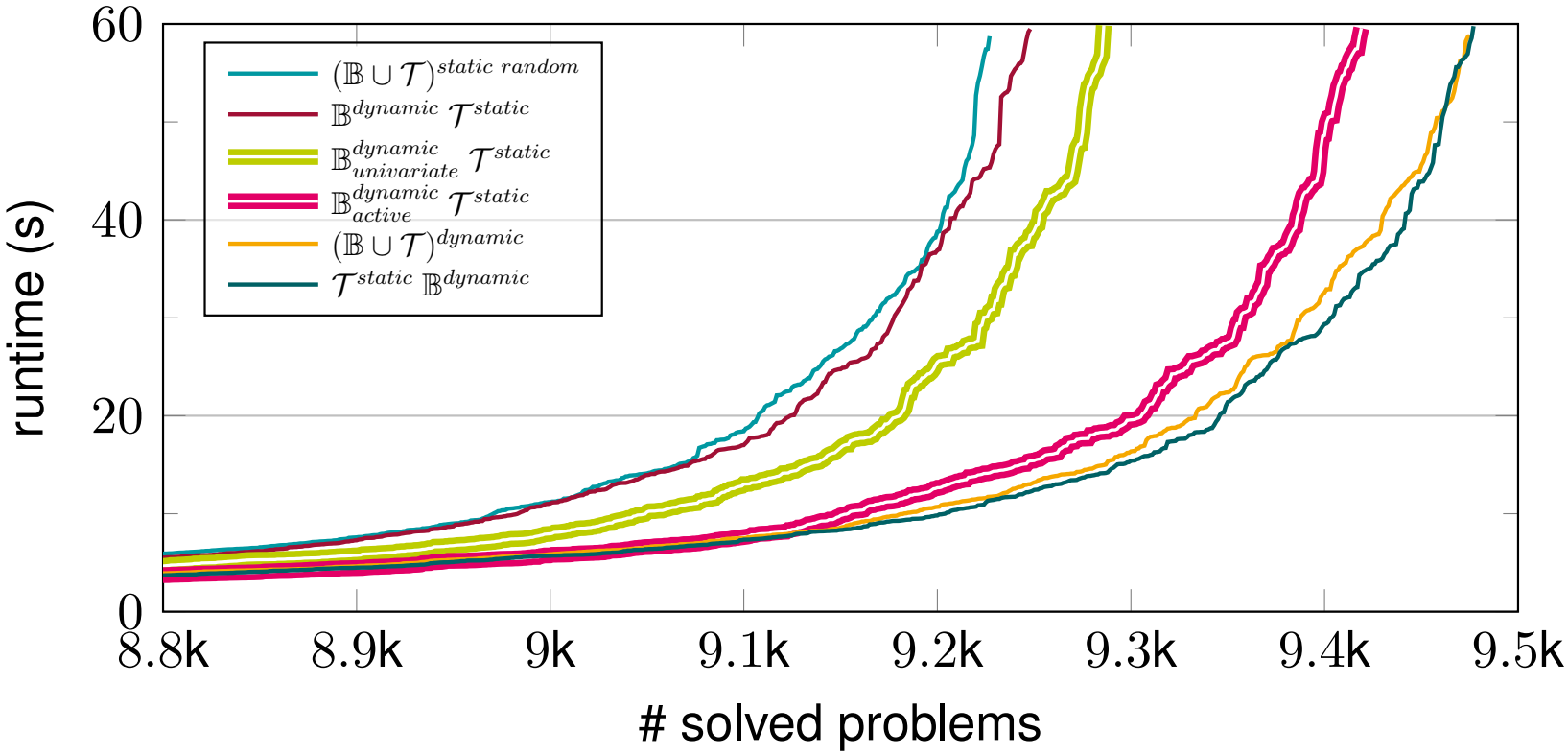
RWTH AACHEN UNIVERSITY

Experiments: Results on SMTLIB benchmarks



RWTH AACHEN UNIVERSITY

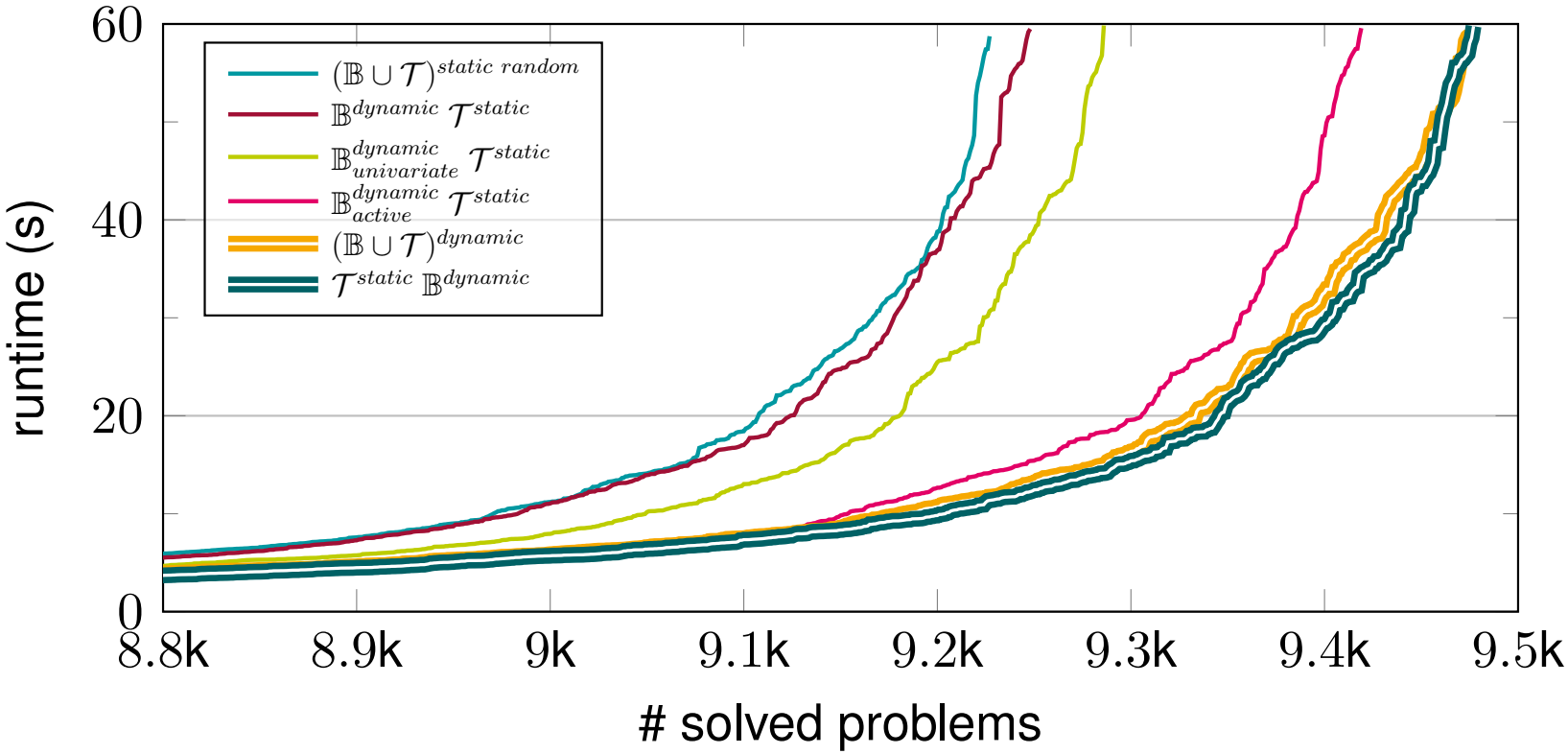
Experiments: Results on SMTLIB benchmarks



- $\mathbb{B}^{dynamic}_{univariate}\ \mathcal{T}^{static}$ and $\mathbb{B}^{dynamic}_{active}\ \mathcal{T}^{static}$ delay Boolean decisions, avoid unnecessary conflicts, interlave model construction

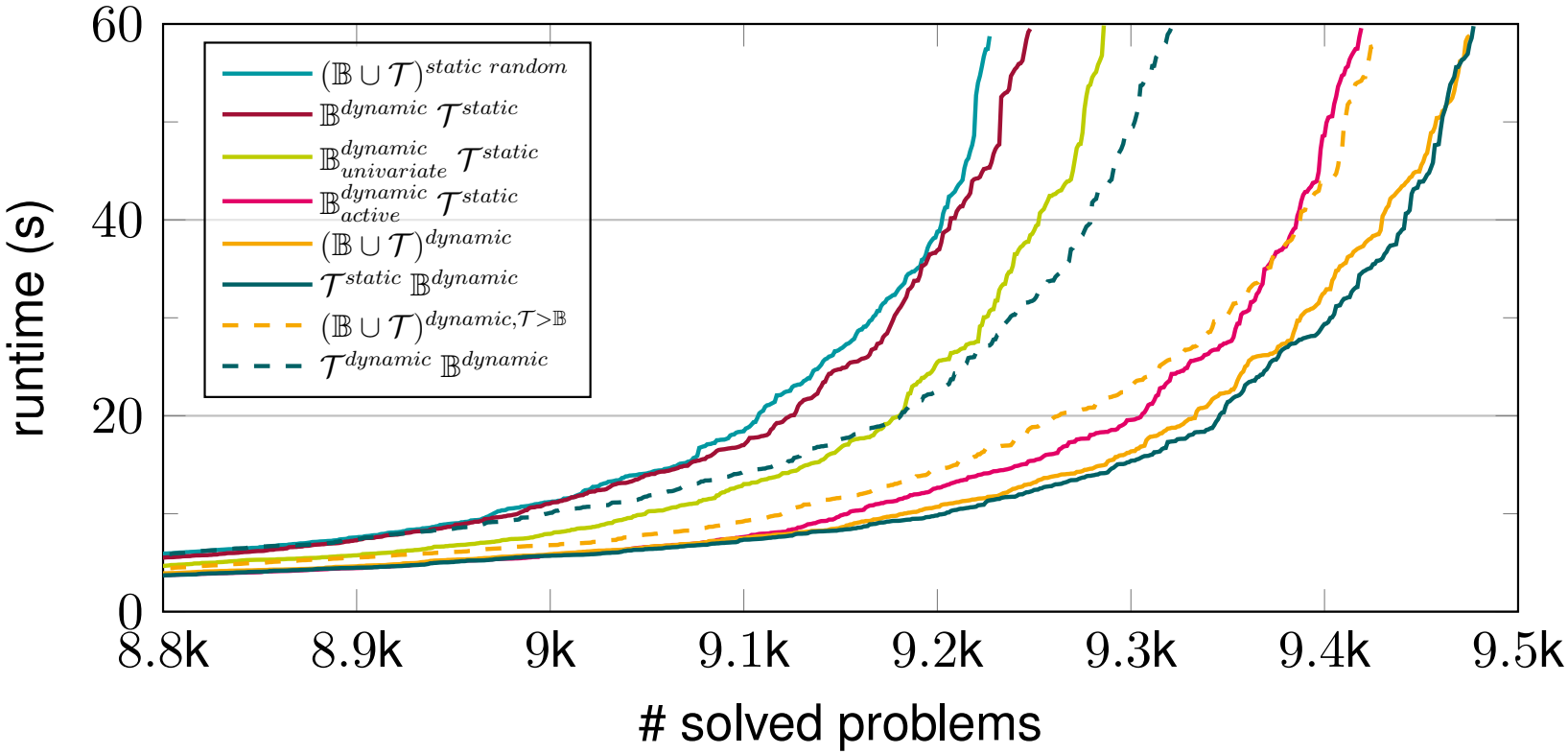
RWTH AACHEN UNIVERSITY

Experiments: Results on SMTLIB benchmarks



- $\mathcal{T}^{static}\ \mathbb{B}^{dynamic}$: almost no interleaving of model construction!
- Does $(\mathbb{B} \cup \mathcal{T})^{dynamic}$ converge towards $\mathcal{T}^{static}\ \mathbb{B}^{dynamic}$?

Experiments: Results on SMTLIB benchmarks



- $(\mathbb{B} \cup \mathcal{T})^{dynamic,\mathcal{T}>\mathbb{B}}$: favor theory in case of equal activities
- $\mathcal{T}^{dynamic}\ \mathbb{B}^{dynamic}$: theory ordering is activity-based

Further Considerations

- domain propagation

  Simplex on linear parts and complete method

Further Considerations

- domain propagation

  Simplex on linear parts and complete method

- explanation backend

  Fourier-Motzkin, Virtual Substitution and (One Cell -) CAD

Open Questions

We saw

- ordering has high impact on running time,
- completeness issues for dynamic theory orderings can be circumvented in practise,
- first insights into orderings.

Open Questions

We saw

- ordering has high impact on running time,
- completeness issues for dynamic theory orderings can be circumvented in practise,
- first insights into orderings.

Can we benefit from

- interleaving of Boolean and theory decisions?
- dynamic theory orderings?
- known heuristics from computer algebra and SAT? (Machine Learning?)

Open Questions

We saw

- ordering has high impact on running time,
- completeness issues for dynamic theory orderings can be circumvented in practise,
- first insights into orderings.

Can we benefit from

- interleaving of Boolean and theory decisions?
- dynamic theory orderings?
- known heuristics from computer algebra and SAT? (Machine Learning?)

- How do we prevent overfitting?
- Are there general strategies?
- Do we have to specialize to certain problem classes?