CS:4980 Topics in Computer Science II

Introduction to Automated Reasoning

Theory Solvers I

Cesare Tinelli

Spring 2024

The
University
of Iowa

Credits

These slides are based on slides originally developed by **Cesare Tinelli** at the University of Iowa, and by **Clark Barrett**, **Caroline Trippel**, and **Andrew (Haoze) Wu** at Stanford University. Adapted by permission.

Roadmap for Today

Theory Solvers

- Difference Logic
- Equality and Uninterpreted Functions
- Arrays

Theory Solvers

A *theory solver* for a theory $\mathcal{T}$ is a specialized procedure for determining whether a conjunction of literals is satisfiable in $\mathcal{T}$

Theory solvers are crucial building blocks in SMT solvers

Theory Solvers

A *theory solver* for a theory $\mathcal{T}$ is a specialized procedure for determining whether a conjunction of literals is satisfiable in $\mathcal{T}$

Theory solvers are crucial building blocks in SMT solvers

A Fragment of Arithmetic: Difference Logic

Difference logic is a fragment of integer arithmetic consisting of conjunction of literals of a very restricted form:

$$x - y \bowtie c$$

where x and y are integer variables, c is a numeral, and $\bowtie\ \in\ \{=, <, \leq, >, \geq\}$

A Fragment of Arithmetic: Difference Logic

Difference logic is a fragment of integer arithmetic consisting of conjunction of literals of a very restricted form:

$$x - y \bowtie c$$

where x and y are integer variables, c is a numeral, and $\bowtie \in \{=, <, \leq, >, \geq\}$

A Fragment of Arithmetic: Difference Logic

Difference logic is a fragment of integer arithmetic consisting of conjunction of literals of a very restricted form:

$$x - y \bowtie c$$

where x and y are integer variables, c is a numeral, and $\bowtie\ \in\ \{=, <, \leq, >, \geq\}$

Note: There is a similar version of difference logic over the reals, which we will not cover, where x and y are integer variables and c is a decimal numeral

Difference Logic

A solver for difference logic consists of three steps:

1. Literal normalization
2. Conversion to a graph
3. Cycle detection in the graph

Difference Logic

Step 1

Rewrite each literal in terms of $\leq$ by applying these transformations to completion:

1. $x - y = c \quad \longrightarrow \quad x - y \leq c \wedge x - y \geq c$
2. $x - y \geq c \quad \longrightarrow \quad y - x \leq -c$
3. $x - y > c \quad \longrightarrow \quad y - x < -c$
4. $x - y < c \quad \longrightarrow \quad x - y \leq c - 1$

Difference Logic

Step 2

From the resulting literals of Step 1, construct a weighted directed graph G with a vertex for each variable

Add the edge $x \xrightarrow{c} y$ to G for each literal $x - y \leq c$

Step 3

Look for a cycle in G where the sum of the weights on the edges is negative

Return UNSAT if there is such a cycle and return SAT otherwise

Note: There are a number of efficient algorithms for detecting negative cycles in graphs

- e.g., Bellman-Ford, $O(v \cdot e)$ where v is the number of vertices and e the number of edges

Difference Logic

Step 2

From the resulting literals of Step 1, construct a weighted directed graph G with a vertex for each variable

Add the edge $x \xrightarrow{c} y$ to G for each literal $x - y \leq c$

Step 3

Look for a cycle in G where the sum of the weights on the edges is negative

Return UNSAT if there is such a cycle and return SAT otherwise

Note: There are a number of efficient algorithms for detecting negative cycles in graphs

- e.g., Bellman-Ford, $O(v \cdot e)$ where v is the number of vertices and e the number of edges

Difference Logic

Step 2

From the resulting literals of Step 1, construct a weighted directed graph G with a vertex for each variable

Add the edge $x \xrightarrow{c} y$ to G for each literal $x - y \leq c$

Step 3

Look for a cycle in G where the sum of the weights on the edges is negative

Return UNSAT if there is such a cycle and return SAT otherwise

Note: There are a number of efficient algorithms for detecting negative cycles in graphs

- e.g., Bellman-Ford, $O(v \cdot e)$ where v is the number of vertices and e the number of edges

Difference Logic Example

$$x - y = 5 \;\wedge\; z - y \geq 2 \;\wedge\; z - x > 2 \;\wedge\; w - x = 2 \;\wedge\; z - w < 0$$

Difference Logic Example

$$x - y = 5 \ \land \ z - y \geq 2 \ \land \ z - x > 2 \ \land \ w - x = 2 \ \land \ z - w < 0$$

$$\begin{aligned} &x - y = 5 \\ &z - y \geq 2 \\ &z - x > 2 \\ &w - x = 2 \\ &z - w < 0 \end{aligned}$$

Difference Logic Example

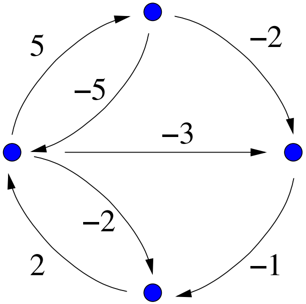
$$x - y = 5 \;\wedge\; z - y \geq 2 \;\wedge\; z - x > 2 \;\wedge\; w - x = 2 \;\wedge\; z - w < 0$$

$$\begin{array}{lcl}
x - y = 5 & & x - y \leq 5 \;\wedge\; y - x \leq -5 \\
z - y \geq 2 & & y - z \leq -2 \\
z - x > 2 & \longrightarrow & x - z \leq -3 \\
w - x = 2 & & w - x \leq 2 \;\wedge\; x - w \leq -2 \\
z - w < 0 & & z - w \leq -1
\end{array}$$

Difference Logic Example

$$x - y = 5 \;\wedge\; z - y \geq 2 \;\wedge\; z - x > 2 \;\wedge\; w - x = 2 \;\wedge\; z - w < 0$$

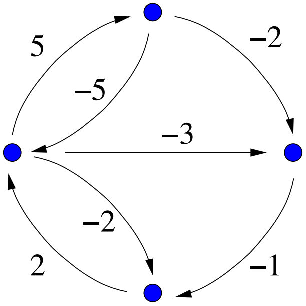
$$\begin{array}{lcl}
x - y = 5 & & x - y \leq 5 \;\wedge\; y - x \leq -5 \\
z - y \geq 2 & & y - z \leq -2 \\
z - x > 2 & \longrightarrow & x - z \leq -3 \\
w - x = 2 & & w - x \leq 2 \;\wedge\; x - w \leq -2 \\
z - w < 0 & & z - w \leq -1
\end{array}$$



Difference Logic Example

$$x - y = 5 \;\wedge\; z - y \geq 2 \;\wedge\; z - x > 2 \;\wedge\; w - x = 2 \;\wedge\; z - w < 0$$

$$\begin{array}{lcl} x - y = 5 & & x - y \leq 5 \;\wedge\; y - x \leq -5 \\ z - y \geq 2 & & y - z \leq -2 \\ z - x > 2 & \longrightarrow & x - z \leq -3 \\ w - x = 2 & & w - x \leq 2 \;\wedge\; x - w \leq -2 \\ z - w < 0 & & z - w \leq -1 \end{array}$$



Return UNSAT because of cycle: $-3, -1, 2$

Theory Solvers as Satisfiability Proof Systems

In general, how do we determine whether a conjunction (or, equivalently, a finite set) of literals is $\mathcal{T}$-satisfiable?

For many theories, we can use the framework of satisfiability proof systems

Theory Solvers as Satisfiability Proof Systems

In general, how do we determine whether a conjunction (or, equivalently, a finite set) of literals is $\mathcal{T}$-satisfiable?

For many theories, we can use the framework of satisfiability proof systems

Notation and Assumptions

A literal is *flat* if it is of the form:

$$x \doteq y \qquad \neg(x \doteq y) \qquad x \doteq f(\boldsymbol{z})$$

where x, y are variables, f is a function symbol and $\boldsymbol{z}$ is a tuple of 0 or more variables

Note: Any set of literals can be converted to an equisatisfiable flat set of literals by introducing fresh variables and equating non-equational atoms to true

Example

$$\{\, x + y > 0,\ y \doteq f(g(z)) \,\} \longrightarrow$$

$$\{\, v_1 \doteq \text{true},\ v_1 \doteq v_2 > v_3,\ v_2 \doteq x + y,\ v_3 \doteq 0,\ y \doteq f(v_4),\ v_4 \doteq g(z) \,\}$$

For the proof systems we present next, we assume that all literals are flat

Notation and Assumptions

A literal is *flat* if it is of the form:

$$x \doteq y \qquad \neg(x \doteq y) \qquad x \doteq f(\boldsymbol{z})$$

where x, y are variables, f is a function symbol and $\boldsymbol{z}$ is a tuple of 0 or more variables

Note: Any set of literals can be converted to an equisatisfiable flat set of literals by introducing fresh variables and equating non-equational atoms to true

Example

$\{\, x + y > 0,\ y \doteq f(g(z)) \,\} \longrightarrow$

$\{\, v_1 \doteq \mathrm{true},\ v_1 \doteq v_2 > v_3,\ v_2 \doteq x + y,\ v_3 \doteq 0,\ y \doteq f(v_4),\ v_4 \doteq g(z) \,\}$

For the proof systems we present next, we assume that all literals are flat

Notation and Assumptions

A literal is *flat* if it is of the form:

$$x \doteq y \qquad \neg(x \doteq y) \qquad x \doteq f(\boldsymbol{z})$$

where x, y are variables, f is a function symbol and $\boldsymbol{z}$ is a tuple of 0 or more variables

Note: Any set of literals can be converted to an equisatisfiable flat set of literals by introducing fresh variables and equating non-equational atoms to true

Example

$\{\, x + y > 0,\ y \doteq f(g(z)) \,\} \longrightarrow$

$\{\, v_1 \doteq \mathrm{true},\ v_1 \doteq v_2 > v_3,\ v_2 \doteq x + y,\ v_3 \doteq 0,\ y \doteq f(v_4),\ v_4 \doteq g(z) \,\}$

For the proof systems we present next, we assume that all literals are flat

Notation and Assumptions

A literal is *flat* if it is of the form:

$$x \doteq y \qquad \neg(x \doteq y) \qquad x \doteq f(\boldsymbol{z})$$

where x, y are variables, f is a function symbol and $\boldsymbol{z}$ is a tuple of 0 or more variables

Note: Any set of literals can be converted to an equisatisfiable flat set of literals by introducing fresh variables and equating non-equational atoms to true

Example

$$\{\, x + y > 0,\ y \doteq f(g(z)) \,\} \longrightarrow$$

$$\{\, v_1 \doteq \mathrm{true},\ v_1 \doteq v_2 > v_3,\ v_2 \doteq x + y,\ v_3 \doteq 0,\ y \doteq f(v_4),\ v_4 \doteq g(z) \,\}$$

For the proof systems we present next, we assume that all literals are flat

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \not\doteq t$

- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we write $\boldsymbol{u} = \boldsymbol{v}$ as an abbreviation for $u_1 \doteq v_1, \ldots, u_n \doteq v_n$

- Proof states, besides SAT and UNSAT, are sets Γ of formulas

- The satisfiable states are those that are $\mathcal{T}$-satisfiable, plus SAT

- We use Γ to refer to the current proof state in rule premises

- We write $\Gamma, s \doteq t$ as an abbreviation of $\Gamma \cup \{ s \doteq t \}$

- From now on, we also assume that if applying a rule R does not change Γ, then R is *not applicable* to Γ, i.e., Γ is irreducible with respect to R

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \not\doteq t$
- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we write $\boldsymbol{u} = \boldsymbol{v}$ as an abbreviation for $u_1 \doteq v_1, \ldots, u_n \doteq v_n$
- Proof states, besides SAT and UNSAT, are sets Γ of formulas
- The satisfiable states are those that are $\mathcal{T}$-satisfiable, plus SAT
- We use Γ to refer to the current proof state in rule premises
- We write $\Gamma, s \doteq t$ as an abbreviation of $\Gamma \cup \{s \doteq t\}$
- From now on, we also assume that if applying a rule R does not change Γ, then R is *not applicable* to Γ, i.e., Γ is irreducible with respect to R

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \not\doteq t$
- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we write $\boldsymbol{u} = \boldsymbol{v}$ as an abbreviation for $u_1 \doteq v_1, \ldots, u_n \doteq v_n$
- Proof states, besides $\textsf{SAT}$ and $\textsf{UNSAT}$, are sets Γ of formulas
- The satisfiable states are those that are $\mathcal{T}$-satisfiable, plus $\textsf{SAT}$
- We use Γ to refer to the current proof state in rule premises
- We write $\Gamma, s \doteq t$ as an abbreviation of $\Gamma \cup \{s \doteq t\}$
- From now on, we also assume that if applying a rule R does not change Γ, then R is *not applicable* to Γ, i.e., Γ is irreducible with respect to R

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \not\doteq t$
- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we write $\boldsymbol{u} = \boldsymbol{v}$ as an abbreviation for $u_1 \doteq v_1, \ldots, u_n \doteq v_n$
- Proof states, besides $\textsc{sat}$ and $\textsc{unsat}$, are sets Γ of formulas
- The satisfiable states are those that are $\mathcal{T}$-satisfiable, plus $\textsc{sat}$
- We use Γ to refer to the current proof state in rule premises
- We write $\Gamma, s \doteq t$ as an abbreviation of $\Gamma \cup \{s \doteq t\}$
- From now on, we also assume that if applying a rule R does not change Γ, then R is *not applicable* to Γ, i.e., Γ is irreducible with respect to R

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \not\doteq t$

- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we write $\boldsymbol{u} = \boldsymbol{v}$ as an abbreviation for $u_1 \doteq v_1, \ldots, u_n \doteq v_n$

- Proof states, besides SAT and UNSAT, are sets Γ of formulas

- The satisfiable states are those that are $\mathcal{T}$-satisfiable, plus SAT

- We use Γ to refer to the current proof state in rule premises

- We write $\Gamma, s \doteq t$ as an abbreviation of $\Gamma \cup \{ s \doteq t \}$

- From now on, we also assume that if applying a rule R does not change Γ, then R is *not applicable* to Γ, i.e., Γ is irreducible with respect to R

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \not\doteq t$

- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we write $\boldsymbol{u} = \boldsymbol{v}$ as an abbreviation for $u_1 \doteq v_1, \ldots, u_n \doteq v_n$

- Proof states, besides $\textsf{sat}$ and $\textsf{unsat}$, are sets Γ of formulas

- The satisfiable states are those that are $\mathcal{T}$-satisfiable, plus $\textsf{sat}$

- We use Γ to refer to the current proof state in rule premises

- We write $\Gamma, s \doteq t$ as an abbreviation of $\Gamma \cup \{ s \doteq t \}$

- From now on, we also assume that if applying a rule R does not change Γ, then R is *not applicable* to Γ, i.e., Γ is irreducible with respect to R

Notation and Assumptions

- We abbreviate $\neg(s \doteq t)$ with $s \mathrel{\dot{\neq}} t$
- For tuples $\boldsymbol{u} = \langle u_1, \ldots, u_n \rangle$ and $\boldsymbol{v} = \langle v_1, \ldots, v_n \rangle$, we write $\boldsymbol{u} = \boldsymbol{v}$ as an abbreviation for $u_1 \doteq v_1, \ldots, u_n \doteq v_n$
- Proof states, besides SAT and UNSAT, are sets Γ of formulas
- The satisfiable states are those that are $\mathcal{T}$-satisfiable, plus SAT
- We use Γ to refer to the current proof state in rule premises
- We write $\Gamma, s \doteq t$ as an abbreviation of $\Gamma \cup \{ s \doteq t \}$
- From now on, we also assume that if applying a rule R does not change Γ, then R is *not applicable* to Γ, i.e., Γ is irreducible with respect to R

A Satisfiability Proof System for QF_UF

Let QF_UF be the quantifier-free fragment of FOL over some signature Σ

The following is a simple satisfiability proof system R_{UF} for QF_UF:

$$\textsc{Contr}\ \frac{x \doteq y \in \Gamma \quad x \not\doteq y \in \Gamma}{\text{UNSAT}} \qquad \textsc{Refl}\ \frac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x}$$

$$\textsc{Symm}\ \frac{x \doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x} \qquad \textsc{Trans}\ \frac{x \doteq y \in \Gamma \quad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z}$$

$$\textsc{Cong}\ \frac{\begin{array}{c} x \doteq f(\mathbf{u}) \in \Gamma \\ y \doteq f(\mathbf{v}) \in \Gamma \quad \mathbf{u} \doteq \mathbf{v} \in \Gamma \end{array}}{\Gamma := \Gamma, x \doteq y} \qquad \textsc{Sat}\ \frac{\text{No other rules apply}}{\text{SAT}}$$

Is R_{UF} sound? Is it terminating?

A Satisfiability Proof System for QF_UF

Let QF_UF be the quantifier-free fragment of FOL over some signature Σ

The following is a simple satisfiability proof system R_{UF} for QF_UF:

$$\textsc{Contr}\ \frac{x \doteq y \in \Gamma \quad x \not\doteq y \in \Gamma}{\text{UNSAT}} \qquad \textsc{Refl}\ \frac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x}$$

$$\textsc{Symm}\ \frac{x \doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x} \qquad \textsc{Trans}\ \frac{x \doteq y \in \Gamma \quad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z}$$

$$\textsc{Cong}\ \frac{\begin{array}{l} x \doteq f(\boldsymbol{u}) \in \Gamma \\ y \doteq f(\boldsymbol{v}) \in \Gamma \quad \boldsymbol{u} \doteq \boldsymbol{v} \in \Gamma \end{array}}{\Gamma := \Gamma, x \doteq y} \qquad \textsc{Sat}\ \frac{\text{No other rules apply}}{\text{SAT}}$$

Is R_{UF} sound? Is it terminating?

A Satisfiability Proof System for QF_UF

Let QF_UF be the quantifier-free fragment of FOL over some signature Σ

The following is a simple satisfiability proof system R_{UF} for QF_UF:

$$\textsc{Contr}\ \frac{x \doteq y \in \Gamma \quad x \not\doteq y \in \Gamma}{\text{UNSAT}} \qquad\qquad \textsc{Refl}\ \frac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x}$$

$$\textsc{Symm}\ \frac{x \doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x} \qquad\qquad \textsc{Trans}\ \frac{x \doteq y \in \Gamma \quad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z}$$

$$\textsc{Cong}\ \frac{\begin{array}{l} x \doteq f(\boldsymbol{u}) \in \Gamma \\ y \doteq f(\boldsymbol{v}) \in \Gamma \quad \boldsymbol{u} \doteq \boldsymbol{v} \in \Gamma \end{array}}{\Gamma := \Gamma, x \doteq y} \qquad\qquad \textsc{Sat}\ \frac{\text{No other rules apply}}{\text{SAT}}$$

Is R_{UF} sound? Is it terminating?

A Satisfiability Proof System for QF_UF

Let QF_UF be the quantifier-free fragment of FOL over some signature Σ

The following is a simple satisfiability proof system R_{UF} for QF_UF:

$$\textsc{Contr}\ \frac{x \doteq y \in \Gamma \qquad x \not\doteq y \in \Gamma}{\text{UNSAT}}$$

$$\textsc{Refl}\ \frac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x}$$

$$\textsc{Symm}\ \frac{x \doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x}$$

$$\textsc{Trans}\ \frac{x \doteq y \in \Gamma \qquad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z}$$

$$\textsc{Cong}\ \frac{\begin{array}{l} x \doteq f(\boldsymbol{u}) \in \Gamma \\ y \doteq f(\boldsymbol{v}) \in \Gamma \qquad \boldsymbol{u} \doteq \boldsymbol{v} \in \Gamma \end{array}}{\Gamma := \Gamma, x \doteq y}$$

$$\textsc{Sat}\ \frac{\text{No other rules apply}}{\text{SAT}}$$

Is R_{UF} sound? Is it terminating?

Example derivation

$$\textsc{Refl}\ \dfrac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x} \qquad \textsc{Contr}\ \dfrac{x \doteq y \in \Gamma \quad x \not\doteq y \in \Gamma}{\text{UNSAT}} \qquad \textsc{Trans}\ \dfrac{x \doteq y \in \Gamma \quad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z}$$

$$\textsc{Symm}\ \dfrac{x \doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x} \qquad \textsc{Cong}\ \dfrac{\begin{array}{l} x \doteq f(\boldsymbol{u}) \in \Gamma \\ y \doteq f(\boldsymbol{v}) \in \Gamma \quad \boldsymbol{u} \doteq \boldsymbol{v} \in \Gamma \end{array}}{\Gamma := \Gamma, x \doteq y} \qquad \textsc{Sat}\ \dfrac{\text{No other rules apply}}{\text{SAT}}$$

Problem Determine the satisfiability of $\{\, a \doteq f(f(a)),\ a \doteq f(f(f(a))),\ g(a, f(a)) \not\doteq g(f(a), a) \,\}$ which can be flattened to

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{a \doteq f(a_1),\ a_1 \doteq f(a),\ a \doteq f(a_2),\ a_2 \doteq f(a_1),\ a_3 \not\doteq a_4,\ a_3 \doteq g(a, a_1),\ a_4 \doteq g(a_1, a)}{a_1 \doteq a_1}\,(\textsc{Refl})}{a \doteq a_2}\,(\textsc{Cong}^1)}{a_1 \doteq a}\,(\textsc{Cong}^2)}{a \doteq a_1}\,(\textsc{Symm})}{a_3 \doteq a_4}\,(\textsc{Cong}^3)}{\text{UNSAT}}\,(\textsc{Contr}^4)$$

Showing only difference with previous state

[1] applied to $a \doteq f(a_1),\ a_2 \doteq f(a_1),\ a_1 \doteq a_1$

[2] applied to $a_1 \doteq f(a),\ a \doteq f(a_2),\ a \doteq a_2$

[3] applied to $a_3 \doteq g(a, a_1),\ a_4 \doteq g(a_1, a),\ a \doteq a_1,\ a_1 \doteq a$

[4] applied to $a_3 \doteq a_4,\ a_3 \not\doteq a_4$

Example derivation

$$\textsc{Refl}\ \frac{x \text{ occurs in } \Gamma}{\Gamma := \Gamma, x \doteq x} \qquad \textsc{Contr}\ \frac{x \doteq y \in \Gamma \quad x \not\doteq y \in \Gamma}{\text{UNSAT}} \qquad \textsc{Trans}\ \frac{x \doteq y \in \Gamma \quad y \doteq z \in \Gamma}{\Gamma := \Gamma, x \doteq z}$$

$$\textsc{Symm}\ \frac{x \doteq y \in \Gamma}{\Gamma := \Gamma, y \doteq x} \qquad \textsc{Cong}\ \frac{\begin{matrix} x \doteq f(\boldsymbol{u}) \in \Gamma \\ y \doteq f(\boldsymbol{v}) \in \Gamma \quad \boldsymbol{u} \doteq \boldsymbol{v} \in \Gamma \end{matrix}}{\Gamma := \Gamma, x \doteq y} \qquad \textsc{Sat}\ \frac{\text{No other rules apply}}{\text{SAT}}$$

Problem Determine the satisfiability of $\{\, a \doteq f(f(a)),\ a \doteq f(f(f(a))),\ g(a, f(a)) \not\doteq g(f(a), a) \,\}$ which can be flattened to

$$\dfrac{a \doteq f(a_1),\ a_1 \doteq f(a),\ a \doteq f(a_2),\ a_2 \doteq f(a_1),\ a_3 \not\doteq a_4,\ a_3 \doteq g(a, a_1),\ a_4 \doteq g(a_1, a)}{\dfrac{a_1 \doteq a_1}{\dfrac{a \doteq a_2}{\dfrac{a_1 \doteq a}{\dfrac{a \doteq a_1}{\dfrac{a_3 \doteq a_4}{\text{UNSAT}}\ (\textsc{Contr}^{4})}\ (\textsc{Cong}^{3})}\ (\textsc{Symm})}\ (\textsc{Cong}^{2})}\ (\textsc{Cong}^{1})}\ (\textsc{Refl})$$

Showing only difference with previous state

[1] applied to $a \doteq f(a_1),\ a_2 \doteq f(a_1),\ a_1 \doteq a_1$

[2] applied to $a_1 \doteq f(a),\ a \doteq f(a_2),\ a \doteq a_2$

[3] applied to $a_3 \doteq g(a, a_1),\ a_4 \doteq g(a_1, a),\ a \doteq a_1,\ a_1 \doteq a$

[4] applied to $a_3 \doteq a_4,\ a_3 \not\doteq a_4$

Soundness

Theorem 1 (Refutation soundness)

A literal set Γ_0 is unsatisfiable if R_{UF} derives $\textsc{unsat}$ from it.

Soundness

Theorem 1 (Refutation soundness)

A literal set Γ_0 is unsatisfiable if R_{UF} derives UNSAT *from it.*

Proof sketch. All rules but **SAT** are clearly satisfiability preserving.

If a derivation from Γ_0 ends with UNSAT, it must then be that Γ_0 is unsatisfiable. □

Soundness

Theorem 1 (Solutions soundness)

A literal set Γ_0 is satisfiable if R_{UF} derives $\textsc{sat}$ *from it.*

Soundness

Theorem 1 (Solutions soundness)

A literal set Γ_0 is satisfiable if R_{UF} derives $\textsc{sat}$ from it.

Proof sketch. Let Γ be a proof state to which **SAT** applies. From Γ, we construct an interpretation that satisfies Γ_0.

Let $s \sim t$ iff $s = t \in \Gamma$. One can show that $\sim$ is an equivalence relation.

Let the domain of $\mathcal{I}$ be the equivalence classes $E_1, \ldots, E_k$ of $\sim$.

For every variable or a constant t, let $t^{\mathcal{I}} = E_i$ if $t \in E_i$ for some i; otherwise, let $t^{\mathcal{I}} = E_1$.

For every unary function symbol f, and equivalence class E_i, let $f^{\mathcal{I}}$ be such that $f^{\mathcal{I}}(E_i) = E_j$ if $f(t) \in E_j$ for some $t \in E_i$, and $f^{\mathcal{I}}(E_i) = E_1$ otherwise. Define $f^{\mathcal{I}}$ for non-unary f similarly.

We can show that $\mathcal{I} \models \Gamma$. This means that $\mathcal{I} \models \Gamma_0$ as well since $\Gamma_0 \subseteq \Gamma$. □

Termination

Theorem 2 (Termination)

Every derivation strategy for R_{UF} terminates.

Termination

Theorem 2 (Termination)

Every derivation strategy for R_{UF} terminates.

Proof sketch. R_{UF} adds to the current state Γ only equalities between variables of Γ_0.

So at some point it will run out of new equalities to add. □

Completeness

Theorem 3 (Refutation completeness)

Every derivation strategy applied to an unsatisfiable state Γ_0 *ends with* UNSAT.

Completeness

Theorem 3 (Refutation completeness)

Every derivation strategy applied to an unsatisfiable state Γ_0 ends with UNSAT.

Proof sketch. Let Γ_0 be an unsatisfiable state.

Suppose there was a derivation from Γ_0 that did not end with UNSAT.

Then, by the termination theorem, it would have to end with SAT.

But then R_{UF} would be not be solution sound. □

Completeness

Theorem 3 (Refutation completeness)

Every derivation strategy applied to an unsatisfiable state Γ_0 ends with $\textsc{unsat}$.

Theorem 4 (Solution completeness)

Every derivation strategy applied to a satisfiable state Γ_0 ends with $\textsc{sat}$.

Completeness

Theorem 3 (Refutation completeness)

Every derivation strategy applied to an unsatisfiable state Γ_0 ends with UNSAT.

Theorem 4 (Solution completeness)

Every derivation strategy applied to a satisfiable state Γ_0 ends with SAT.

Proof sketch. Let Γ_0 be a satisfiable state.

Suppose there was a derivation from Γ_0 that did not end with SAT.

Then, by the termination theorem, it would have to end with UNSAT.

But then R_{UF} would be refutation unsound. □

Theory of Arrays $\mathcal{T}_A$

Recall: $\mathcal{T}_A = \langle \Sigma, \boldsymbol{M} \rangle$ where

- $\Sigma^S = \{\, A, I, E \,\}$ (for arrays, indices, elements)

  $\Sigma^F = \{\, \text{read}, \text{write} \,\}$, $\quad \mathrm{rank}(\text{read}) = \langle A, I, E \rangle$ and $\mathrm{rank}(\text{write}) = \langle A, I, E, A \rangle$

- $\boldsymbol{M}$ is the class of Σ-interpretations that satisfy the following axioms:
  1. $\forall a.\, \forall i.\, \forall v.\, \text{read}(\text{write}(a, i, v), i) \doteq v$
  2. $\forall a.\, \forall i.\, \forall i'.\, \forall v.\, (i \not\doteq i' \Rightarrow \text{read}(\text{write}(a, i, v), i') \doteq \mathit{read}(a, i'))$
  3. $\forall a.\, \forall a'_1.\, (\forall i.\text{read}(a, i) \doteq \text{read}(a'_1, i) \Rightarrow a \doteq a'_1)$

Example

```
void ReadBlock(int data[], int x, int len)
{
 int i = 0;
 int next = data[0];
 for (; i < next && i < len; i = i + 1) {
  if (data[i] == x)
   break;
  else
   Process(data[i]);
 }
 assert(i < len);
}
```

One path through this code can be translated using the theory of arrays as:

$$i \doteq 0 \ \wedge \ \mathit{next} \doteq \mathrm{read}(\mathit{data}, 0) \ \wedge \ i < \mathit{next} \wedge \\ i < \mathit{len} \ \wedge \ \mathrm{read}(\mathit{data}, i) = x \ \wedge \ \neg(i < \mathit{len})$$

A Satisfiability Proof System for $\mathcal{T}_A$

The satisfiability proof system R_A for $\mathcal{T}_A$ extends the proof system R_{UF} for QF_UF with the following rules:

$$\textsc{RIntro1}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b, i)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma \quad u \doteq \mathrm{read}(c, j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a, j),\ u \doteq \mathrm{read}(b, j)}$$

$$\textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a, k),\ v \doteq \mathrm{read}(b, k)}$$

where e_1, e_2 and k are fresh variables

A Satisfiability Proof System for $\mathcal{T}_A$

The satisfiability proof system R_A for $\mathcal{T}_A$ extends the proof system R_{UF} for QF_UF with the following rules:

$$\textsc{RIntro1}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b, i)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma \quad u \doteq \mathrm{read}(c, j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a, j),\ u \doteq \mathrm{read}(b, j)}$$

$$\textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a, k),\ v \doteq \mathrm{read}(b, k)}$$

where e_1, e_2 and k are fresh variables

RIntro1: If b results from writing v in a at position i, then reading b at that position gives you v

A Satisfiability Proof System for $\mathcal{T}_A$

The satisfiability proof system R_A for $\mathcal{T}_A$ extends the proof system R_{UF} for QF_UF with the following rules:

$$\textbf{RIntro1}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b, i)}$$

$$\textbf{RIntro2}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma \quad u \doteq \mathrm{read}(c, j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a, j),\ u \doteq \mathrm{read}(b, j)}$$

$$\textbf{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a, k),\ v \doteq \mathrm{read}(b, k)}$$

where e_1, e_2 and k are fresh variables

RIntro2: If b results from writing v in a at position i, and a or b is read at position j, then separately consider two cases: (1) i equals j; (2) a and b have the same value at position j

A Satisfiability Proof System for $\mathcal{T}_A$

The satisfiability proof system R_A for $\mathcal{T}_A$ extends the proof system R_{UF} for QF_UF with the following rules:

$$\textsc{RIntro1}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b, i)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma \quad u \doteq \mathrm{read}(c, j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a, j),\ u \doteq \mathrm{read}(b, j)}$$

$$\textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a, k),\ v \doteq \mathrm{read}(b, k)}$$

where e_1, e_2 and k are fresh variables

Ext: If arrays a_1 and a_2 are distinct, they must differ in the value they store at some position k

Example

$$\textsc{RIntro1}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b,i)} \qquad \textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a,b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a,k),\ v \doteq \mathrm{read}(b,k)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma \quad u \doteq \mathrm{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a,b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a,j),\ u \doteq \mathrm{read}(b,j)}$$

Example

$$\textsc{RIntro1}\ \frac{b \doteq \mathsf{write}(a, i, v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathsf{read}(b, i)} \qquad \textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathsf{read}(a, k),\ v \doteq \mathsf{read}(b, k)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathsf{write}(a, i, v) \in \Gamma \quad u \doteq \mathsf{read}(c, j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathsf{read}(a, j),\ u \doteq \mathsf{read}(b, j)}$$

Determine the satisfiability of $\{\, \mathsf{write}(a_1, i, \mathsf{read}(a_2, i)) \doteq \mathsf{write}(a_2, i, \mathsf{read}(a_1, i)),\ a_1 \not\doteq a_2 \,\}$

Example

$$\textsc{RIntro1}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b,i)} \qquad \textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a,b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a,k),\ v \doteq \mathrm{read}(b,k)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma \quad u \doteq \mathrm{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a,b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a,j),\ u \doteq \mathrm{read}(b,j)}$$

Determine the satisfiability of $\{\, \mathrm{write}(a_1, i, \mathrm{read}(a_2, i)) \doteq \mathrm{write}(a_2, i, \mathrm{read}(a_1, i)),\ a_1 \not\doteq a_2 \,\}$

First, we convert the problem to flat form:

$$\{\, \mathrm{write}(a_1, i, \mathrm{read}(a_2, i)) \doteq \mathrm{write}(a_2, i, \mathrm{read}(a_1, i)),\ a_1 \not\doteq a_2 \,\}$$

$$\longrightarrow \{\, a_1' \doteq a_2',\ a_1' \doteq \mathrm{write}(a_1, i, \mathrm{read}(a_2, i)),\ a_2' \doteq \mathrm{write}(a_2, i, \mathrm{read}(a_1, i)),\ a_1 \not\doteq a_2 \,\}$$

$$\longrightarrow \{\, a_1' \doteq a_2',\ a_1' \doteq \mathrm{write}(a_1, i, v_2),\ v_2 \doteq \mathrm{read}(a_2, i),\ a_2' \doteq \mathrm{write}(a_2, i, v_1),\ v_1 \doteq \mathrm{read}(a_1, i),\ a_1 \not\doteq a_2 \,\}$$

Example

$$\textsc{RIntro1}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b,i)} \qquad \textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a,k),\ v \doteq \mathrm{read}(b,k)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma \quad u \doteq \mathrm{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a,j),\ u \doteq \mathrm{read}(b,j)}$$

$$\cfrac{a'_1 \doteq a'_2,\ a'_1 \doteq \mathrm{write}(a_1,i,v_2),\ v_2 \doteq \mathrm{read}(a_2,i),\ a'_2 \doteq \mathrm{write}(a_2,i,v_1),\ v_1 \doteq \mathrm{read}(a_1,i),\ a_1 \not\doteq a_2}{a_1 \doteq a_1}\ (\textsc{Refl})$$

$$\frac{a_1 \doteq a_1}{a_2 \doteq a_2}\ (\textsc{Refl}) \qquad \frac{a_2 \doteq a_2}{u_1 \not\doteq u_2,\ u_1 \doteq \mathrm{read}(a_1,n),\ u_2 \doteq \mathrm{read}(a_2,n)}\ (\textsc{Ext}^1)$$

$$\frac{u_1 \not\doteq u_2,\ u_1 \doteq \mathrm{read}(a_1,n),\ u_2 \doteq \mathrm{read}(a_2,n)}{i \doteq n \qquad\qquad i \not\doteq n, u_1 \doteq \mathrm{read}(a'_1,n)}\ (\textsc{RIntro2}^2)$$

$$\frac{i \doteq n}{v_1 \doteq u_1}\ (\textsc{Cong}^3) \quad \frac{v_1 \doteq u_1}{u_1 \doteq v_1}\ (\textsc{Symm}) \quad \frac{u_1 \doteq v_1}{v_2 \doteq u_2}\ (\textsc{Cong}^4) \quad \frac{v_2 \doteq u_2}{v_2 \doteq \mathrm{read}(a'_1,i)}\ (\textsc{RIntro1}^5)$$

$$\frac{v_2 \doteq \mathrm{read}(a'_1,i)}{v_1 \doteq \mathrm{read}(a'_2,i)}\ (\textsc{RIntro1}^6) \quad \frac{v_1 \doteq \mathrm{read}(a'_2,i)}{i \doteq i}\ (\textsc{Refl}) \quad \frac{i \doteq i}{v_1 \doteq v_2}\ (\textsc{Cong}^7) \quad \frac{v_1 \doteq v_2}{u_1 \doteq u_2}\ (\textsc{Trans}^8) \quad \frac{u_1 \doteq u_2}{\mathrm{UNSAT}}\ (\textsc{Contr})$$

$$\frac{i \not\doteq n, u_1 \doteq \mathrm{read}(a'_1,n)}{i \doteq n \qquad\qquad i \not\doteq n,\ u_2 \doteq \mathrm{read}(a'_2,n)}\ (\textsc{RIntro2}^9)$$

$$\frac{i \doteq n}{\mathrm{UNSAT}}\ (\textsc{Contr})$$

$$\frac{i \not\doteq n,\ u_2 \doteq \mathrm{read}(a'_2,n)}{n \doteq n}\ (\textsc{Refl}) \quad \frac{n \doteq n}{u_1 \doteq u_2}\ (\textsc{Cong}^{10}) \quad \frac{u_1 \doteq u_2}{\mathrm{UNSAT}}\ (\textsc{Contr})$$

Showing only difference with previous state

[1] applied to $a_1 \not\doteq a_2$ [2] applied to $a'_1 \doteq \mathrm{write}(a_1,i,v_2), u_1 \doteq \mathrm{read}(a_1,n), a_1 \doteq a_1$

Example

$$\text{RIntro1}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b,i)} \qquad \text{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a,b\ \text{arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a,k),\ v \doteq \mathrm{read}(b,k)}$$

$$\text{RIntro2}\ \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma \quad u \doteq \mathrm{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{a,b\}}{\Gamma := \Gamma,\ i \doteq j \qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a,j),\ u \doteq \mathrm{read}(b,j)}$$

$$\begin{array}{cl}
a'_1 \doteq a'_2,\ a'_1 \doteq \mathrm{write}(a_1,i,v_2),\ v_2 \doteq \mathrm{read}(a_2,i),\ a'_2 \doteq \mathrm{write}(a_2,i,v_1),\ v_1 \doteq \mathrm{read}(a_1,i),\ a_1 \not\doteq a_2 & \\
\hline
a_1 \doteq a_1 & (\text{Refl}) \\
\hline
a_2 \doteq a_2 & (\text{Refl}) \\
\hline
u_1 \not\doteq u_2,\ u_1 \doteq \mathrm{read}(a_1,n),\ u_2 \doteq \mathrm{read}(a_2,n) & (\text{Ext}^{1}) \\
\hline
 & (\text{RIntro2}^{2})
\end{array}$$

$$\begin{array}{cl}
i \doteq n & \\
\hline
v_1 \doteq u_1 & (\text{Cong}^{3}) \\
\hline
u_1 \doteq v_1 & (\text{Symm}) \\
\hline
v_2 \doteq u_2 & (\text{Cong}^{4}) \\
\hline
v_2 \doteq \mathrm{read}(a'_1,i) & (\text{RIntro1}^{5}) \\
\hline
v_1 \doteq \mathrm{read}(a'_2,i) & (\text{RIntro1}^{6}) \\
\hline
i \doteq i & (\text{Refl}) \\
\hline
v_1 \doteq v_2 & (\text{Cong}^{7}) \\
\hline
u_1 \doteq u_2 & (\text{Trans}^{8}) \\
\hline
\text{UNSAT} & (\text{Contr})
\end{array}
\qquad
\begin{array}{cl}
i \not\doteq n,\ u_1 \doteq \mathrm{read}(a'_1,n) & \\
\hline
 & (\text{RIntro2}^{9})
\end{array}$$

$$\begin{array}{cl}
i \doteq n & \\
\hline
\text{UNSAT} & (\text{Contr})
\end{array}
\qquad
\begin{array}{cl}
i \not\doteq n,\ u_2 \doteq \mathrm{read}(a'_2,n) & \\
\hline
n \doteq n & (\text{Refl}) \\
\hline
u_1 \doteq u_2 & (\text{Cong}^{10}) \\
\hline
\text{UNSAT} & (\text{Contr})
\end{array}$$

Showing only difference with previous state

[3] applied to $v_1 \doteq \mathrm{read}(a_1,i), u_1 \doteq \mathrm{read}(a_1,n), a_1 \doteq a_1, i \doteq n$

[4] appl. to $v_2 \doteq \mathrm{read}(a_2,i), u_2 \doteq \mathrm{read}(a_2,n), a_2 \doteq a_2, i \doteq n$

Example

$$\textsc{RIntro1}\ \frac{b \doteq \mathsf{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathsf{read}(b,i)} \qquad \textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathsf{read}(a,k),\ v \doteq \mathsf{read}(b,k)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathsf{write}(a,i,v) \in \Gamma \quad u \doteq \mathsf{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{a,b\}}{\Gamma := \Gamma,\ i \doteq j \qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathsf{read}(a,j),\ u \doteq \mathsf{read}(b,j)}$$

$$\dfrac{a'_1 \doteq a'_2,\ a'_1 \doteq \mathsf{write}(a_1,i,v_2),\ v_2 \doteq \mathsf{read}(a_2,i),\ a'_2 \doteq \mathsf{write}(a_2,i,v_1),\ v_1 \doteq \mathsf{read}(a_1,i),\ a_1 \not\doteq a_2}{\dfrac{a_1 \doteq a_1}{\dfrac{a_2 \doteq a_2}{\dfrac{u_1 \not\doteq u_2,\ u_1 \doteq \mathsf{read}(a_1,n),\ u_2 \doteq \mathsf{read}(a_2,n)}{\dfrac{i \doteq n}{\dfrac{v_1 \doteq u_1}{\dfrac{u_1 \doteq v_1}{\dfrac{v_2 \doteq u_2}{\dfrac{v_2 \doteq \mathsf{read}(a'_1,i)}{\dfrac{v_1 \doteq \mathsf{read}(a'_2,i)}{\dfrac{i \doteq i}{\dfrac{v_1 \doteq v_2}{\dfrac{u_1 \doteq u_2}{\text{UNSAT}}\,(\textsc{Contr})}\,(\textsc{Trans}^{8})}\,(\textsc{Cong}^{7})}\,(\textsc{Refl})}\,(\textsc{RIntro1}^{6})}\,(\textsc{RIntro1}^{5})}\,(\textsc{Cong}^{4})}\,(\textsc{Symm})}\,(\textsc{Cong}^{3}) \qquad \dfrac{i \not\doteq n, u_1 \doteq \mathsf{read}(a'_1,n)}{\dfrac{i \doteq n}{\text{UNSAT}}\,(\textsc{Contr}) \qquad \dfrac{i \not\doteq n,\ u_2 \doteq \mathsf{read}(a'_2,n)}{\dfrac{n \doteq n}{\dfrac{u_1 \doteq u_2}{\text{UNSAT}}\,(\textsc{Contr})}\,(\textsc{Cong}^{10})}\,(\textsc{Refl})}\,(\textsc{RIntro2}^{9})}\,(\textsc{RIntro2}^{2})}\,(\textsc{Ext}^{1})}\,(\textsc{Refl})}\,(\textsc{Refl})$$

Showing only difference with previous state

[5] applied to $a'_1 \doteq \mathsf{write}(a_1,i,v_2)$ [6] applied to $a'_2 \doteq \mathsf{write}(a_2,i,v_1)$

Example

$$\textsc{RIntro1}\ \frac{b \doteq \mathsf{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathsf{read}(b,i)} \qquad \textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a,b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathsf{read}(a,k),\ v \doteq \mathsf{read}(b,k)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \mathsf{write}(a,i,v) \in \Gamma \quad u \doteq \mathsf{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{a,b\}}{\Gamma := \Gamma,\ i \doteq j \qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathsf{read}(a,j),\ u \doteq \mathsf{read}(b,j)}$$

$$\begin{array}{cl}
a_1' \doteq a_2',\ a_1' \doteq \mathsf{write}(a_1,i,v_2),\ v_2 \doteq \mathsf{read}(a_2,i),\ a_2' \doteq \mathsf{write}(a_2,i,v_1),\ v_1 \doteq \mathsf{read}(a_1,i),\ a_1 \not\doteq a_2 & \\ \hline
a_1 \doteq a_1 & (\textsc{Refl}) \\ \hline
a_2 \doteq a_2 & (\textsc{Refl}) \\ \hline
u_1 \not\doteq u_2,\ u_1 \doteq \mathsf{read}(a_1,n),\ u_2 \doteq \mathsf{read}(a_2,n) & (\textsc{Ext}^{1}) \\ \hline
\begin{array}{cc}
\begin{array}{cl}
i \doteq n & (\textsc{RIntro2}^{2}) \\ \hline
v_1 \doteq u_1 & (\textsc{Cong}^{3}) \\ \hline
u_1 \doteq v_1 & (\textsc{Symm}) \\ \hline
v_2 \doteq u_2 & (\textsc{Cong}^{4}) \\ \hline
v_2 \doteq \mathsf{read}(a_1',i) & (\textsc{RIntro1}^{5}) \\ \hline
v_1 \doteq \mathsf{read}(a_2',i) & (\textsc{RIntro1}^{6}) \\ \hline
i \doteq i & (\textsc{Refl}) \\ \hline
v_1 \doteq v_2 & (\textsc{Cong}^{7}) \\ \hline
u_1 \doteq u_2 & (\textsc{Trans}^{8}) \\ \hline
\mathsf{UNSAT} & (\textsc{Contr})
\end{array}
&
\begin{array}{c}
i \not\doteq n, u_1 \doteq \mathsf{read}(a_1',n) \\ \hline
\begin{array}{cc}
\begin{array}{cl}
i \doteq n & (\textsc{RIntro2}^{9}) \\ \hline
\mathsf{UNSAT} & (\textsc{Contr})
\end{array}
&
\begin{array}{cl}
i \not\doteq n,\ u_2 \doteq \mathsf{read}(a_2',n) & \\ \hline
n \doteq n & (\textsc{Refl}) \\ \hline
u_1 \doteq u_2 & (\textsc{Cong}^{10}) \\ \hline
\mathsf{UNSAT} & (\textsc{Contr})
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}$$

Showing only difference with previous state

[7] applied to $v_1 \doteq \mathsf{read}(a_2',i)$, $v_2 \doteq \mathsf{read}(a_1',i)$, $a_1' \doteq a_2'$, $i \doteq i$ [8] applied to $u_1 \doteq v_1$, $v_1 \doteq v_2$, $v_2 \doteq u_2$

Example

$$\textsc{RIntro1}\ \dfrac{b \doteq \mathsf{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathsf{read}(b,i)} \qquad \textsc{Ext}\ \dfrac{a \not\doteq b \in \Gamma \quad a,b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathsf{read}(a,k),\ v \doteq \mathsf{read}(b,k)}$$

$$\textsc{RIntro2}\ \dfrac{b \doteq \mathsf{write}(a,i,v) \in \Gamma \quad u \doteq \mathsf{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a,b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathsf{read}(a,j),\ u \doteq \mathsf{read}(b,j)}$$

$$a'_1 \doteq a'_2,\ a'_1 \doteq \mathsf{write}(a_1,i,v_2),\ v_2 \doteq \mathsf{read}(a_2,i),\ a'_2 \doteq \mathsf{write}(a_2,i,v_1),\ v_1 \doteq \mathsf{read}(a_1,i),\ a_1 \not\doteq a_2$$

(Refl)

$$a_1 \doteq a_1$$

(Refl)

$$a_2 \doteq a_2$$

(Ext[1])

$$u_1 \not\doteq u_2,\ u_1 \doteq \mathsf{read}(a_1,n),\ u_2 \doteq \mathsf{read}(a_2,n)$$

(RIntro2[2])

$$i \doteq n$$

(Cong[3])

$$v_1 \doteq u_1$$

(Symm)

$$u_1 \doteq v_1$$

(Cong[4])

$$v_2 \doteq u_2$$

(RIntro1[5])

$$v_2 \doteq \mathsf{read}(a'_1,i)$$

(RIntro1[6])

$$v_1 \doteq \mathsf{read}(a'_2,i)$$

(Refl)

$$i \doteq i$$

(Cong[7])

$$v_1 \doteq v_2$$

(Trans[8])

$$u_1 \doteq u_2$$

(Contr)

UNSAT

$$i \not\doteq n, u_1 \doteq \mathsf{read}(a'_1,n)$$

(RIntro2[9])

$$i \doteq n$$

(Contr)

UNSAT

$$i \not\doteq n,\ u_2 \doteq \mathsf{read}(a'_2,n)$$

(Refl)

$$n \doteq n$$

(Cong[10])

$$u_1 \doteq u_2$$

(Contr)

UNSAT

Showing only difference with previous state

[9] applied to $a'_2 \doteq \mathsf{write}(a_2,i,v_1), u_2 \doteq \mathsf{read}(a_2,n), a_2 \doteq a_2$ [10] appl. to $u_1 \doteq \mathsf{read}(a'_1,n), u_2 \doteq \mathsf{read}(a'_2,n), a'_1 \doteq a'_2, n \doteq n$

A Satisfiability Proof System for $\mathcal{T}_A$

The satisfiability proof system R_A for $\mathcal{T}_A$ extends the proof system R_{UF} for QF_UF with the following rules:

$$\textbf{RIntro1}\quad \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b,i)}$$

$$\textbf{RIntro2}\quad \frac{b \doteq \mathrm{write}(a,i,v) \in \Gamma \quad u \doteq \mathrm{read}(c,j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a,b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a,j),\ u \doteq \mathrm{read}(b,j)}$$

$$\textbf{Ext}\quad \frac{a \not\doteq b \in \Gamma \quad a,b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a,k),\ v \doteq \mathrm{read}(b,k)}$$

Is R_A sound? Is it terminating?

A Satisfiability Proof System for $\mathcal{T}_A$

The satisfiability proof system R_A for $\mathcal{T}_A$ extends the proof system R_{UF} for QF_UF with the following rules:

$$\textbf{RIntro1}\quad \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \mathrm{read}(b, i)}$$

$$\textbf{RIntro2}\quad \frac{b \doteq \mathrm{write}(a, i, v) \in \Gamma \quad u \doteq \mathrm{read}(c, j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad\qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \mathrm{read}(a, j),\ u \doteq \mathrm{read}(b, j)}$$

$$\textbf{Ext}\quad \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \mathrm{read}(a, k),\ v \doteq \mathrm{read}(b, k)}$$

Is R_A sound? Is it terminating?

A Satisfiability Proof System for $\mathcal{T}_A$

The satisfiability proof system R_A for $\mathcal{T}_A$ extends the proof system R_{UF} for QF_UF with the following rules:

$$\textsc{RIntro1}\ \frac{b \doteq \text{write}(a, i, v) \in \Gamma}{\Gamma := \Gamma,\ v \doteq \text{read}(b, i)}$$

$$\textsc{RIntro2}\ \frac{b \doteq \text{write}(a, i, v) \in \Gamma \quad u \doteq \text{read}(c, j) \in \Gamma \quad x \doteq c \in \Gamma \quad x \in \{\, a, b \,\}}{\Gamma := \Gamma,\ i \doteq j \qquad \Gamma := \Gamma,\ i \not\doteq j,\ u \doteq \text{read}(a, j),\ u \doteq \text{read}(b, j)}$$

$$\textsc{Ext}\ \frac{a \not\doteq b \in \Gamma \quad a, b \text{ arrays}}{\Gamma := \Gamma,\ u \not\doteq v,\ u \doteq \text{read}(a, k),\ v \doteq \text{read}(b, k)}$$

Is R_A sound? Is it terminating?

Soundness, Termination, and Completeness

Refutation soundness is straightforward and follows from the $\mathcal{T}_A$ axioms.

Termination follows from the following argument. Once we add all of the $i_{a,b}$ variables, no rule introduces new variables. There are only a finite number of terms that match the conclusions that can be constructed with a finite number of variables, so eventually, Γ will become reducible only by the **SAT** rule.

Solution soundness is again by constructing an interpretation but is much more involved. Essentially, we construct an interpretation much as we did for R_{UF}, but then we modify it to ensure the array axioms are satisfied.

Refutation and solution completeness follow from soundness and termination, as in R_{UF} case.

More details in Section 5 of Jovanović and Barrett, "Being Careful about Theory Combination", 2013.

Soundness, Termination, and Completeness

Refutation soundness is straightforward and follows from the $\mathcal{T}_A$ axioms.

Termination follows from the following argument. Once we add all of the $i_{a,b}$ variables, no rule introduces new variables. There are only a finite number of terms that match the conclusions that can be constructed with a finite number of variables, so eventually, Γ will become reducible only by the **SAT** rule.

Solution soundness is again by constructing an interpretation but is much more involved. Essentially, we construct an interpretation much as we did for R_{UF}, but then we modify it to ensure the array axioms are satisfied.

Refutation and solution completeness follow from soundness and termination, as in R_{UF} case.

More details in Section 5 of Jovanović and Barrett, “Being Careful about Theory Combination”, 2013.

Soundness, Termination, and Completeness

Refutation soundness is straightforward and follows from the $\mathcal{T}_A$ axioms.

Termination follows from the following argument. Once we add all of the $i_{a,b}$ variables, no rule introduces new variables. There are only a finite number of terms that match the conclusions that can be constructed with a finite number of variables, so eventually, Γ will become reducible only by the **SAT** rule.

Solution soundness is again by constructing an interpretation but is much more involved. Essentially, we construct an interpretation much as we did for R_{UF}, but then we modify it to ensure the array axioms are satisfied.

Refutation and solution completeness follow from soundness and termination, as in R_{UF} case.

More details in Section 5 of Jovanović and Barrett, “Being Careful about Theory Combination”, 2013.

Soundness, Termination, and Completeness

Refutation soundness is straightforward and follows from the $\mathcal{T}_A$ axioms.

Termination follows from the following argument. Once we add all of the $i_{a,b}$ variables, no rule introduces new variables. There are only a finite number of terms that match the conclusions that can be constructed with a finite number of variables, so eventually, Γ will become reducible only by the **SAT** rule.

Solution soundness is again by constructing an interpretation but is much more involved. Essentially, we construct an interpretation much as we did for R_{UF}, but then we modify it to ensure the array axioms are satisfied.

Refutation and solution completeness follow from soundness and termination, as in R_{UF} case.

More details in Section 5 of Jovanović and Barrett, “Being Careful about Theory Combination”, 2013.

Soundness, Termination, and Completeness

Refutation soundness is straightforward and follows from the $\mathcal{T}_A$ axioms.

Termination follows from the following argument. Once we add all of the $i_{a,b}$ variables, no rule introduces new variables. There are only a finite number of terms that match the conclusions that can be constructed with a finite number of variables, so eventually, Γ will become reducible only by the **SAT** rule.

Solution soundness is again by constructing an interpretation but is much more involved. Essentially, we construct an interpretation much as we did for R_{UF}, but then we modify it to ensure the array axioms are satisfied.

Refutation and solution completeness follow from soundness and termination, as in R_{UF} case.

More details in Section 5 of Jovanović and Barrett, "Being Careful about Theory Combination", 2013.