# Verification

Lecture 31

Martin Zimmermann

UNIVERSITÄT
DES
SAARLANDES

# Plan for today

- Deductive verification
    - Congruence closure
    - DAG method

# Review: The Theory of Equality $T_E$

$$\Sigma_E : \{=,\ a,\ b,\ c,\ \ldots,\ f,\ g,\ h,\ \ldots,\ p,\ q,\ r,\ \ldots\}$$

uninterpreted symbols:

- constants $a, b, c, \ldots$
- functions $f, g, h, \ldots$
- predicates $p, q, r, \ldots$

Example:

$x = y \wedge f(x) \neq f(y)$ $T_E$-unsatisfiable

$f(x) = f(y) \wedge x \neq y$ $T_E$-satisfiable

$f(f(f(a))) = a \wedge f(f(f(f(f(a))))) = a \wedge f(a) \neq a$

$T_E$-unsatisfiable

## Axioms of $T_E$

1. $\forall x.\ x = x$ (reflexivity)
2. $\forall x, y.\ x = y \rightarrow y = x$ (symmetry)
3. $\forall x, y, z.\ x = y \wedge y = z \rightarrow x = z$ (transitivity)

define $=$ to be an equivalence relation.

Axiom schema

4. for each positive integer $n$ and $n$-ary function symbol $f$,

$$\forall x_1, \ldots, x_n, y_1, \ldots, y_n.\ \bigwedge_i x_i = y_i \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n) \qquad \text{(congruence)}$$

For example,

$$\forall x, y.\ x = y \rightarrow f(x) = f(y)$$

Then

$$x = g(y, z) \rightarrow f(x) = f(g(y, z))$$

is $T_E$-valid.

Axiom schema

5. for each positive integer $n$ and $n$-ary predicate symbol $p$,

$$\forall x_1, \ldots, x_n, y_1, \ldots, y_n. \bigwedge_i x_i = y_i \rightarrow (p(x_1, \ldots, x_n) \leftrightarrow p(y_1, \ldots, y_n)) \qquad \text{(equivalence)}$$

Thus,

$$x = y \rightarrow (p(x) \leftrightarrow p(y))$$

is $T_E$-valid.

## We discuss $T_E$-formulae without predicates

For example, for $\Sigma_E$-formula

$$F: \; p(x) \wedge q(x,y) \wedge q(y,z) \rightarrow \neg q(x,z)$$

introduce fresh constant $\bullet$, and fresh functions $f_p$ and $f_q$, and transform $F$ to

$$G: \; f_p(x) = \bullet \wedge f_q(x,y) = \bullet \wedge f_q(y,z) = \bullet \rightarrow f_q(x,z) \neq \bullet \, .$$

# Equivalence and Congruence Relations: Basics

Binary relation $R$ over set $S$

- is an equivalence relation if
  - reflexive: $\forall s \in S.\ sRs$;
  - symmetric: $\forall s_1, s_2 \in S.\ s_1 R s_2 \ \rightarrow\ s_2 R s_1$;
  - transitive: $\forall s_1, s_2, s_3 \in S.\ s_1 R s_2 \ \land\ s_2 R s_3 \ \rightarrow\ s_1 R s_3$.

Example:
Define the binary relation $\equiv_2$ over the set $\mathbb{Z}$ of integers

$$m \equiv_2 n \quad \text{iff} \quad (m \bmod 2) = (n \bmod 2)$$

That is, $m, n \in \mathbb{Z}$ are related iff they are both even or both odd.
$\equiv_2$ is an equivalence relation

- is a congruence relation if in addition

$$\forall \bar{s}, \bar{t}.\ \bigwedge_{i=1}^{n} s_i R t_i \ \rightarrow\ f(\bar{s}) R f(\bar{t})\ .$$

## Classes

For $\left\{\begin{array}{c}\text{equivalence}\\ \text{congruence}\end{array}\right\}$ relation $R$ over set $S$,

The $\left\{\begin{array}{c}\text{equivalence}\\ \text{congruence}\end{array}\right\}$ class of $s \in S$ under $R$ is

$$[s]_R \stackrel{\text{def}}{=} \{s' \in S \;:\; sRs'\} \,.$$

Example:

The equivalence class of 3 under $\equiv_2$ over $\mathbb{Z}$ is

$$[3]_{\equiv_2} = \{n \in \mathbb{Z} \;:\; n \text{ is odd}\} \,.$$

## Partitions

A partition $P$ of $S$ is a set of subsets of $S$ that is

- total $\left(\bigcup_{S' \in P} S'\right) = S$
- disjoint $\forall S_1, S_2 \in P.\, S_1 \cap S_2 = \varnothing$

## Quotient

The quotient $S/R$ of $S$ by $\left\{\begin{array}{c}\text{equivalence}\\ \text{congruence}\end{array}\right\}$ relation $R$ is the set of $\left\{\begin{array}{c}\text{equivalence}\\ \text{congruence}\end{array}\right\}$ classes

$$S/R \; = \; \{[s]_R \; : \; s \in S\} \, .$$

It is a partition

Example: The quotient $\mathbb{Z}/\equiv_2$ is a partition of $\mathbb{Z}$. The set of equivalence classes

$$\{\{n \in \mathbb{Z} \; : \; n \text{ is odd}\}, \{n \in \mathbb{Z} \; : \; n \text{ is even}\}\}$$

Note duality between relations and classes

# Refinements

Two binary relations $R_1$ and $R_2$ over set $S$.

$R_1$ is refinement of $R_2$, $R_1 \prec R_2$, if

$$\forall s_1, s_2 \in S.\ s_1 R_1 s_2 \ \rightarrow\ s_1 R_2 s_2 \ .$$

$R_1$ refines $R_2$.

Examples:

- For $S = \{a, b\}$,

  $R_1 : \{aR_1b\}$ $\qquad$ $R_2 : \{aR_2b,\ bR_2b\}$

  Then $R_1 \prec R_2$
- For set $S$,

  $R_1$ induced by the partition $P_1 : \{\{s\} \ : \ s \in S\}$

  $R_2$ induced by the partition $P_2 : \{S\}$

  Then $R_1 \prec R_2$.
- For set $\mathbb{Z}$

  $R_1 : \{xR_1y \ : \ x \bmod 2 = y \bmod 2\}$

  $R_2 : \{xR_2y \ : \ x \bmod 4 = y \bmod 4\}$

  Then $R_2 \prec R_1$.

## Closures

Given binary relation $R$ over $S$.
The equivalence closure $R^E$ of $R$ is the equivalence relation s.t.

- $R$ refines $R^E$, i.e. $R \prec R^E$;
- for all other equivalence relations $R'$ s.t. $R \prec R'$,
  either $R' = R^E$ or $R^E \prec R'$

That is, $R^E$ is the "smallest" equivalence relation that "covers" $R$.

Example: If $S = \{a, b, c, d\}$ and $R = \{aRb, bRc, dRd\}$, then

- $aRb, bRc, dRd \in R^E$ since $R \subseteq R^E$;
- $aRa, bRb, cRc \in R^E$ by reflexivity;
- $bRa, cRb \in R^E$ by symmetry;
- $aRc \in R^E$ by transitivity;
- $cRa \in R^E$ by symmetry.

Hence,

$$R^E = \{aRb, bRa, aRa, bRb, bRc, cRb, cRc, aRc, cRa, dRd\}.$$

Similarly, the congruence closure $R^C$ of $R$ is the "smallest" congruence relation that "covers" $R$.

# Congruence Closure Algorithm

Given $\Sigma_E$-formula

$$F:\ s_1 = t_1 \ \wedge\ \cdots\ \wedge\ s_m = t_m \ \wedge\ s_{m+1} \neq t_{m+1} \ \wedge\ \cdots\ \wedge\ s_n \neq t_n$$

decide if $F$ is $\Sigma_E$-satisfiable.

Definition: For $\Sigma_E$-formula $F$,
the subterm set $S_F$ of $F$ is the set that contains precisely the subterms of $F$.

Example: The subterm set of

$$F:\ f(a,b) = a \ \wedge\ f(f(a,b),b) \neq a$$

is

$$S_F = \{a,\ b,\ f(a,b),\ f(f(a,b),b)\}\ .$$

## The Algorithm

Given $\Sigma_E$-formula $F$

$$F: \; s_1 = t_1 \; \wedge \; \cdots \; \wedge \; s_m = t_m \; \wedge \; s_{m+1} \neq t_{m+1} \; \wedge \; \cdots \; \wedge \; s_n \neq t_n$$

with subterm set $S_F$, $F$ is $T_E$-satisfiable iff there exists a congruence relation $\sim$ over $S_F$ such that

- for each $i \in \{1, \ldots, m\}$, $s_i \sim t_i$;
- for each $i \in \{m+1, \ldots, n\}$, $s_i \not\sim t_i$.

Such congruence relation $\sim$ defines $T_E$-interpretation $I : (D_I, \alpha_I)$ of $F$. $D_I$ consists of $|S_F/\sim|$ elements, one for each congruence class of $S_F$ under $\sim$.

Instead of writing $I \models F$ for this $T_E$-interpretation, we abbreviate

$$\sim \models F$$

The goal of the algorithm is to construct the congruence relation of $S_F$, or to prove that no congruence relation exists.

$$F: \quad \underbrace{s_1 = t_1 \wedge \cdots \wedge s_m = t_m}_{\text{generate congruence closure}} \quad \wedge \underbrace{s_{m+1} \neq t_{m+1} \wedge \cdots \wedge s_n \neq t_n}_{\text{search for contradiction}}$$

The algorithm performs the following steps:

1. Construct the congruence closure $\sim$ of

$$\{s_1 = t_1, \ldots, s_m = t_m\}$$

   over the subterm set $S_F$. Then

$$\sim \models s_1 = t_1 \wedge \cdots \wedge s_m = t_m .$$

2. If for any $i \in \{m+1, \ldots, n\}$, $s_i \sim t_i$, return unsatisfiable.
3. Otherwise, $\sim \models F$, so return satisfiable.

How do we actually construct the congruence closure in Step 1?

Initially, begin with the finest congruence relation $\sim_0$ given by the partition

$$\{\{s\} \; : \; s \in S_F\} \, .$$

That is, let each term of $S_F$ be its own congruence class.

Then, for each $i \in \{1, \ldots, m\}$, impose $s_i = t_i$ by merging the congruence classes

$$[s_i]_{\sim_{i-1}} \quad \text{and} \quad [t_i]_{\sim_{i-1}}$$

to form a new congruence relation $\sim_i$. To accomplish this merging,

- form the union of $[s_i]_{\sim_{i-1}}$ and $[t_i]_{\sim_{i-1}}$
- propagate any new congruences that arise within this union.

The new relation $\sim_i$ is a congruence relation in which $s_i \sim t_i$.