CS389L: Automated Logical Reasoning

Lecture 11: Theory of Equality with Uninterpreted Functions

Işıl Dillig

Review

- Previous lecture: talked about signature and axioms of $T_=$

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

- Axioms:
  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)
  4. $\forall x_1, \ldots, x_n, y_1, \ldots, y_n.\ \bigwedge_i x_i = y_i \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$ (congruence)
  5. for each positive integer n and n-ary predicate symbol p, $\forall x_1, \ldots, x_n, y_1, \ldots, y_n.\ \bigwedge_i x_i = y_i \ \rightarrow\ (p(x_1, \ldots, x_n)\ \leftrightarrow\ p(y_1, \ldots, y_n))$ (equivalence)

Overview

- Today: look at decision procedures for deciding satisfiability in the quantifier-free fragment of $T_=$
- However, our decision procedure has two "restrictions":
  - formulas consist of conjunctions of literals
  - we'll allow functions, but no predicates
- However, these "restrictions" are not real restrictions – why?

Eliminating Predicates

- Simple transformation yields equisatisfiable formula with only functions
- The trick: For each relation constant p:
  1. introduce a fresh function constant f_p
  2. rewrite $p(x_1, \ldots, x_n)$ as $f_p(x_1, \ldots, x_n) = t$

  where t is a fresh object constant
- Example: How do we transform $x = y \rightarrow (p(x) \leftrightarrow p(y))$ to equisat formula?

$T_=$ without Predicates

- Signature without predicates:

$$\Sigma_= : \{=,\ a,\ b,\ c,\ \ldots,\ f,\ g,\ h,\ \ldots\}$$

- Axioms:
  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)
  4. $\forall x_1, \ldots, x_n, y_1, \ldots, y_n.\ \bigwedge_i x_i = y_i \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$ (congruence)

Examples

- Let's consider some examples
- Is the formula $x \neq y \wedge f(x) = f(y)$ sat, unsat, valid?
- What about $x = g(y, z) \rightarrow f(x) = f(g(y, z))$?
- What about $f(a) = a \wedge f(f(a)) \neq a$?
- What about $f(f(f(a))) = a \wedge f(f(f(f(f(a))))) = a \wedge f(a) \neq a$?

Equivalence Relations

- Decision procedure for theory of equality known as congruence closure algorithm
- Computes the congruence closure of the binary relation defined by formula $\Rightarrow$ need to understand congruence closure
- A binary relation R over a set S is an equivalence relation if
  1. reflexive: $\forall s \in S.\ sRs$
  2. symmetric: $\forall s_1, s_2 \in S.\ s_1 R s_2 \ \rightarrow\ s_2 R s_1$;
  3. transitive: $\forall s_1, s_2, s_3 \in S.\ s_1 R s_2 \ \wedge\ s_2 R s_3 \ \rightarrow\ s_1 R s_3$.

Examples

- Which of these are equivalence relations?
  - The relation $\equiv_2$ over $\mathbb{Z}$?
  - The relation $\geq$ over $\mathbb{N}$?
  - The relation $R(x, y)$ defined as $|x| = |y|$ on $\mathbb{R}$?

Congruence Relations

- Consider set S equipped with functions $F = \{f_1, \ldots, f_n\}$
- A relation R over S is a congruence relationif it is an equivalence relation and for every n'ary function $f \in F$:

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

- Which of these are congruence relations?
  - The relation $=$ on $\mathbb{N}$ equipped with a successor function?
  - The relation $\equiv_2$ on $\mathbb{N}$ equipped with a successor function?
  - The relation $R(x, y)$ defined as $|x| = |y|$ on $\mathbb{Z}$ equipped with successor function?

Equivalence and Congruence Classes

- For a given equivalence relation over S, every member of S belongs to an equivalence class
- The equivalence class of $s \in S$ under R is the set:

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

- If R is a congruence relation, then this set is called congruence class
- Example: What is the equivalence class of 1 under $\equiv_2$?
- What is the equivalence class of 6 under $\equiv_3$?

Equivalence Closure

- The equivalence closure R^E of a binary relation R over S is the equivalence relation such that:
  1. $R \subseteq R^E$
  2. for all other equivalence relations R' s.t. $R \subseteq R'$, $R^E \subseteq R'$
- Thus, R^E is the smallest equivalence relation that includes R.

Equivalence Closure Example

- Consider set $S = \{a, b, c, d\}$ and binary relation

$$R : \{\langle a, b\rangle, \langle b, c\rangle, \langle d, d\rangle\}$$

- Is R an equivalence relation?
- What is the equivalence closure of R?

Congruence Closure

- Given a set S and binary relation R, we also define congruence closure of R
- Congruence closure is similar to equivalence closure, but it is the smallest congruence relation that covers R
- Formally, the congruence closure R^C of a binary relation R over S is the congruence relation such that:
  1. $R \subseteq R^E$
  2. for all other congruence relations R' s.t. $R \subseteq R'$, $R^E \subseteq R'$

Example

- Consider the set $S = \{a, b, c\}$ and function f such that:

$$f(a) = b, \ f(b) = c, \ f(c) = c$$

- What is the congruence closure of relation $\{\langle a, b \rangle\}$?

Congruence Closure Algorithm

- The decision procedure for $T_=$ computes congruence closure of equality over the subterm set of formula
- Subterm set S_F of F is the set of all subterms of F
- Example: Consider formula $F : f(a, b) = a \land f(f(a, b), b) \neq a$
- What is S_F?

Satisfiability using Congruence Relations

- We can now define satisfiability of a $\Sigma_=$ formula in terms of congruence closure over subterm set
- Consider $\Sigma_=$ formula F:

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

- Let $R_F = \{\langle x, y \rangle \mid x = s_i, y = t_i, i \in [1, m]\}$
- Theorem: F is satisfiable if the congruence closure $\sim$ of R_F satisfies $s_i \not\sim t_i$ for all $i \in [m + 1, n]$

Congruence Closure Algorithm: Basic Idea

Congruence closure algorithm decide satisfiability of

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

1. Construct the congruence closure $\sim$ of R_F (defined previously) over the subterm set S_F.
2. If $s_i \sim t_i$ for any i in $[m + 1, n]$, F is unsatisfiable
3. Otherwise, F is satisfiable

Example

- Consider the formula $F : f(a, b) = a \land f(f(a, b), b) \neq a$
- We'll represent $\sim$ as a set of congruence classes, i.e., if t_1 and t_2 are in the same set, this means $t_1 \sim t_2$, otherwise $t_1 \not\sim t_2$
- First, construct subterm set S_F and place each subterm in a separate set:
- Because of equality $f(a, b) = a$, merge congruence classes of $f(a, b)$ and a:

Example, cont

- Formula $F: f(a,b) = a \wedge f(f(a,b),b) \neq a$
- Current congruence classes:

$$\{\{a, f(a,b)\}, \{b\}, \{f(f(a,b),b)\}\}$$

- Using $a \sim f(a,b)$ and $b \sim b$, what does function congruence imply?
- Thus, merge congruence classes of $f(a,b)$ and $f(f(a,b),b)$:

$$\{\{a, f(a,b), f(f(a,b),b)\}, \{b\}\}$$

- This represents the congruence closure over S_F.

Example, cont

- Formula $F: f(a,b) = a \wedge f(f(a,b),b) \neq a$
- Congruence closure: $\{\{a, f(a,b), f(f(a,b),b)\}, \{b\}\}$
- Is F satisfiable?
- Since a and $f(f(a,b),b)$ are in same congruence class, we have $a \sim f(f(a,b),b)$
- This contradicts $f(f(a,b),b) \neq a$!

Another Example

- Consider formula:

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

- What is the subterm set S_F?
- Initially, place each subterm in its own congruence class:

$$\{\{a\}, \{f(a)\}, \{f^2(a)\}, \{f^3(a)\}, \{f^4(a)\}, \{f^5(a)\}\}$$

- Because of equality $f^3(a) = a$, $f^3(a)$ and a are placed in same congruence class:

Another Example, cont

- Formula $F: f^3(a) = a \wedge f^5(a) = a \wedge f(a) \neq a$
- Current congruence classes:

$$\{\{a, f^3(a)\}, \{f(a)\}, \{f^2(a)\}, \{f^4(a)\}, \{f^5(a)\}\}$$

- From $a = f^3(a)$, what can we infer using function congruence?
- Resulting congruence classes:

Another Example, cont

- Formula $F: f^3(a) = a \wedge f^5(a) = a \wedge f(a) \neq a$
- Current congruence classes:

$$\{\{a, f^3(a)\}, \{f(a), f^4(a)\}, \{f^2(a), f^5(a)\}\}$$

- Now, process equality $f^5(a) = a$; which classes do we merge?
- From $a = f^2(a)$, what can we infer via function congruence?
- Thus, merge the two congruence classes:

$$\{\{a, f(a), f^2(a), f^3(a), f^4(a), f^5(a)\}\}$$

Another Example, cont

- Formula $F: f^3(a) = a \wedge f^5(a) = a \wedge f(a) \neq a$
- Currenct congruence classes:

$$\{\{a, f(a), f^2(a), f^3(a), f^4(a), f^5(a)\}\}$$

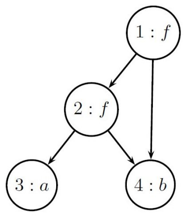
- Is the formula satisfiable?
- Since $f(a)$ and a are in same congruence class, this contradicts $f(a) \neq a$

One More Example

- Consider formula $F: f(x) = f(y) \land x \neq y$
- What is the subterm set? $\{x, y, f(x), f(y)\}$
- Each subterm starts in its own congruence class: $\{\{x\}, \{y\}, \{f(x)\}, \{f(y)\}\}$
- Process equality $f(x) = f(y) \Rightarrow$
- What new equalities can we infer from congruence?
- Is the formula satisfiable?

Algorithm to Compute Congruence Closure

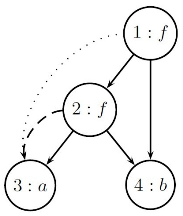
- To compute congruence closure efficiently, we'll represent the subterm set of the formula as a DAG



- Each node corresponds to a subterm and has unique id
- Edges point from function symbol to arguments
- Question: What subterm does node labeled 1 represent? f(f(a,b), b)

Representative of Congruence Class

- To compute congruence closure, we need to merge congruence classes
- To do this efficiently, each congruence class has a representative: When merging two classes, only need to update the representative



- Each subterm contains a find pointer that eventually leads to the representative of its congruence class (representative points to itself)
- In this example, $a, f(a, b), f(f(a, b), b)$ are in same congruence class; a is the representative

Parents of a Subterm

- In addition to efficiently finding representative, also need to efficiently find parents of terms – why?
- Thus, keep pointer from representative of congruence class to parents of all subterms in the congruence class
- If a term is not a representative, then its parents field is empty

Merging Congruence Classes

- Using this data structure, how do we merge congruence classes of two terms t_1 and t_2?
- First find representatives of t_1 and t_2 by chasing pointers
- Want to make $Rep(t_2)$ new representative for merged class
- Thus, change find field of $Rep(t_1)$ to point to $Rep(t_2)$
- Update parents: add parent terms stored in $Rep(t_1)$ to those of $Rep(t_2)$, and remove parents stored in $Rep(t_1)$

Processing Equalities, cont

To process equality $t_1 = t_2$:

1. Find representatives of t_1 and t_2
2. Merge equivalence classes
3. Retrieve the set of parents P_1, P_2 stored in $Rep(t_1), Rep(t_2)$
4. For each $(p_i, p_j) \in P_1 \times P_2$, if p_i and p_j are congruent, process equality $p_i = p_j$

Observe: Processing one equality creates new equalities, which in turn might generate other new equalities!

Full Algorithm for Deciding Satisfiability

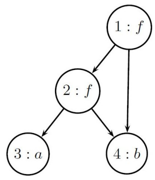
Algorithm to decide satisfiability of $T_=$ formula

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

1. Compute subterms and construct initial DAG (each node's representative is itself)
2. For each $i \in [1, m]$, process equality $s_i = t_i$ as described
3. For each $i \in [m+1, n]$, check if $Rep(s_i) = Rep(t_i)$
4. If there exists some $i \in [m+1, n]$ for which $Rep(s_i) = Rep(t_i)$, return UNSAT
5. If for all i, $Rep(s_i) \neq Rep(t_i)$, return SAT

Example

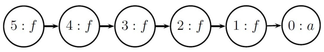
- Consider formula $F: f(a,b) = a \wedge f(f(a,b),b) \neq a$
- Subterms: $a, b, f(a,b), f(f(a,b),b)$



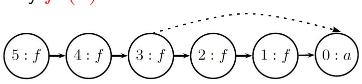
- Construct initial DAG
- Process equality $f(a,b) = a$
- Are parents $f(a,b)$ and $f(f(a,b),b)$ congruent?
- Yes, so process equality $f(a,b) = f(f(a,b),b)$
- Formula unsatisfiable because $f(f(a,b),b)$ and a have same representative!

Example II

- Consider formula: $F: f^3(a) = a \wedge f^5(a) = a \wedge f(a) \neq a$
- Initial DAG:



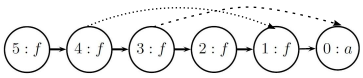
- Process equality $f^3(a) = a$:



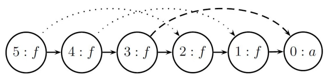
- Are parents congruent? Yes
- Process equality $f^4(a) = f(a)$

Example II, cont

- After merging classes:

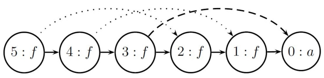


- Are $f^4(a)$'s and $f(a)$'s parents congruent? Yes
- Process equality $f^5(a) = f^2(a)$

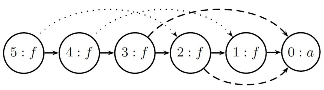


Example II, cont

- Formula: $F: f^3(a) = a \wedge f^5(a) = a \wedge f(a) \neq a$



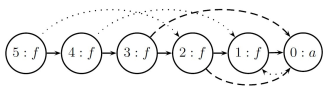
- Process equality $f^5(a) = a$:



- Now, parents $f^2(a)$ and a congruent; so process equality $f^3(a) = f(a)$

Example II, cont

- Formula: $F: f^3(a) = a \wedge f^5(a) = a \wedge f(a) \neq a$



- Now, everything in same congruence class; so we are done.
- Formula UNSAT because a and $f(a)$ have same representative

Summary

- Congruence closure algorithm is used for determining satisfiability of $T_=$ formulas (without disjunction)
- Deciding conjuctive $T_=$ formulas is inexpensive: our algorithm is $O(e^2)$, but can be solved in $O(e\ log(e))$
- To decide satisfiability of formulas containing disjunctions, can either convert to DNF or use $\text{DPLL}(\mathcal{T})$ (more on this later)