Lecture 21

Pete Manolios
Northeastern

Presentation/Project Schedule

- 11/27
  - Ben B (40 min)
  - Dustin (40 min)
  - Alex (20 min)
- 11/30
  - Ankit (40 min)
  - Taylor (20 min)
  - Nathaniel (20 min)
  - Daniel (20 min)
- 12/4
  - Michael (20 min)
  - Drew (40 min)
  - Ben Q (40 min)

Meet with me to review slides at least 3 days before your presentations

Exam 2:

Distribute 11/30 after class

Due 12/1 by 3PM (email)

Recall: Sequent Rules

Reflexivity Rule for Equality ($\equiv$)

$$\frac{}{t \equiv t}$$

Substitution Rule for Equality (Sub)

$$\frac{\Gamma \qquad\qquad \varphi\frac{t}{x}}{\Gamma \quad t \equiv t' \quad \varphi\frac{t'}{x}}$$

- Can derive that equality is symmetric and transitive (so equivalence)
- Can derive that equality is a congruence
- Suppose Φ is a set of equations (universal formulas of the form s = t) and ϕ is an equation
  - Then, $\Phi \vDash \phi$ iff $\Phi \vdash \phi$ where we only use Assm, Sub, equivalence and congruence rules (Birkhoff's theorem)
  - More on this soon

Equality Decision Procedure

- Consider a universal formula $\langle\forall x_1,\ldots,x_n\ \phi(x_1,\ldots,x_n)\rangle$ which does not contain any predicates, but can contain =, vars, functions, constants
- The formula is valid iff $\langle\exists x_1,\ldots,x_n\ \neg\phi(x_1,\ldots,x_n)\rangle$ is Unsat
- Iff $\neg\phi(c_1,\ldots,c_n)$ is Unsat, via Skolemization
- We can generate equivalent DNF: $\psi_1(c_1,\ldots,c_n) \vee \cdots \vee \psi_k(c_1,\ldots,c_n)$
- Which is Unsat iff $\psi_i(c_1,\ldots,c_n)$ is Unsat for all i (there are no vars)
- Note: $\psi_i(c_1,\ldots,c_n)$ is of the form $s_1{=}t_1 \wedge \cdots \wedge s_l{=}t_l \wedge u_1{\neq}v_1 \wedge \cdots \wedge u_m{\neq}v_m$
- Which is Unsat iff $s_1{=}t_1 \wedge \cdots \wedge s_l{=}t_l \Rightarrow u_1{=}v_1 \vee \cdots \vee u_m{=}v_m$ is Valid
- Iff for some j, $s_1{=}t_1 \wedge \cdots \wedge s_l{=}t_l \Rightarrow u_j{=}v_j$ is Valid
- So, we can reduce validity of FO formulas with no predicates to validity of equational logic with ground terms:
  - $\Phi \vDash s{=}t$ where $s{=}t$ and all elements of Φ are ground equations
  - By Birkhoff's theorem, equivalent to $\Phi \vdash \phi$ where we only use Assm, ~~Sub~~ (no vars), equivalence and congruence rules

Reduction to Propositional Logic

- Ackermann's idea: reduce the problem to propositional logic
- Consider: $f(f(f(c)))=c \wedge f(f(c))=c \;\Rightarrow\; f(c) = c$ (Valid or not?)
- Remove functions: Introduce variables for subterms, say $x_k=f^k(c)$ for $0 \leq k \leq 3$ and add constraints for congruence properties over subterms
  - $x_3=x_0 \wedge x_2=x_0 \wedge (x_0=x_1 \Rightarrow x_1=x_2) \wedge (x_0=x_2 \Rightarrow x_1=x_3) \wedge (x_1=x_2 \Rightarrow x_2=x_3)$
  - Check if this implies $x_1=x_0$
- Remove =: replace equations, say $s=t$, with propositional atoms, say $P_{s,t}$, and add constraints for equivalence properties $(P_{s,t} \wedge P_{t,u} \Rightarrow P_{s,u})$
- Now, we can use a propositional SAT solver

Ackermann Example

- Consider: $f(f(f(c)))=c \wedge f(f(c))=c \ \Rightarrow\ f(c) = c$
- Remove functions: Introduce variables for subterms, say
  - $x_k=f^k(c)$ for $0\leq k\leq 3$, so: $x_0=c$, $x_1=f(c)$, $x_2=f(f(c))$, $x_3=f(f(f(c)))$
- Rewrite problem: $x_3=x_0 \wedge x_2=x_0 \ \Rightarrow\ x_1=x_0$
- Add hyps: constraints for congruence properties over subterms
  - $(x_0=x_1 \Rightarrow x_1=x_2) \wedge (x_0=x_2 \Rightarrow x_1=x_3) \wedge (x_1=x_2 \Rightarrow x_2=x_3)$
  - Note $(x_0=x_3 \Rightarrow x_1=x_4)$, etc not needed since x_4 is not a subterm
- Remove =: replace equations with propositional atoms
  - $P_{3,0} \wedge P_{2,0} \wedge (P_{0,1} \Rightarrow P_{1,2}) \wedge (P_{0,2} \Rightarrow P_{1,3}) \wedge (P_{1,2} \Rightarrow P_{2,3}) \Rightarrow P_{1,0}$
- Add equivalence properties (as hyps) *Finish the reduction*

Optimizations?

  - $P_{0,0} \wedge P_{1,1} \wedge P_{2,2} \wedge P_{3,3} \wedge$
  - $(P_{0,1}\equiv P_{1,0})\wedge(P_{0,2}\equiv P_{2,0})\wedge(P_{0,3}\equiv P_{3,0})\wedge(P_{1,2}\equiv P_{2,1})\wedge(P_{1,3}\equiv P_{3,1})\wedge(P_{2,3}\equiv P_{3,2})\wedge$
  - $(P_{1,0}\wedge P_{0,2}\Rightarrow P_{1,2})\wedge(P_{1,0}\wedge P_{0,3}\Rightarrow P_{1,3})\wedge(P_{2,0}\wedge P_{0,3}\Rightarrow P_{2,3})\wedge(P_{0,1}\wedge P_{1,2}\Rightarrow P_{0,2})\wedge$
    $(P_{0,1}\wedge P_{1,3}\Rightarrow P_{0,3})\wedge(P_{2,1}\wedge P_{1,3}\Rightarrow P_{2,3})\wedge(P_{0,2}\wedge P_{2,1}\Rightarrow P_{0,1})\wedge(P_{0,2}\wedge P_{2,3}\Rightarrow P_{0,3})\wedge$
    $(P_{1,2}\wedge P_{2,3}\Rightarrow P_{1,3})\wedge(P_{0,3}\wedge P_{3,1}\Rightarrow P_{0,1})\wedge(P_{0,3}\wedge P_{3,2}\Rightarrow P_{0,2})\wedge(P_{1,3}\wedge P_{3,2}\Rightarrow P_{1,2})$

Congruence Closure

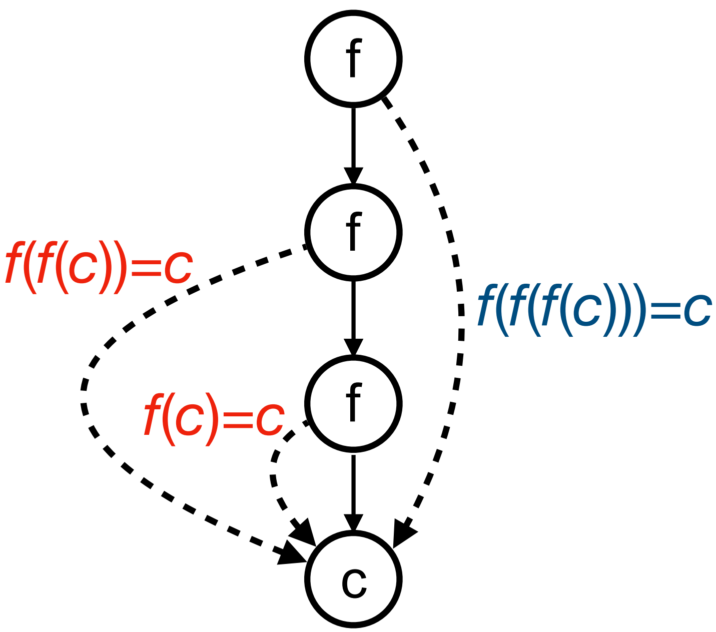
- Decision procedure for $\Phi \vDash s=t$ where $s=t$ and all elements of Φ are ground equations
- Let G be a set of terms closed under subterms
  - If $t \in G$ and s is a subterm of t, then $t \in G$
- ~ is a congruence on G: an equivalence, congruence on terms in G
- For $R \subseteq G \times G$, the *congruence closure* of R on G is the smallest congruence on G extending R
  - Start with R and apply equivalence, congruence rules until fixpoint
- Let $\Phi=\{s_1=t_1, \ldots, s_n=t_n\}$, G is the minimal set closed under subterms of $\{s_1, t_1, \ldots, s_n, t_n, s, t\}$, ~ the congruence closure of Φ on G. Then:
  - $\Phi \vDash s=t$ iff $s \sim t$
  - Can do this in P-time

Congruence Closure Algorithm

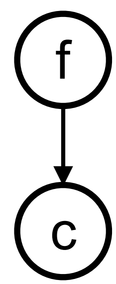
- Decision procedure for $\Phi \vDash s=t$ where $s=t$ and all elements of Φ are ground equations
- Main idea: use a graph with structure sharing to represent terms
- Start with ~ being the identity
- Each node (term) is mapped to its equivalence class
- For each assumption, $s_i=t_i$,
  - merge equivalence classes $[s_i]$, $[t_i]$
  - propagate congruences efficiently (using predecessor pointers)
- Check is $[s] = [t]$ after processing all hypotheses
- $O(m^2)$ algorithm due to Nelson, Oppen (m is the # edges in graph)

Congruence Closure Example

Consider: $f(f(f(c)))=c \wedge f(f(c))=c \;\Rightarrow\; f(c) = c$



$[f(f(c))]=[c]$, so $[f(f(f(c)))]=[f(c)]$ ie,
$[c]=[f(c)]$
congruence propagation



corresponds to $f(c)$

Graph representation allows structure sharing

corresponds to c

equivalence class of term

So, when we extend the congruence, by *union*ing [*s*] [*t*], we also have to union any terms of the form f(…s…) and f(…t…) if the rest of the arguments are in same class

Congruence Closure Algorithm

For each node n, we have:

- $l(n)$: function/constant symbol of n
- $d(n)$: # of successors of n (= arity $l(n)$)
- $n[i]$: the i^{th} successor of n
- $p(n) = \{m \mid \exists i\ m[i] \sim n\}$
- $c(n,m) = l(n)=l(m) \wedge \forall i\ n[i] \sim m[i]$

~ is a congruence if

- it is an equivalence
- if $l(n)=l(\mathrm{m}) \wedge \forall i\ n[i] \sim m[i]$ then $n \sim m$

merge(n, m):

if $n \nsim m$ then

 $P := p(n)$; $Q = p(m)$

 Union(n,m)

 for all $(p,q) \in P \times Q$ do

  if $p \nsim q \wedge c(p,q)$ then *merge*(p,q)

Merge all $s_i = t_i$

Use Union-Find algorithm