Logic of Equality and Uninterpreted Functions (EUF)

Deepak D'Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

26, 28 Apr and 03 Mar 2021

Outline

Logic of Equality and Uninterpreted Functions (EUF) (KS Ch 4)

- Boolean combinations of equality predicates.

EUF syntax

(Formula) $\varphi ::= \mathit{Atom} \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi$
(Atom) $\mathit{Atom} ::= \mathit{Term} = \mathit{Term}$
(Term) $\mathit{Term} ::= \mathit{Var} \mid \mathit{Const} \mid F(\mathit{Term})$

Example formula φ_1

$$(x_1 = x_2) \wedge (x_2 = x_3) \wedge (F(x_1) \neq F(x_3)).$$

Example formula φ_2

$$F(x) = F(G(y)) \vee x = y$$

Questions we want to answer

Given an EUF formula φ:

- Satisfiability: Does there exist (M, v), with $M = (D, I)$ and $D = \mathbb{Z}$ (or any infinite domain), such that $M, v \vDash \varphi$.
- Validity: Does $M, v \vDash \varphi$, for *every* (M, v), with $M = (D, I)$ and $D = \mathbb{Z}$ (or any infinite domain).

Exercise:

- Give examples of satisfiable, unsatisfiable, valid EUF formulas.
- What is the relation between satisfiabililty and validity?

Importance of EUF logic

Many practical applications. Arguing correctness of:

- Program transformation, compilation.
- Pipelining in a hardware circuit.

Example: Are these programs equivalent?

```
S1: z := (x1 + y1) * (x2 + y2);
```

```
T1: u1 := (x1 + y1);
T2: u2 := (x2 + y2);
T3: z := u1 * u2;
```

We want to check whether:

$$[u_1 = x_1 + y_1 \wedge u_2 = x_2 + y_2 \wedge z = u_1 * u_2] \implies z = (x_1 + y_1) * (x_2 + y_2).$$

We could check whether the EUF formula is valid:

$$[u_1 = F(x_1, y_1) \wedge u_2 = F(x_2, y_2) \wedge z = G(u_1, u_2)] \implies z = G(F(x_1, y_1), F(x_2, y_2)).$$

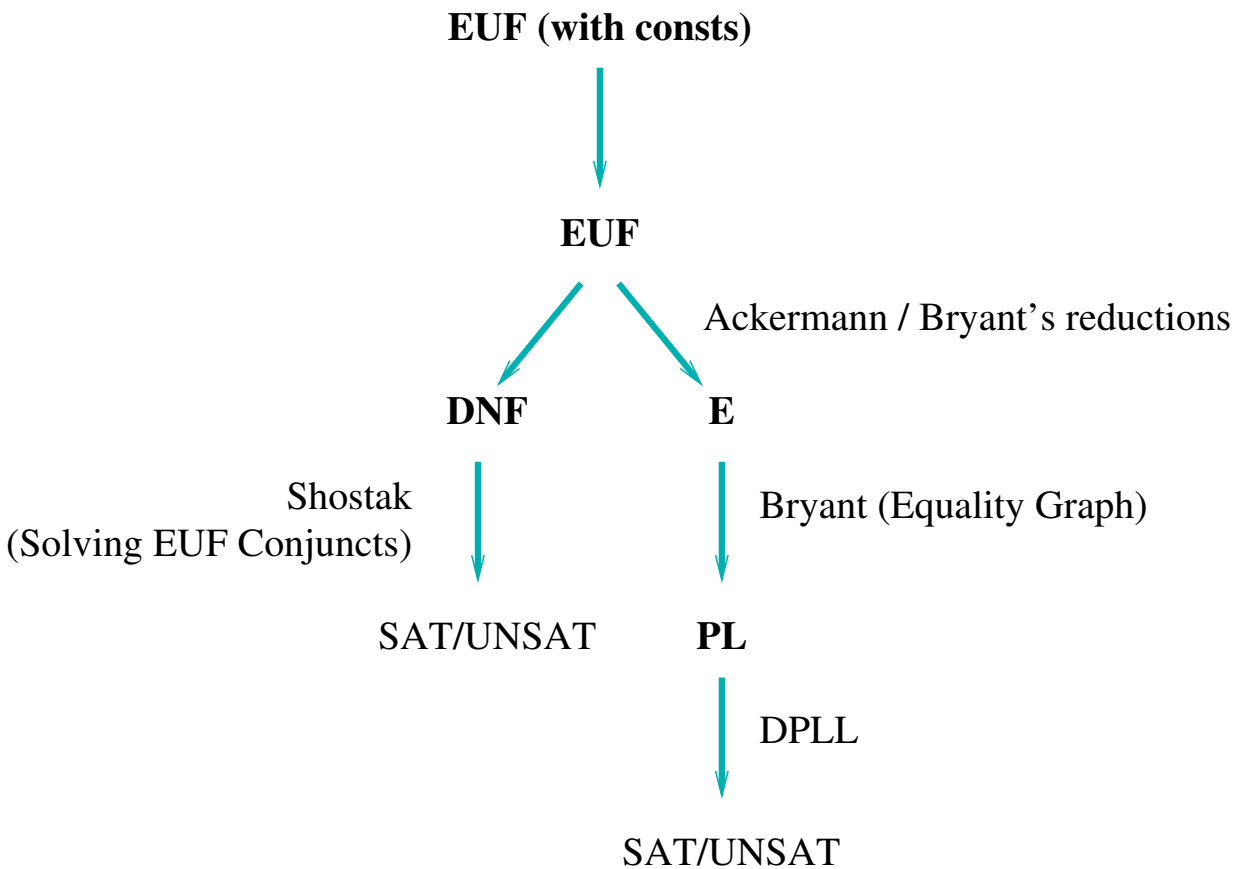
How do we decide satisfiability of EUF formulas?

How?

How do we decide satisfiability of EUF formulas?

How?

Strategies we will look at:



A brute-force algorithm

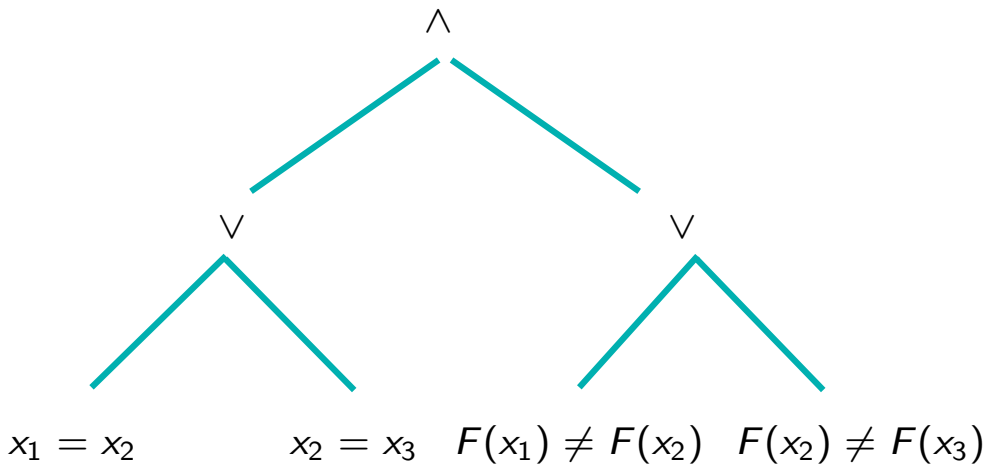
Given EUF formula φ:

1. Let V be the variables in φ, and let k be the number of distinct terms in φ.
2. Let $U = \{1, \ldots, k\}$
3. For each possible valuation v of V over U, check if $u \models \varphi$.
4. If some v satisfies φ, output SAT; else output UNSAT.

Example

Example

$$(x_1 = x_2 \vee x_2 = x_3) \wedge (F(x_1) \neq F(x_2) \vee F(x_2) \neq F(x_3))$$

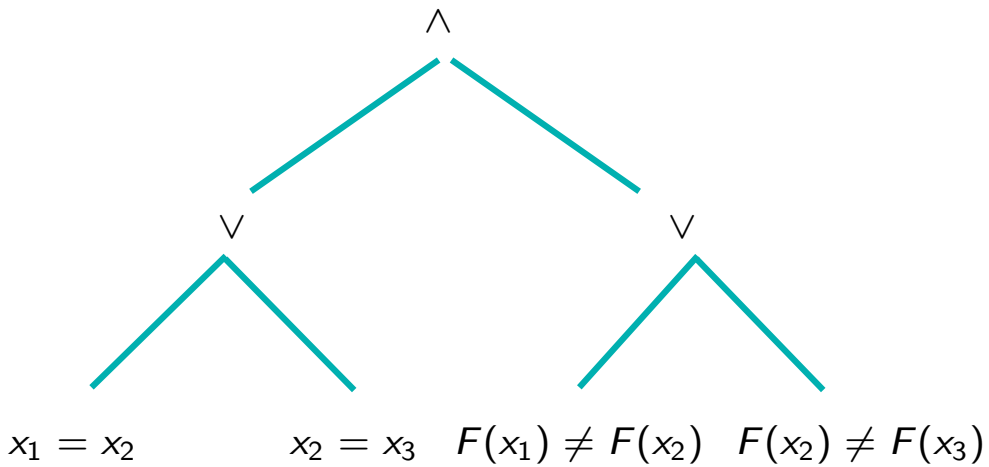


Correctness?

Example

Example

$$(x_1 = x_2 \vee x_2 = x_3) \wedge (F(x_1) \neq F(x_2) \vee F(x_2) \neq F(x_3))$$



Correctness? Claim: If M, v satisfies φ, then we can construct M', v' such that $M', v' \models \varphi$, and v' is over U.

Getting rid of constants (KS Sec 4.1.3)

Given φ in EUF (with consts), replace each constant k in φ by a new variable c_k and add conjuncts saying that $c_k \neq c_{k'}$ for each distinct constants k, k'. Example: Replace φ

$$(y = z \wedge z \neq 1) \vee ((x \neq z) \wedge x = 2)$$

by equisatisfiable φ':

$$[(y = z \wedge z \neq c_1) \vee ((x \neq z) \wedge x = c_2)] \wedge (c_1 \neq c_2).$$

Claim: φ is satisfiable iff φ' is.

Congruence Closure Algorithm (Shostak 1978) (KS Sec 4.3)

Given EUF formula φ as conjunction of literals:

1. Consider all subterms t of φ.
2. If $(t_1 = t_2)$ is a predicate in φ, put t_1, t_2 in the same equivalence class. All other terms in their own singleton equivalence classes.
3. If two classes share a term, merge them.
4. Apply congruence closure: If t_1 and t_2 are in the same equivalence class, then merge the equivalence classes of $F(t_1)$ and $F(t_2)$.
5. If there is a disequality $t_1 \neq t_2$ in φ, with t_1 and t_2 in the same equivalence class, return UNSAT. Else return SAT.

Example:

$$(x_1 = x_2) \wedge (x_2 = x_3) \wedge (x_4 = x_5) \wedge (x_5 \neq x_1) \wedge (F(x_1) \neq F(x_3)).$$

Correctness?

Congruence Closure Algorithm (Shostak 1978) (KS Sec 4.3)

Given EUF formula φ as conjunction of literals:

1. Consider all subterms t of φ.
2. If $(t_1 = t_2)$ is a predicate in φ, put t_1, t_2 in the same equivalence class. All other terms in their own singleton equivalence classes.
3. If two classes share a term, merge them.
4. Apply congruence closure: If t_1 and t_2 are in the same equivalence class, then merge the equivalence classes of $F(t_1)$ and $F(t_2)$.
5. If there is a disequality $t_1 \neq t_2$ in φ, with t_1 and t_2 in the same equivalence class, return UNSAT. Else return SAT.

Example:

$$(x_1 = x_2) \wedge (x_2 = x_3) \wedge (x_4 = x_5) \wedge (x_5 \neq x_1) \wedge (F(x_1) \neq F(x_3)).$$

Correctness? Claim: Algo is correct if it outputs SAT/UNSAT.

Ackermann's reduction (KS Sec 11.2.1)

Given EUF formula φ output E formula:

$$\varphi_{flat} \wedge FC_{\varphi},$$

where φ_{flat} replaces function applications like $F(G(x))$ by $f(g_1)$ etc, and FC encodes functional consistency:

$$x = y \implies F(x) \implies F(y).$$

Claim: φ is sat/valid iff $\varphi_{flat} \wedge FC_{\varphi}$ is sat/valid.

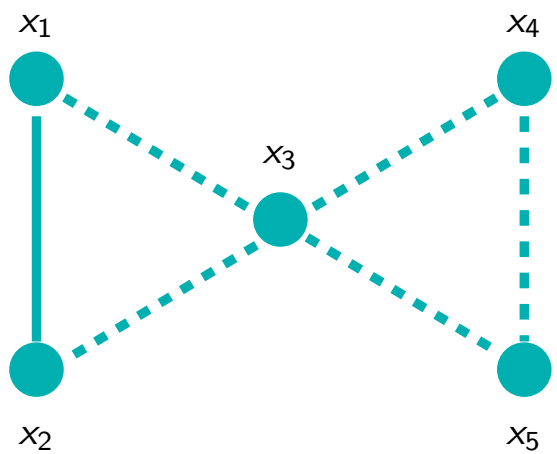
Exercise

Exercise: Give Ackermann's reduction for this formula

$$(x_1 = x_2) \implies F(F(G(x_1))) = F(F(G(x_2))).$$

Equality graph induced by an E formula (KS Sec 11.3)

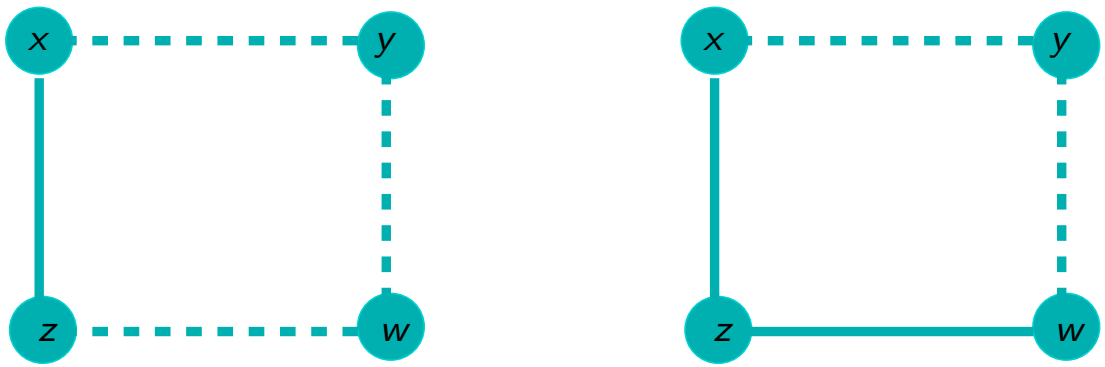
Let φ be a E formula. Then the equality graph G_φ induced by φ is an undirected graph with nodes as variables and "- - -"edge (x_i, x_j) iff the literal $x_i = x_j$ occurs in φ, and "—"edge (x_i, x_j) iff the literal $x_i \neq x_j$ occurs in φ.



- Contradictory cycles (cycle with exactly one disequality edge).
- Is an abstraction of the original formula.
- Can be used to simplify an E formula.

Contradictory Cycles

A conjunction φ of equality constraints is satisfiable iff the equality graph G_φ induced by φ has no contradictory cycles.



Proof (Exercise).

Solving E-formulas: DNF approach

Example φ

$$x = y \wedge [(y = z \wedge \neg(x = z)) \vee (y = z \wedge \neg(x = w))].$$

$e(\varphi)$: $\quad e_{xy} \wedge [(e_{yz} \wedge \neg e_{xz}) \vee (e_{yz} \wedge \neg e_{xw})]$.

Convert $e(\varphi)$ to DNF and check each disjunct using equality graph.

Solving E-formulas: DPLL(T) approach (KS Sec 3.4)

Example φ

$$x = y \wedge [(y = z \wedge \neg(x = z)) \vee (y = z \wedge \neg(x = w))].$$

$e(\varphi)$: $\quad e_{xy} \wedge [(e_{yz} \wedge \neg e_{xz}) \vee (e_{yz} \wedge \neg e_{xw})]$.

Check satisfiabilty of $e(\varphi)$ using DPLL:

1. If $e(\varphi)$ is SAT, check if satisfying assignment is T-valid;
   - If T-valid, then return SAT;
   - If not T-valid, add negation of the assignment as conflicting clause to $e(\varphi)$, and go back Step 1.
2. If UNSAT, report UNSAT.

Bryant's Graph-Based reduction of E to PL (KS Sec 11.5)

Given E formula φ output PL formula:

$$e(\varphi) \wedge \mathcal{B}_{trans},$$

where $e(\varphi)$ replaces each literal $x_i = x_j$ by a propositional symbol p_{ij}, and $\mathcal{B}_{trans}$ encodes transitivity constraints based on the non-polar graph induced by φ.

Bryant's Graph-Based reduction of E to PL (KS Sec 11.5)

Given E formula φ output PL formula:

$$e(\varphi) \wedge \mathcal{B}_{trans},$$

where $e(\varphi)$ replaces each literal $x_i = x_j$ by a propositional symbol p_{ij}, and $\mathcal{B}_{trans}$ encodes transitivity constraints based on the non-polar graph induced by φ.
Claim: φ is equisatisfiable with $e(\varphi) \wedge \mathcal{B}_{trans}$.

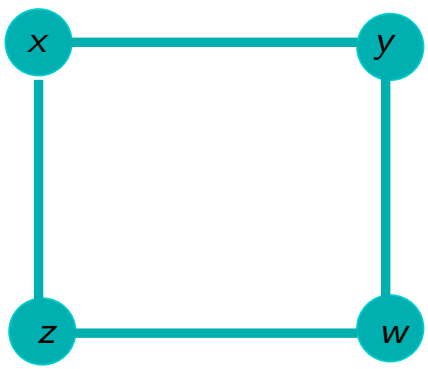
Bryant's Graph-Based reduction: Example

Example φ

$$x = y \wedge [(y = w \wedge w = z \wedge \neg(x = z)) \vee (y = w \wedge \neg(w = z))].$$

$e(\varphi)$: $e_{xy} \wedge [(e_{wy} \wedge e_{wz} \wedge \neg e_{xz}) \vee (e_{wy} \wedge \neg e_{wz})]$.

G_{φ}^{NP}:

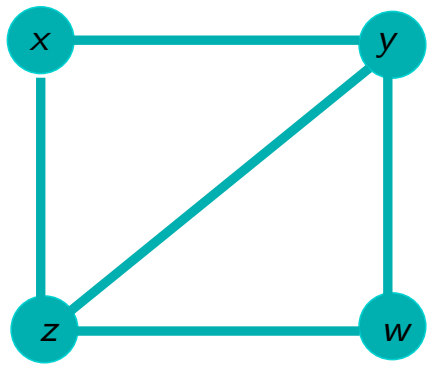


$\mathcal{B}_{trans}$:

$$\begin{aligned}
(e_{xy} \wedge e_{wy} \wedge e_{wz}) &\implies e_{xz} \wedge \\
(e_{wy} \wedge e_{wz} \wedge e_{xz}) &\implies e_{xy} \wedge \\
(e_{wz} \wedge e_{xz} \wedge e_{xy}) &\implies e_{wy} \wedge \\
(e_{xz} \wedge e_{xy} \wedge e_{wy}) &\implies e_{wz}
\end{aligned}$$

Bryant's Graph-Based reduction: Using chordal graph

Make G_φ^{NP} chordal (every simple cycle of ≥ 4 vertices has a chord).



Sufficient to check no contradictary triangles [Bryant-Velev CAV 2000):

$\mathcal{B}_{trans}$:

$$
\begin{aligned}
(e_{xy} \wedge e_{yz}) &\Longrightarrow e_{xz} \wedge \\
(e_{yz} \wedge e_{xz}) &\Longrightarrow e_{xy} \wedge \\
(e_{xz} \wedge e_{xy}) &\Longrightarrow e_{yz} \wedge \\
(e_{wy} \wedge e_{wz}) &\Longrightarrow e_{yz} \wedge \\
(e_{wz} \wedge e_{yz}) &\Longrightarrow e_{wy} \wedge \\
(e_{yz} \wedge e_{wy}) &\Longrightarrow e_{wz}.
\end{aligned}
$$

Using equality graph to simplify E formulas

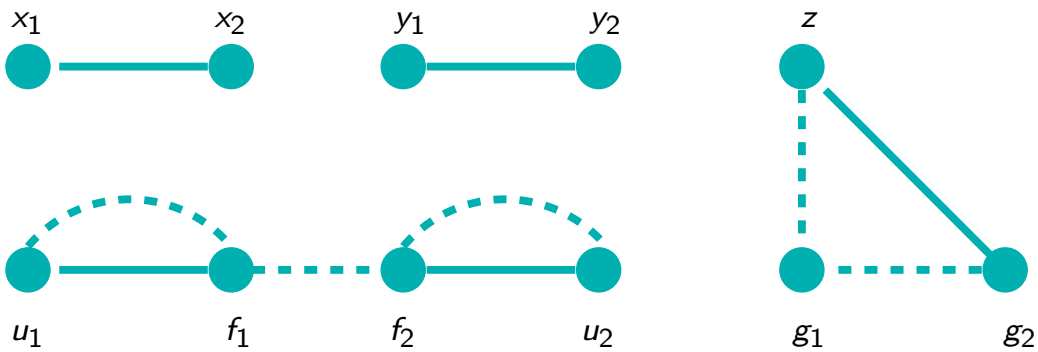
Given E formula φ:

1. Construct equality graph G_φ for φ.
2. If a literal does not occur as part of a contradictory cycle in G_φ, set it to *true*. Obtain φ' in this way.
3. Simplify φ' and go back to Step 2.
4. Ouput φ' as equisatisfiable to φ.

Equality graph: example

Example

$$(u_1 \neq f_1 \vee y_1 \neq y_2 \vee f_1 = f_2) \wedge$$
$$(x_1 \neq x_2 \vee u_2 \neq f_2 \vee g_1 = g_2) \wedge$$
$$(u_1 = f_1 \wedge u_2 = f_2 \wedge z = g_1 \wedge z \neq g_2).$$



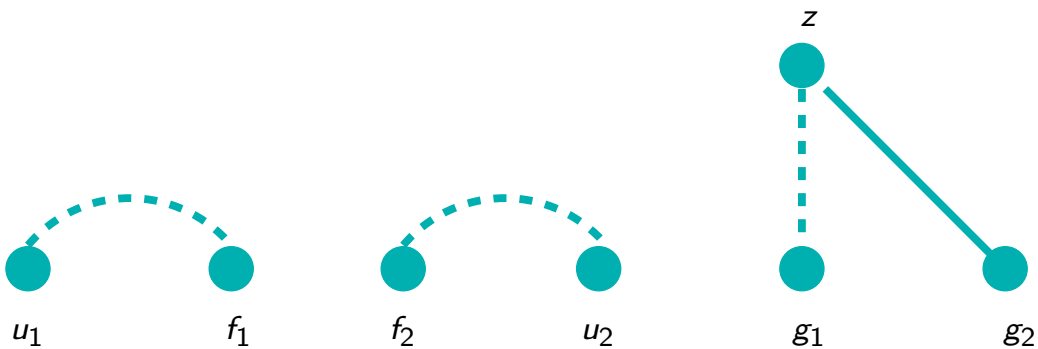
$$(u_1 \neq f_1 \vee \mathit{true} \vee \mathit{true}) \wedge$$
$$(\mathit{true} \vee u_2 \neq f_2 \vee g_1 = g_2) \wedge$$
$$(u_1 = f_1 \wedge u_2 = f_2 \wedge z = g_1 \wedge z \neq g_2).$$

Simplifies to:

$$u_1 = f_1 \wedge u_2 = f_2 \wedge z = g_1 \wedge z \neq g_2.$$

Equality graph: example contd.

$$u_1 = f_1 \wedge u_2 = f_2 \wedge z = g_1 \wedge z \neq g_2.$$



$$\mathit{true} \wedge \mathit{true} \wedge \mathit{true} \wedge \mathit{true}.$$

Simplifies to:

$$\mathit{true}.$$