Decision Procedures in First Order Logic

Decision Procedures for Equality Logic

Outline

- √ Introduction
  - √ Definition, complexity
  - √ Reducing Uninterpreted Functions to Equality Logic
  - √ Using Uninterpreted Functions in proofs
  - √ Simplifications
- Introduction to the decision procedures
  - The framework: assumptions and Normal Forms
  - General terms and notions
  - Solving a conjunction of equalities
  - Simplifications

Basic assumptions and notations

- Input formulas are in NNF
- Input formulas are checked for satisfiability

- Formula with Uninterpreted Functions: ϕ^{UF}
- Equality formula: ϕ^{E}
- We assume each EUF is transformed to EF.
  - Brynat and Ackermann Reduction (Left as reading)

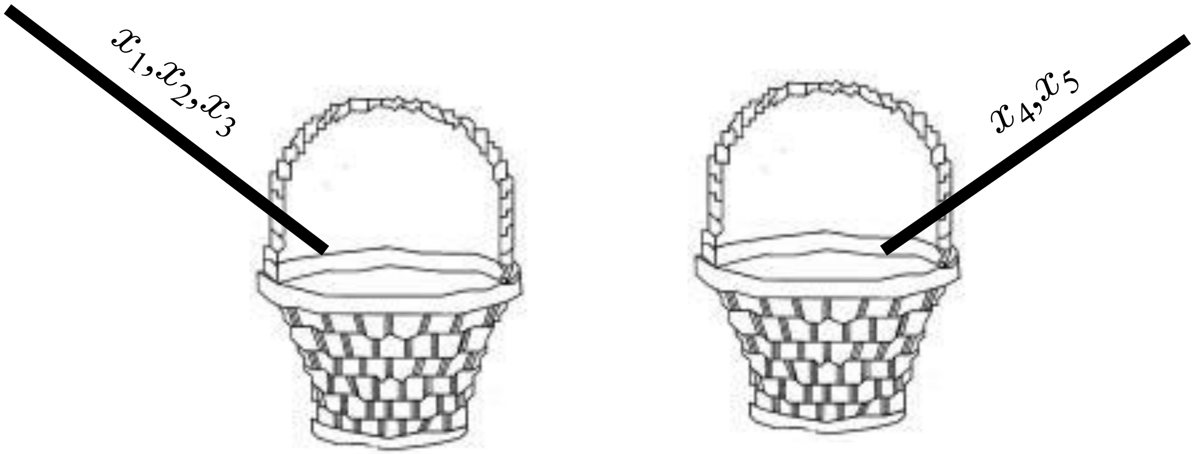
First: conjunction of equalities

- Input: A conjunction of equalities and disequalities

1. Define an equivalence class for each variable. For each equality $x = y$ unite the equivalence classes of x and y. Repeat until convergence.
2. For each disequality $u \neq v$ if u is in the same equivalence class as v return 'UNSAT'.
3. Return 'SAT'.

Example

- $x_1 = x_2 \wedge x_2 = x_3 \wedge x_4 = x_5 \wedge x_5 \neq x_1$

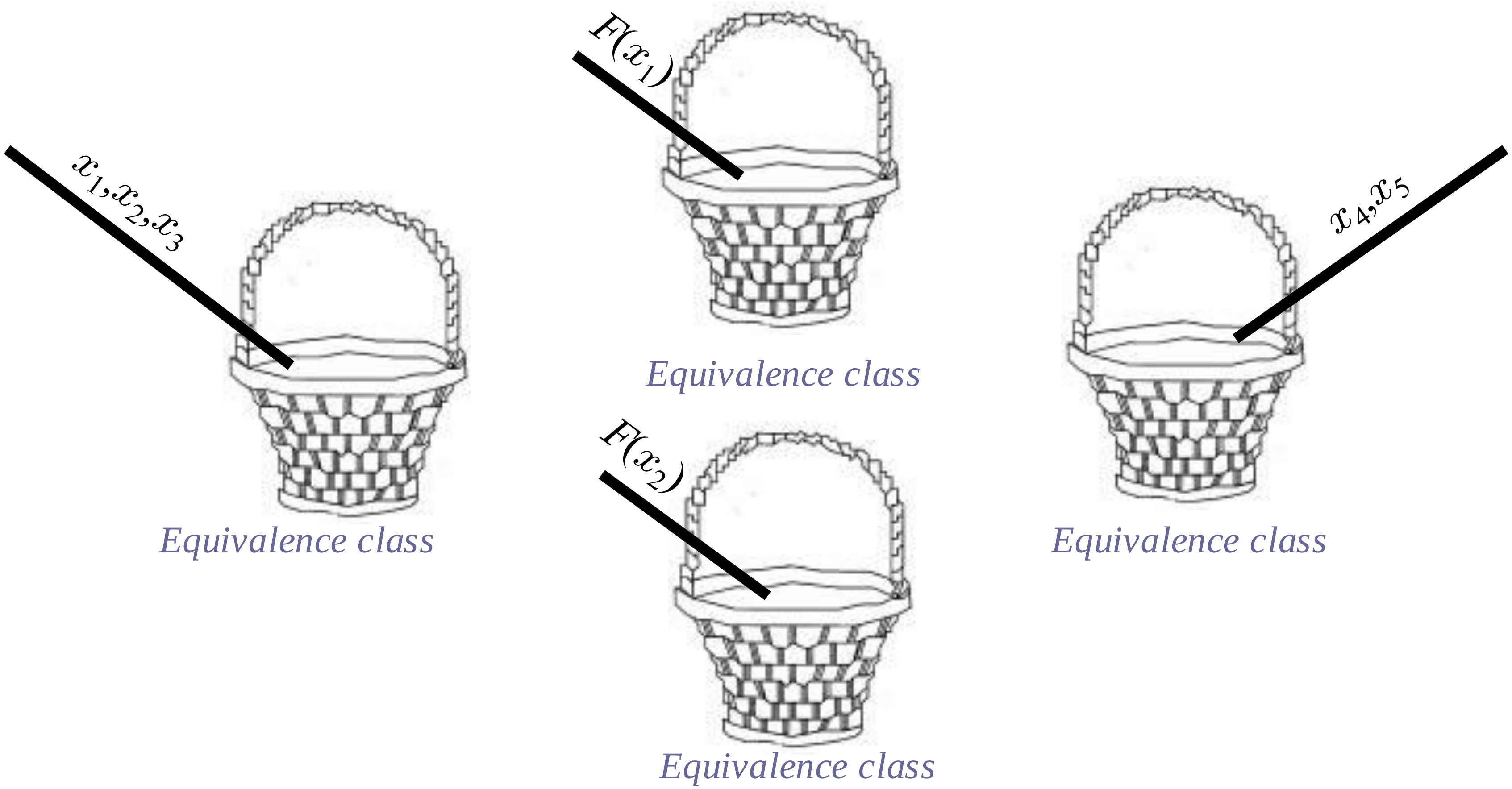


Equivalence class *Equivalence class*

Is there a disequality between members of the same class ?

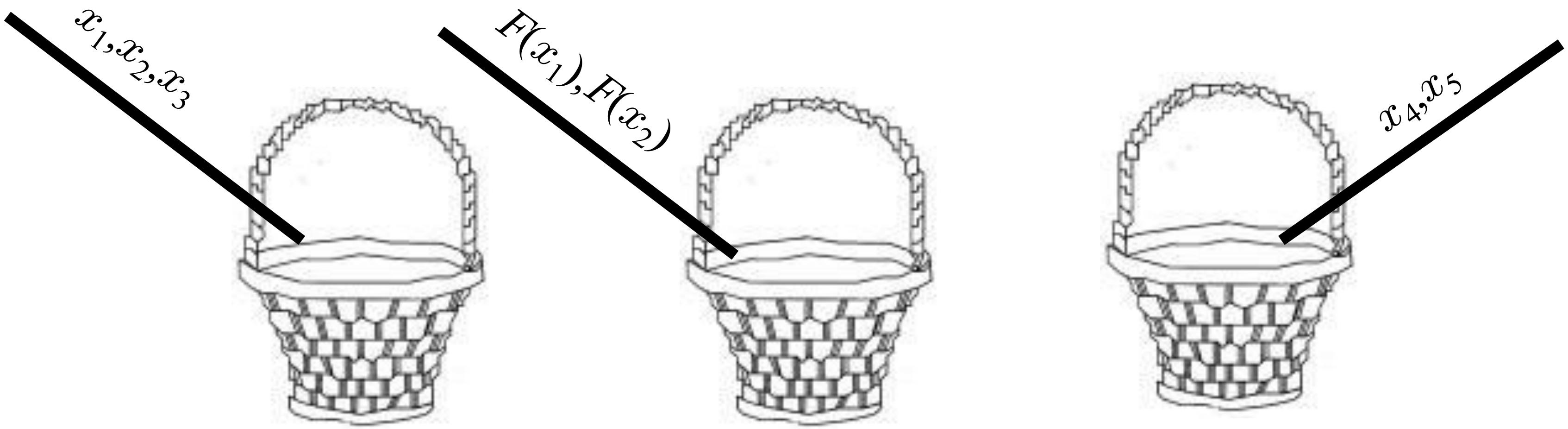
Next: add Uninterpreted Functions

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



Next: Compute the *Congruence Closure*

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



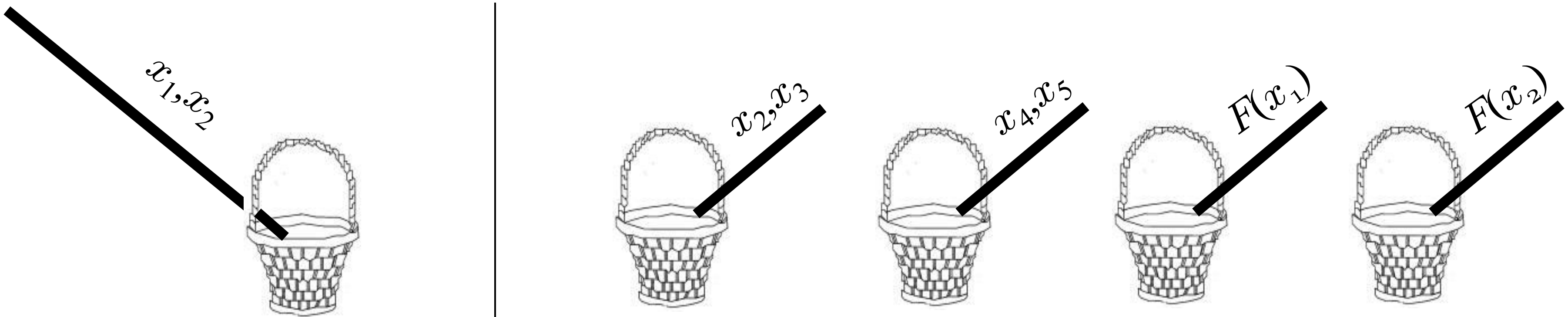
Equivalence class *Equivalence class*

Now - is there a disequality between members of the same class ?

This is called the Congruence Closure

And now: consider a Boolean structure

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



Equivalence class

Equivalence classes

case 1

case 2

Syntactic case splitting: this is what we want to avoid!

Deciding Equality Logic with UFs

- Input: Equality Logic formula ϕ^{UF}
- Convert ϕ^{UF} to DNF
- For each clause:
  - Define an equivalence class for each variable and each function instance.
  - For each equality $x = y$ unite the equivalence classes of x and y. For each function symbol F, unite the classes of $F(x)$ and $F(y)$. Repeat until convergence.
  - If all disequalities are between terms from different equivalence classes, return 'SAT'.
- Return 'UNSAT'.

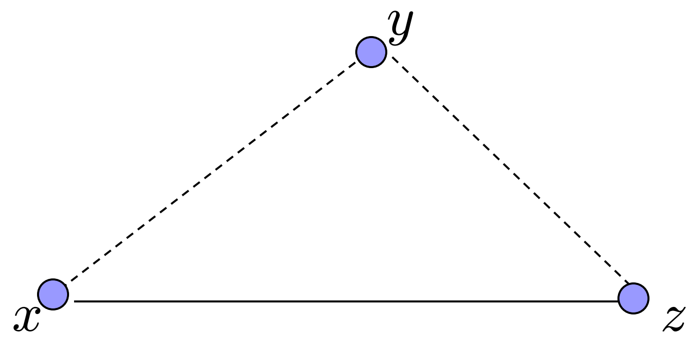
Basic notions

$$\phi^E: \; x = y \wedge y = z \wedge z \neq x$$

- The Equality predicates: $\{x = y, y = z, z \neq x\}$ which we can break to two sets:

  $E_= = \{x = y, y = z\}$, $\qquad E_{\neq} = \{z \neq x\}$

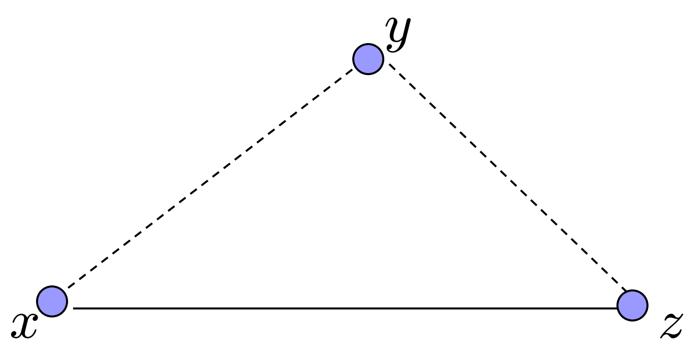
- The Equality Graph $G^E(\phi^E) = \langle V, E_=, E_{\neq} \rangle$ (a.k.a "E-graph")



Basic notions

ϕ_1^E: $x = y \wedge y = z \wedge z \neq x$ *unsatisfiable*

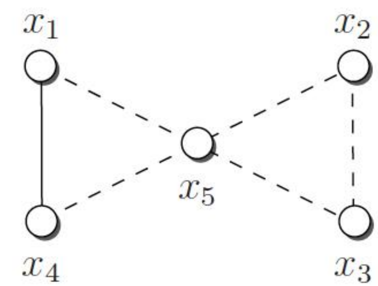
ϕ_2^E: $x = y \wedge y = z \vee z \neq x$ *satisfiable*

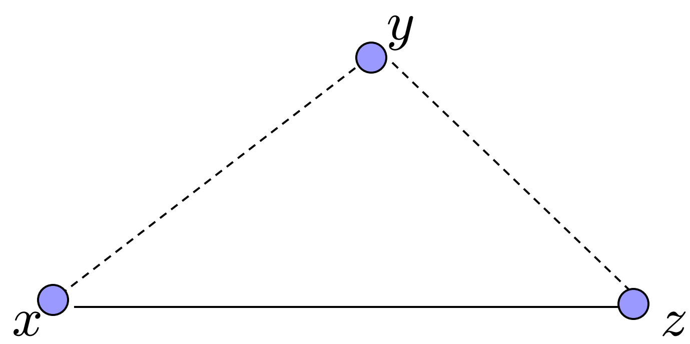


The graph $G^E(\phi^E)$ represents an abstraction of ϕ^E

It ignores the Boolean structure of ϕ^E

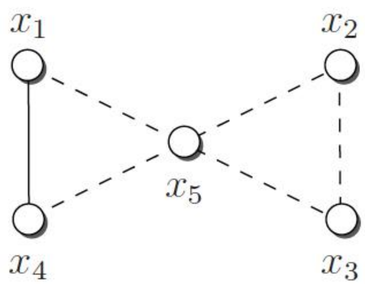
Basic notions





- *Dfn:* a path made of $E_=$ edges is an *Equality Path*. we write $x =^* z$.
- *Dfn:* a path made of $E_=$ edges + exactly one edge from $E_{\neq}$ is a *Disequality Path*. We write $x \neq^* y$.
- What do they convey?

Basic notions



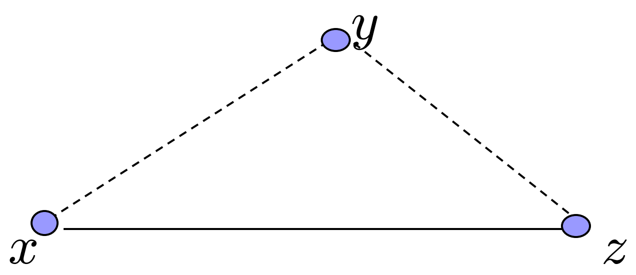
X2 =* x4 (due to x2, x5, x4)

X2 ≠* x4 (due to x2, x5, x1, x4)

What we know

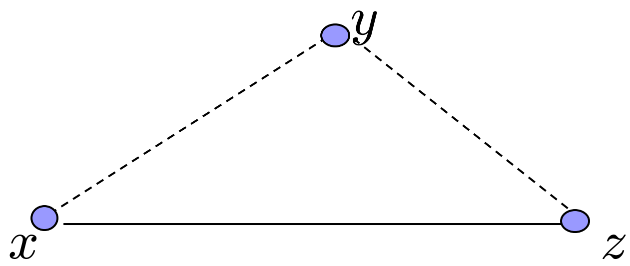
x =* y <=> \exists \phi. Whose equality graph is G and in any assignment satisfying them, x = y.

Basic notions



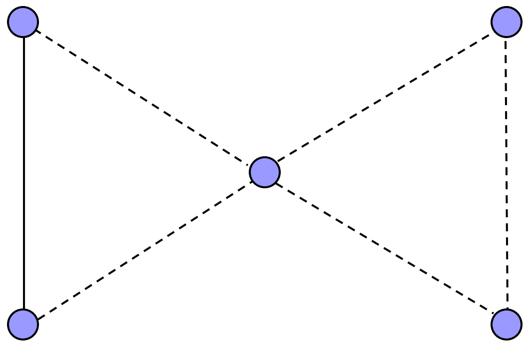
- Dfn. *A cycle with one disequality edge is a* Contradictory Cycle.
- In a Contradictory Cycle, for every two nodes x,y it holds that $x =^* y$ and $x \neq^* y$.

Basic notions



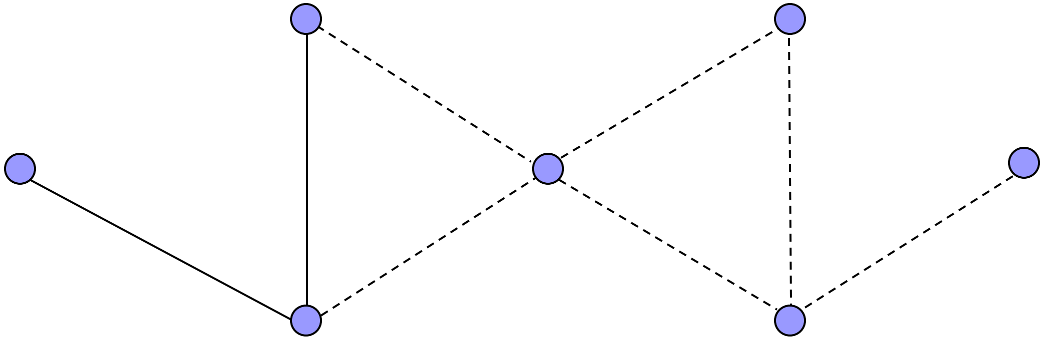
- Dfn: *A subgraph is called satisfiable iff the conjunction of the predicates represented by its edges is satisfiable.*
- Thm: *A subgraph is unsatisfiable iff it contains a Contradictory cycle*
- *Hint: Conjunction of formulas in a c-cyle is unsat.*

Basic notions



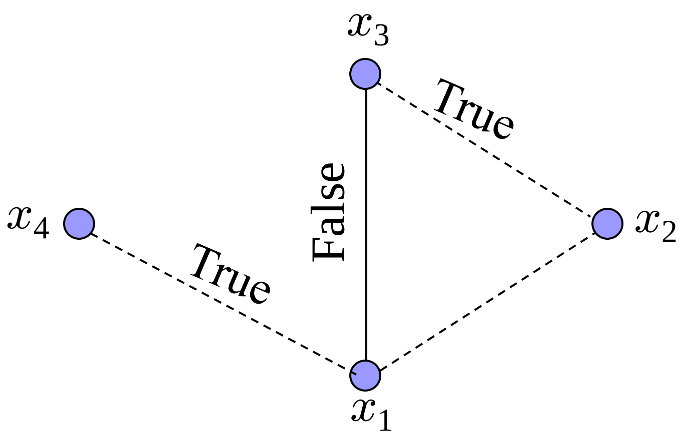
- *Thm: Every Contradictory Cycle is either simple or contains a simple contradictory cycle*

Simplifications, again



- Let S be the set of edges that are not part of any Contradictory Cycle
- Thm: *replacing all solid edges in* S *with False, and all dashed edges in* S *with True, preserves satisfiability*

Simplification: example

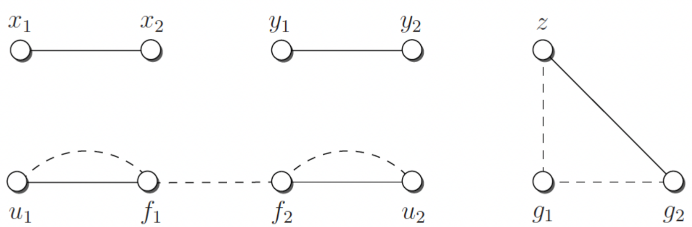


- $(x_1 = x_2 \vee x_1 = x_4) \wedge$
  $(x_1 \neq x_3 \vee x_2 = x_3)$
- $\cancel{(x_1 = x_2 \vee \text{True}) \wedge}$
  $(x_1 \neq x_3 \vee \mathrm{x}_2 = \mathrm{x}_3)$
- $(\neg\text{False} \vee \text{True}) = \text{True}$
- Satisfiable!

Example

$$\neg\varphi^{\mathrm{E}} := \begin{pmatrix} (x_1 \neq x_2 \vee y_1 \neq y_2 \vee f_1 = f_2) \wedge \\ (u_1 \neq f_1 \vee u_2 \neq f_2 \vee g_1 = g_2) \wedge \\ (u_1 = f_1 \wedge u_2 = f_2 \wedge z = g_1) \end{pmatrix} \wedge \ z \neq g_2 \,.$$

$$\begin{aligned} E_= &:= \{(f_1 = f_2),\ (g_1 = g_2),\ (u_1 = f_1),\ (u_2 = f_2),\ (z = g_1)\} \\ E_{\neq} &:= \{(x_1 \neq x_2),\ (y_1 \neq y_2),\ (u_1 \neq f_1),\ (u_2 \neq f_2),\ (z \neq g_2)\} \,. \end{aligned}$$



$$\varphi^{\mathrm{E}'} := \begin{pmatrix} (\mathrm{TRUE} \quad \vee \mathrm{TRUE} \quad \vee \mathrm{TRUE}) \quad \wedge \\ (u_1 \neq f_1 \vee u_2 \neq f_2 \vee g_1 = g_2) \wedge \\ (u_1 = f_1 \wedge u_2 = f_2 \wedge z = g_1 \quad \wedge z \neq g_2) \end{pmatrix}$$

Syntactic vs. Semantic splits

- So far we saw how to handle disjunctions through syntactic case-splitting.
- There are much better ways to do it than simply transforming it to DNF:
  - Semantic Tableaux,
  - SAT-based splitting,
  - others…
- We will investigate some of these methods later in the course.

Syntactic vs. Semantic splits

- Now we start looking at methods that split the search space instead. This is called *semantic splitting*.

- SAT is a very good engine for performing semantic splitting, due to its ability to guide the search, prune the search-space etc.