Last time we proved Thm 12.5 (Gödel’s Completeness Theorem): A first-order sentence, $\gamma$, is valid iff it’s negation, $\varphi$, is unsatisfiable. This in turn holds iff $\varphi$’s Skolemization, $\varphi_S$ is unsatisfiable. This is true iff $E(\varphi_S)$ – the set of all possible substitutions of closed terms into the variables of $\varphi_S$ – is unsatisfiable. Finally, this holds iff there is a resolution proof of $\square$ from assumptions $E(\varphi_S)$.

Today we will show a more efficient and easy-to-program implementation of FO resolution via unification which is intuitively, a lazy substitution into the universally quantified variables. We are given a universal sentence whose quantifier-free part is in CNF, $\varphi = \forall x_1...x_n(C_1(x) \land C_2(x)\ldots \land C_m(x))$. In order to perform a resolution step, we have find an occurrences of a literal and its negation in substitutions into two clauses.

**Definition 13.1** [Unification, Most General Unifier] A substitution, $s$, unifies a set of atoms, $\{A_1, \ldots, A_k\}$, iff $A_1 s = A_2 s = \cdots = A_k s$. If $g$ is a unifier of a set of atoms, such that for any other unifier, $s$ of these atoms, there is a substitution, $s'$, such that $s = gs'$, then $g$ is called the most general unifier (mgu).

**Example 13.2** Let $\varphi_0 = \{C_1, C_2\}$ where $C_1 = \{\neg P(x), \neg P(f(w)), Q(y)\}$, $C_2 = \{P(z)\}$ be a set of two (universally quantified) clauses.

The substitution, $s = [x/z]$, unifies $\{P(x), P(z)\}$. Thus, we can derive the clause

$$C_3 = \{\neg P(f(w)), Q(y)\} = \text{Res}\{\neg P(x), \neg P(f(w)), Q(y)\}, \{P(x)\} = \text{Res}(C_1 s, C_2 s)$$

Furthermore, the substitution, $t = [x/z, f(w)/x]$, unifies the set $\{P(x), P(f(w)), P(z)\}$. Thus, in one step, we can derive

$$C_4 = \{Q(y)\} = \text{Res}\{\neg P(f(w)), Q(y)\}, \{P(f(w))\} = \text{Res}(C_1 t, C_2 t)$$

In fact $s$ and $t$ are mgu’s.

**Algorithm 13.3** [Unification Algorithm]

**Input:** Set of atoms $S = \{A_1, A_2, \ldots, A_r\}$ to be unified.

**Output:** Most General Unifier (mgu) for $S$ if it exists, else ”not unifiable”

1. $s := \emptyset$ # empty substitution
2. while ($|S| > 1$) :
   3. scan each atom from left to right until first difference
   4. if none of these symbols is a variable: return(”not unifiable”)
   5. Let $x$ be the variable and $t \neq x$ be a term starting at same point
   6. if $x$ occurs in $t$: return(”not unifiable”)
   7. $s := s[t/x]$
8. return(”mgu is ”, s)
**Example 13.4** [Running Unification Algorithm on Example 13.2]

First Iteration: \( S = \{ P(x), P(f(w)), P(z) \} \) Start scanning each character, the first difference is at the 3rd character: \( x, f, \) and \( z \).

Choose variable \( x \) and term \( f(w) \): \( s := [f(w)/x] \)

Second Iteration: \( S = \{ P(f(w)), P(z) \} \) Still at 3rd character, choose variable \( z \) and term \( f(w) \): \( s := [f(w)/x, f(w)/z] \)

return(mgu is \([f(w)/x, f(w)/z]\)) □

**Proposition 13.5** If the set of atoms, \( S \), is unifiable, then the Unification Algorithm returns the mgu, otherwise it returns "not unifiable"

**Running time:** In the worst case can be exponential because we can have weird nested repeated substitutions (example in text).

**Good news:** With the right data structure, where atoms are represented with a DAG (circuit), the algorithm is linear because we progress by one character at each step and each substitution is constant time.

We now have a useful, complete proof system for FO Logic. In the following algorithm, **rename variables** if necessary so that no variable occurs in more than one clause. This will allow also possible unifications to be found.

**Algorithm 13.6** [Complete FO Resolution Algorithm]

**Input:** \( \psi \in L(\Sigma) \)

**Output:** if \( \psi \in \text{FO-VALID} \) then a proof of this fact; otherwise, algorithm might never halt

1. \( \psi' := \text{universal closure of } \psi \)
2. \( \varphi := \neg \psi' \)
3. \( \varphi' := \text{formula equivalent to } \varphi \text{ but in RPF, with quantifier-free part in CNF} \)
4. \( \varphi_S := \forall x_1 \ldots x_k(C_1 \land \cdots \land C_m), \text{ the Skolemization of } \varphi' \)
5. \# **Note:** \( \varphi_S \in \text{FO-UNSAT} \iff \psi \in \text{FO-VALID} \)
6. \( \text{ClauseSet} := \{C_1, \ldots, C_m\} \)
7. \textbf{while} (\( \Box \notin \text{ClauseSet} \)):
   8. \textbf{use Algorithm 13.3 to apply resolution, adding resulting clauses to ClauseSet}
9. return("\( \psi \in \text{FO-VALID} \)"