Procedural Semantics

Soundness of SLD-Resolution
Properties of Substitution

Propositions:
Let $\theta$, $\rho$, $\gamma$ be substitutions, $E$ an expression.

- $\theta \varepsilon = \varepsilon \theta = \theta$ (Identity)
- $(E\theta)\rho = E(\theta\rho)$
- $(\theta\rho)\gamma = \theta(\rho\gamma)$ (Associativity)

Proof:
Follows from definition of $\varepsilon$
prove proposition for $E=x$
prove $E(\theta\rho)\gamma = E\theta(\rho\gamma)$ for $E=x$ and 2.

Example
Definition:
Let S be a finite set of simple expressions. A Substitution $\theta$ is called a unifier for S, if $S\theta$ is a singleton.
A unifier $\theta$ is called a most general unifier (mgu) for S if, for each unifier $\rho$ of S there exists a substitution $\gamma$ such that $\rho = \theta \gamma$.

Example

Note: If there exist two mgu's then they are variants.
**Definition:**
Let S be a finite set of simple expressions. Locate the leftmost symbol position at which not all expressions in S have the same symbol and extract from each expression in S the subexpression beginning at that symbol position. The set of all such subexpressions is the *disagreement set*.

**Example:**
Let $S = \{p(f(x), h(y), a), p(f(x), z, a), p(f(x), h(y), b)\}$, then the disagreement set is $\{h(y), z\}$.
Unification Algorithm

1. put \( k := 0 \) and \( \rho_0 := \varepsilon \)

2. \underline{If} \( S_{\rho_k} \) is a singleton, \underline{Then} return(\( \rho_k \))
   \underline{Else} find the disagreement set \( D_k \) of \( S_{\rho_k} \)

3. \underline{If} there exist a variable \( v \) and a term \( t \) in \( D_k \) such that \( v \)
   does not occur in \( t \),
   \underline{// non-deterministic choice}
   \underline{Then} put \( \rho_{k+1} := \rho_k \{ v/t \} \), \( k++ \), \underline{goto} 2
   \underline{Else} exit \underline{// S is not unifiable}
$\rho_0 = \varepsilon$, $k=1$

$S_{\rho_0} = \{\text{even}(0), \text{even}(y)\}$

$D_0 = \{0, y\}$

choose variable $y$, term 0

put $\rho_1 := \varepsilon\{y/0\}$, $k=1$

$S_{\rho_1} = \{\text{even}(0)\}$

return.
Theorem:
Let $S$ be a finite set of simple expressions. If $S$ is unifiable, then the unification algorithm terminates and gives a mgu for $s$. If $S$ is not unifiable, then the unification algorithm terminates and reports this fact.

Proof Sketch:
Assume $\theta$ is a unifier for $S$. Show that until termination for all $k$ :

$$\theta = \rho_k \gamma_k$$
Unification Theorem

Proof Sketch:
Assume $\theta$ is a unifier for $S$. Show that until termination for all $k$:

$\theta = \rho_k \gamma_k$

Induction start: $\rho_0 = \varepsilon$, $\gamma_0 = \theta$

From $k$ to $k+1$ (we only need to consider $S_{\rho_k}$, because otherwise, we are done):

$|S\theta| = 1 \Rightarrow |D_{\gamma_k}| = 1$

Pick a variable $v$ and a term $t$, then:

- $v_{\gamma_k} = t_{\gamma_k}$
- $\rho_{k+1} = \rho_k \{v/t\}$
- $\gamma_{k+1} = \gamma_k \setminus \{v/v_{\gamma_k}\}$, i.e. if $v$ is bound in $\gamma_k$ then
  - $\gamma_k = \{v/v_{\gamma_k}\} \cup \gamma_{k+1}$
  - $\theta = \rho_k \gamma_k = \rho_k \{v/t\} \gamma_{k+1} = \rho_{k+1} \gamma_{k+1}$
SLD-Resolution

- SLD: SL-resolution for definite clauses
- SL: Linear resolution with selection function
Definition:

Let $G$ be $\leftarrow A_1, \ldots, A_m, \ldots, A_k$ and $C$ be $A \leftarrow B_1, \ldots, B_q$.

Then $G'$ is derived from $G$ and $C$ using mgu $\theta$, if:

a. $A_m$ is an Atom, called the selected atom, in $G$

b. $\theta$ is an mgu of $A_m$ and $A$.

c. $G'$ is the goal $\leftarrow (A_1, \ldots, B_1, \ldots, B_q, \ldots, A_k)\theta$.

In resolution terminology $G'$ is called a resolvent of $G$ and $C$. 
**SLD-Derivation**

**Definition:**

Let $P$ be a definite program and $G_0$ a definite goal. An *SLD-Derivation* of $P \cup \{G_0\}$ consists of a (finite or infinite) sequence $G_0, G_1, G_2, \ldots$ of goals, a sequence $C_1, C_2, \ldots$ of variants of program clauses of $P$ and a sequence $\theta_1, \theta_2, \ldots$ of mgu's such that each $G_{i+1}$ is derived from $G_i$ and $C_{i+1}$ using $\theta_{i+1}$.

**standardising apart the variables:**

subscribe all variables in $C_i$ with $i$.

Otherwise $\leftarrow p(x).$ could not be unified with $p(f(x)) \leftarrow$.

each program clause variant $C_1, C_2, \ldots$ is called an *input clause* of the derivation.
SLD-Derivation visualised

\[ (\leftarrow A_1, \ldots, B_1, \ldots, B_q, \ldots, A_k) \theta_1 \]

\[ \leftarrow A_1, \ldots, A_m, \ldots, A_k \]

\[ (\leftarrow A_1 \theta_1, \ldots, B_1 \theta_1, \ldots, D_1, \ldots, D_l, \ldots, B_q \theta_1, \ldots, A_k \theta_1) \theta_2 \]

\[ G_0 \rightarrow G_1 \rightarrow G_2 \rightarrow G_3 \rightarrow \cdots \rightarrow G_{n-1} \rightarrow G_n \]

\[ C_1, \theta_1 \quad C_2, \theta_2 \quad C_3, \theta_3 \quad C_n, \theta_n \]

\[ A \leftarrow B_1, \ldots, B_q. \]
\[ \theta_1 = \text{mgu}(A, A_m). \]

\[ B \leftarrow D_1, \ldots, D_l. \]
\[ \theta_2 = \text{mgu}(B, B_0 \theta_1). \]
Example – Restricted SLD-Refutation

Program P
1 Q(x) :- R(g(x)).
2 R(y).

Goal: Q(f(z)).

\[ \theta_1 = \text{mgu}(Q(x), Q(f(z))) = \{x/f(z)\} \]

\[ \theta_2 = \text{mgu}(R(y), R(g(f(z)))) = \{y/g(f(z))\} \]

Computed Answer \{x/f(z), y/g(f(z))\} restricted to variables of Q(f(z)) results in \(\varepsilon\)
Example – Unrestricted SLD-Refutation

unrestricted → unifiers need not be mgu’s

Program P
1 \( Q(x) \leftarrow R(g(x)) \).
2 \( R(y) \).

Goal: \( Q(f(z)) \).

\( Q(x) \leftarrow R(g(x)) \).
\( \theta_1 = \{x/f(a), z/a\} \)

\( R(y) \leftarrow \).
\( \theta_2 = \{y/g(f(a))\} \)

\( \leftarrow Q(f(z)) \)

\( \leftarrow R(g(f(a))). \)

Correct Answer:
\( \{x/f(a), z/a, y/g(f(z))\} \)
restricted to variables of \( Q(f(z)) \) results in \( \{z/a\} \)
**Definition:**

An *SLD-refutation* of \( P \cup \{G\} \) is a finite SLD-derivation of \( P \cup \{G\} \), which has \( \square \) as the last goal in the derivation. If \( G_n = \square \), we say the refutation has *length* \( n \).

SLD-derivations can be *finite* or *infinite*.

A finite SLD-derivation can be *successful* or *fail*.

An SLD-derivation is successful, if it ends in \( \square \).

An SLD-derivation is *failed*, if it ends in a non-empty goal, which cannot be unified with the head of a program clause.
Definition:
Let P be a definite program. The *success set* of P is the set of all \( A \in B_P \) such that \( P \cup \{ \leftarrow A \} \) has an SLD-refutation.

Procedural Counterpart of the Least Herbrand Model!
Definition:
Let $P$ be a definite program and $G$ a definite goal. Let $	heta_1 \ldots \theta_n$ be the sequence of mgu's used in an SLD-refutation of $P \cup \{G\}$.

A *computed answer* $\theta$ for $P \cup \{G\}$ is the substitution obtained by restricting the composition $\theta_1 \ldots \theta_n$ to the variables of $G$. 
Example: $P=$ Slowsort

**goal:**

\[
\leftarrow \text{sort}(17.22.6.5.\text{nil},y)
\]

**computed answer:**

\[
\{y/5.6.17.22.\text{nil}\}
\]

\[
\begin{align*}
\text{sort}(x,y) & \leftarrow \text{sorted}(y), \text{perm}(x,y) \\
\text{sorted}(\text{nil}) & \leftarrow \\
\text{sorted}(x.\text{nil}) & \leftarrow \\
\text{sorted}(x.y.z) & \leftarrow x \leq y, \text{sorted}(y.z) \\
\text{perm}(\text{nil},\text{nil}) & \leftarrow \\
\text{perm}(x.y.u.v) & \leftarrow \text{delete}(u,x.y,z),\text{perm}(z,v) \\
\text{delete}(x,x.y,y) & \leftarrow \\
\text{delete}(x,y.z,y.w) & \leftarrow \text{delete}(x,z,w) \\
0 \leq x & \leftarrow \\
f(x) \leq f(y) & \leftarrow x \leq y.
\end{align*}
\]
Soundness of SLD-Resolution - 1

Theorem

Let P be a definite program and G a definite goal. Then every computed answer for $P \cup \{G\}$ is a correct answer for $P \cup \{G\}$.

Proof

Let G be the goal $\leftarrow A_1, \ldots, A_k$ and $\theta_1 \ldots \theta_n$ the sequence of mgu's in a refutation of $P \cup \{G\}$.

Show that $\forall ((A_1, \ldots, A_k) \theta_1 \ldots \theta_n)$ is a logical consequence of P using induction (starting at the last goal) over the length of the derivation.
Corollary

The success set of a definite program is contained in its least Herbrand model.

Proof

Let the program be $P$, let $A \in B_P$ and suppose $P \cup \{\leftarrow A\}$ has a refutation. By the theorem on the prior slide, $A$ is a logical consequence of $P$. Thus $A$ is in the least Herbrand model of $P$. 
• **strengthen this corollary**
  If $A \in B_P$ and $P \cup \{ \leftarrow A \}$ has a refutation of length $n$, then $A \in T_P \uparrow n$.

• **Notation**
  $[A] = \{ A' \in B_P : A' = A\theta \text{ for some substitution } \theta \}$
Completeness of SLD-Resolution

- Not treated this year, check out

- http://isweb.uni-koblenz.de/Teaching/SS08/adm08