Equational Logic

Consider a first order language \( \mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V}) \).

We consider the following precedence hierarchy:

\[
\{\forall, \exists\} > \neg > \wedge > \vee > \{\leftarrow, \rightarrow\} \leftrightarrow .
\]

\( \approx /2 \) binary predicate symbol written infix.

Equation \( s \approx t \).

Equational system \( \mathcal{E} \) set of universally closed equations.

\[
\mathcal{E} = \{ \begin{array}{l}
(\forall X, Y, Z) (X \cdot Y) \cdot Z \approx X \cdot (Y \cdot Z), \\
(\forall X) 1 \cdot X \approx X, \\
(\forall X) X \cdot 1 \approx X, \\
(\forall X) X^{-1} \cdot X \approx 1, \\
(\forall X) X \cdot X^{-1} \approx 1
\end{array} \} \]

where \( \cdot/2, ^{-1}/1, 1/0 \in \mathcal{F} \).
Axioms of Equality

\[ \mathcal{E}_{\approx} = \]
\[
\{ (\forall X) X \approx X, \quad \text{reflexivity} \\
(\forall X, Y) (X \approx Y \rightarrow Y \approx X), \quad \text{symmetry} \\
(\forall X, Y, Z) (X \approx Y \land Y \approx Z \rightarrow X \approx Z) \}
\]
\[ \cup \]
\[
\{ (\forall (\wedge_{i=1}^{n} X_i \approx Y_i \rightarrow f(X_1, \ldots, X_n) \approx f(Y_1, \ldots, Y_n)) \mid f/n \in \mathcal{F}) \} \quad \text{f-substitutivity} \\
\cup \]
\[
\{ (\forall (\wedge_{i=1}^{n} X_i \approx Y_i \land p(X_1, \ldots, X_n) \rightarrow p(Y_1, \ldots, Y_n)) \mid p/n \in \mathcal{R}) \} \quad \text{r-substitutivity} 
\]
Equality and Logical Consequence

- \( \mathcal{E} \cup \mathcal{E} \approx \models (\exists X) \; X \cdot a \approx 1? \)
- \( \mathcal{E} \cup \mathcal{E} \approx \cup \{ (\forall X) \; X \cdot X \approx 1 \} \models (\forall X, Y) \; X \cdot Y \approx Y \cdot X? \)
- **Apply resolution**: \( 10^{21} \) resolution steps.
- **Problem**: \( \mathcal{E} \cup \mathcal{E} \approx \) causes large search space.
- **Idea**: Remove troublesome formulas and built them into the deductive machinery.
- **Two possibilities**:
  - Additional rule of inference: paramodulation,
  - Built equational theory into unification computation.
- \( \mathcal{E} \cup \mathcal{E} \approx \) can be written as a set of definite clauses.
- There exists a least model.
  - **Least congruence relation**: \( s \approx_{\mathcal{E}} t \) iff \( \mathcal{E} \cup \mathcal{E} \approx \models \forall s \approx t. \)
Paramodulation

- $L[\pi]$ term occurring at position $\pi \in \mathcal{P}(L)$ in literal $L$;
- $L[\pi \mapsto t]$ Literal $L$ where subterm at $\pi \in \mathcal{P}(L)$ has been replaced by $t$.

- **Paramodulation:**

\[
\frac{\{L_1, \ldots, L_n\} \quad \{l \approx r, K_1, \ldots, K_m\}}{\{L_1[\pi \mapsto r], L_2, \ldots, L_n, K_1, \ldots, K_m\} \theta} \quad \theta = \text{mgu}(L_1[\pi], l), \; \pi \in \mathcal{P}(L_1)
\]

- **Notation:** $\neg s \approx t \not\Rightarrow s \not\approx t$.

- **Remember:** $\mathcal{E} \cup \mathcal{E}_\approx \models \forall s \approx t$ iff $\bigwedge_{s, t} \varphi \to \forall s \approx t$ is valid
  iff $\neg (\bigwedge_{s, t} \varphi \to \forall s \approx t)$ is unsatisfiable
  iff $\mathcal{E} \cup \mathcal{E}_\approx \cup \{\neg \forall s \approx t\}$ is unsatisfiable
  iff $\mathcal{E} \cup \mathcal{E}_\approx \cup \{\exists s \not\approx t\}$ is unsatisfiable
  iff $\mathcal{E} \cup \mathcal{E}_\approx \cup \{\exists s \not\approx t\}$ is unsatisfiable.

- **Theorem 4.1:** If $\mathcal{E} \cup \mathcal{E}_\approx \cup \{\exists s \not\approx t\}$ is unsatisfiable,
  then there is a refutation of $\mathcal{E} \cup \{(\forall X) \; X \approx X, \; \exists s \not\approx t\}$
  with respect to paramodulation, resolution and factoring.
An Example

\[ \mathcal{E} \cup \{ (\forall X) X \approx X, (\forall X) X \cdot X \approx 1 \} \models (\forall X, Y) X \cdot Y \approx Y \cdot X \]

1. \( a \cdot b \not\approx b \cdot a \)  
   initial query

2. \( 1 \cdot X_1 \approx X_1 \)  
   left unit

3. \( X_2 \approx X_2 \)  
   reflexivity

4. \( X_1 \approx 1 \cdot X_1 \)  
   pm(2,3)

5. \( a \cdot b \not\approx (1 \cdot b) \cdot a \)  
   pm(1,4)

6. \( X_3 \cdot X_3 \approx 1 \)  
   hypothesis

7. \( X_4 \approx X_4 \)  
   reflexivity

8. \( 1 \approx X_3 \cdot X_3 \)  
   pm(6,7)

9. \( a \cdot b \not\approx ((X_3 \cdot X_3) \cdot b) \cdot a \)  
   pm(5,8)

\[ a \cdot b \not\approx ((X_3 \cdot X_3) \cdot b) \cdot (a \cdot 1) \]

- \( a \cdot b \not\approx ((X_3 \cdot X_3) \cdot b) \cdot (a \cdot X_4) \)  
  hypothesis

- \( a \cdot b \not\approx (X_3 \cdot ((X_3 \cdot b) \cdot (a \cdot X_4))) \cdot X_4 \)  
  associativity

- \( a \cdot b \not\approx (a \cdot 1) \cdot b \)  
  hypothesis

- \( a \cdot b \not\approx a \cdot b \)  
  right unit

- \( X_5 \approx X_5 \)  
  reflexivity

\[ \text{res } (n), (n') \]
Shorthand Notation

\[ b \cdot a \approx (1 \cdot b) \cdot a \]
\[ \approx ((X_3 \cdot X_3) \cdot b) \cdot a \]
\[ \approx ((X_3 \cdot X_3) \cdot b) \cdot (a \cdot 1) \]
\[ \approx ((X_3 \cdot X_3) \cdot b) \cdot (a \cdot (X_4 \cdot X_4)) \]
\[ \approx (X_3 \cdot ((X_3 \cdot b) \cdot (a \cdot X_4))) \cdot X_4 \]
\[ \approx (a \cdot 1) \cdot b \]
\[ \approx a \cdot b \]

- Search space: \(10^{11}\) steps (instead of \(10^{21}\)).
- There are still many redundant and useless steps.
  - restricted use of equations: term rewriting system.
Term Rewriting Systems

- \( s \approx t \leadsto s \rightarrow t \) and is called rewrite rule.
- Term rewriting system \( \mathcal{R} \) set of rewrite rules.
- \( s[\pi] \) subterm of term \( s \) at position \( \pi \in \mathcal{P}(s) \).
- \( s[\pi \mapsto v] \) term \( s \) where subterm at \( \pi \in \mathcal{P}(s) \) has been replaced by \( v \).
- Rewriting \( s \rightarrow_{\mathcal{R}} t \) iff there are \( l \rightarrow r \in \mathcal{R}, \pi \in \mathcal{P}(s) \) and \( \theta \) such that \( s[\pi] = l\theta \) and \( t = s[\pi \mapsto r\theta] \).

\[
\mathcal{R} = \{ \begin{align*}
append([], X) & \rightarrow X, \\
append([X|Y], Z) & \rightarrow [X|append(Y, Z)], \\
reverse([]) & \rightarrow [], \\
reverse([X|Y]) & \rightarrow append(reverse(Y), [X])
\end{align*} \}
\]

\[
append([1, 2], [3, 4]) \rightarrow_{\mathcal{R}} [1|append([2, [3, 4]])] \\
\rightarrow_{\mathcal{R}} [1, 2|append([], [3, 4])] \\
\rightarrow_{\mathcal{R}} [1, 2, 3, 4].
\]
Term Rewriting and Equational Logic

- $\rightarrow^*_R$ denotes the reflexive and transitive closure of $\rightarrow_R$.
- $s \leftrightarrow_R t$ iff $s \leftarrow_R t$ or $s \rightarrow_R t$.
- $\leftrightarrow^*_R$ is the reflexive and transitive closure of $\leftrightarrow_R$.
- $E_R = \{l \approx r \mid l \rightarrow r \in \mathcal{R}\} \cup E_\approx$.
- Theorem (i) $s \rightarrow^*_R t$ implies $s \approx E_R t$.
  (ii) $s \approx E_R t$ iff $s \leftrightarrow^*_R t$.
- Proof $\rightsquigarrow$ Exercise
- Notation We sometimes omit the subscript $R$.
- Matching problem
  Given terms $u$ and $l$, does there exist a substitution $\theta$ such that $u = l\theta$?
  Such a $\theta$ is called matcher.
Normal Form

- *s* is reducible wrt *R* iff there exists *t* such that \( s \rightarrow^*_R t \); otherwise it is irreducible.

- *t* is a normal form of *s* wrt *R* iff \( s \rightarrow^*_R t \) and *t* irreducible.

- [1, 2, 3, 4] is the normal form of *append*([1, 2], [3, 4]).

- Normal forms are not unique:

\[
\begin{align*}
\{ & \text{not(not(X))} \rightarrow X, \\
& \text{not(or(X, Y))} \rightarrow \text{and(not(X), not(Y))}, \\
& \text{not(and(X, Y))} \rightarrow \text{or(not(X), not(Y))}, \\
& \text{and(X, or(Y, Z))} \rightarrow \text{or(and(X, Y), and(X, Z))}, \\
& \text{and(or(X, Y), Z)} \rightarrow \text{or(and(Y, Z), and(Z, X))} \}
\end{align*}
\]

\( \text{and(or(X, Y), or(U, V))} \) has the normal forms

\[
\begin{align*}
& \text{or(or(and(X, U), and(Y, U)), or(and(X, V), and(Y, V))) and} \\
& \text{or(or(and(Y, U), and(Y, V)), or(and(V, X), and(X, U))}).
\end{align*}
\]
Confluent Term Rewriting Systems

- \( s \downarrow t \) iff there exists \( u \) such that \( s \rightarrow^* u \rightarrow^* t \).
- \( s \uparrow t \) iff there exists \( u \) such that \( s \leftarrow^* u \leftarrow^* t \).
- \( \mathcal{R} \) is confluent iff for all terms \( s \) and \( t \) we find \( s \uparrow t \) implies \( s \downarrow t \).
- \( \mathcal{R} \) is ground confluent iff it is confluent for ground terms.
- \( \mathcal{R} \) is Church-Rosser iff for all terms \( s \) and \( t \) we find \( s \leftrightarrow^* t \) iff \( s \downarrow t \).
- Theorem \( \mathcal{R} \) is Church-Rosser iff \( \mathcal{R} \) is confluent.
- Proof \( \rightsquigarrow \) Exercise.
- Remember \( s \leftrightarrow_{\mathcal{R}}^* t \) iff \( s \approx_{\mathcal{E}_\mathcal{R}} t \).
  - Rewriting has only to be applied in one direction!
Canonical Term Rewriting Systems

- $\mathcal{R}$ is terminating iff it has no infinite rewriting sequences.
  - The question whether $\mathcal{R}$ is terminating is undecidable.
- $\mathcal{R}$ is canonical iff $\mathcal{R}$ confluent and terminating.
  - If $\mathcal{R}$ is canonical then $s \approx_{\mathcal{E}_\mathcal{R}} t$ iff $s \downarrow t$.
  - If $\mathcal{R}$ is canonical then $\mathcal{E}_\mathcal{R}$ is decidable.
Termination

▶ Is a given term rewriting system terminating?
▶ **Idea:** Find a well-founded ordering $\succ /2$ on terms such that $s \rightarrow t$ implies $s \succ t$.
▶ Let $\geq /2$ be a partial ordering on terms.
▶ $s \succ t$ iff $s \geq t$ and $s \neq t$.
▶ $\succ /2$ is well-founded iff there is no infinite sequence $s_1 \succ s_2 \succ \ldots$.
▶ A termination ordering $\succ /2$ is a well-founded, transitive and antisymmetric relation on the set of terms satisfying the following properties:
  ▶ **full invariance property** if $s \succ t$ then $s\theta > t\theta$,
  ▶ **replacement property**
    if $s \succ t$, $\pi \in \mathcal{P}(u)$ and $u[\pi] = s$ then $u > u[\pi \mapsto t]$.
▶ **Theorem 4.2** Let $\mathcal{R}$ be a term rewriting system and $\succ /2$ a termination ordering.
  If for all rules $l \rightarrow r \in \mathcal{R}$ we find that $l \succ r$ then $\mathcal{R}$ is terminating.
  ▶ **Proof** ⇝ **Exercise.**
Termination Orderings: Two Examples

- Let $|s|$ denote the size of the term. $s > t$ iff for all grounding substitutions $\theta$ we find that $|s\theta| > |t\theta|$.
  - $f(X, Y) > g(X)$,
  - $f(X, Y)$ and $h(X, Y)$ can not be ordered.

- Polynomial ordering
  assign to each $f$ a polynomial with coefficients taken from $\mathbb{N}$.
  - $f(X, Y)^{I, Z} = 2X + Y$,
  - $h(X, Y)^{I, Z} = X + Y$.
  $s > t$ iff $s^{I, Z} > t^{I, Z}$.

- There are many other termination orderings!
- $>/2$ is more powerful than $>/2$ iff $s > t$ implies $s >' t$, but not vice versa.
Confluence

► Is a given terminating term rewriting system confluent?

► $\mathcal{R}$ is locally confluent iff for all terms $r$, $s$ and $t$ the following holds:
  If $r \rightarrow s$ and $r \rightarrow t$ then $s \downarrow t$.

► **Theorem 4.3** Let $\mathcal{R}$ be a terminating term rewriting system. $\mathcal{R}$ is confluent iff it is locally confluent.
  
  ▶ **Proof** ~ Exercise.
Local Confluence

- Is a given terminating term rewriting system locally confluent?
- A subterm $u$ of $t$ is called a redex iff there exists $\theta$ and $l \rightarrow r \in \mathcal{R}$ such that $u = l\theta$.
- Let $l_1 \rightarrow r_1 \in \mathcal{R}$ and $l_2 \rightarrow r_2 \in \mathcal{R}$ be applicable to $t$ to two redexes.

▷ Case analysis
  (a) They are disjoint.
  (b) one redex is a subterm of the other one and corresponds to a variable position in the left-hand-side of the other rule.
  (c) one redex is a subterm of the other one but does not correspond to a variable position in the left-hand-side of the other rule (the redexes overlap).
Example: Consider $t = (g(a) \cdot f(b)) \cdot c$

(a) $\mathcal{R} = \{a \to c, b \to c\}$.
   - $a$ and $b$ are disjoint redecls in $t$,
   - ok.

(b) $\mathcal{R} = \{a \to c, g(X) \to f(X)\}$.
   - $a$ and $g(a)$ are redecls in $t$; $a$ corresponds to the variable position in $g(x)$,
   - ok.

(c) $\mathcal{R} = \{(X \cdot Y) \cdot Z \to X, g(a) \cdot f(b) \to c\}$.
   - $(g(a) \cdot f(b)) \cdot c$ and $g(a) \cdot f(b)$ are overlapping redecls in $t$.
   - problematic!
Critical Pairs

- Suppose \( \{ l_1 \rightarrow r_1, l_2 \rightarrow r_2 \} \subseteq \mathcal{R} \) and \( l_2 \) is unifiable with a non-variable subterm \( u \) of \( l_1 \) using mgu \( \theta \). Then the pair

\[
\langle (l_1[u/r_2])\theta, r_1\theta \rangle
\]

is said to be critical. It is obtained by superposing \( l_1 \) and \( l_2 \).

- \( (X \cdot Y) \cdot Z \rightarrow X \) and \( g(a) \cdot f(b) \rightarrow c \) form the critical pair \( \langle c \cdot Z, g(a) \rangle \).

- **Theorem 4.4** A term rewriting system \( \mathcal{R} \) is locally confluent iff for all critical pairs \( \langle s, t \rangle \) of \( \mathcal{R} \) we find \( s \downarrow t \).

- **Proof** \( \sim \) Exercise.
Completion

- Can a terminating and non-confluent $\mathcal{R}$ be turned into a confluent one?
- Two term rewriting systems $\mathcal{R}$ and $\mathcal{R}'$ are equivalent iff $\approx_{\mathcal{R}} = \approx_{\mathcal{R}'}$.
- Idea: if $\langle s, t \rangle$ is a critical pair, then add either $s \rightarrow t$ or $t \rightarrow s$ to $\mathcal{R}$.
  - This is called completion.
  - The equational theory remains unchanged.
Completion Procedure

▶ Given a terminating \( R \) together with a termination ordering \( \succ / 2 \).

1. If for all critical pairs \( \langle s, t \rangle \) of \( R \) we find that \( s \downarrow t \)
   then return “success”; \( R \) is canonical.
2. If \( R \) has a critical pair whose elements do not rewrite to a common term,
   then transform the elements of the critical pair to some normal form.
   Let \( \langle s, t \rangle \) be the normalized critical pair:
   ▶ If \( s > t \) then add the rule \( s \rightarrow t \) to \( R \) and goto 1.
   ▶ If \( t > s \) then add the rule \( t \rightarrow s \) to \( R \) and goto 1.
   ▶ If neither \( s > t \) nor \( t > s \) then return “fail”.

▶ The completion procedure may either succeed or fail or loop.
Completion: An Example

\[ \mathcal{R} = \{ c \rightarrow b, f \rightarrow b, f \rightarrow a, e \rightarrow a, e \rightarrow d \} \]
\[ f > e > d > c > b > a. \]

- **Critical pairs:** \( \langle b, a \rangle \) and \( \langle d, a \rangle \).
- **New rules:** \( b \rightarrow a \) and \( d \rightarrow a \).
- **\( \mathcal{R}' = \{ c \rightarrow b, f \rightarrow b, f \rightarrow a, e \rightarrow a, e \rightarrow d, b \rightarrow a, d \rightarrow a \} \).**
- **\( \mathcal{R}' \) is canonical.**
- **\( s \approx_{\mathcal{R}} t \) iff \( s \approx_{\mathcal{R}'} t \).**
- **All proofs for \( s \approx_{\mathcal{R}}, t \) are in valley form.**
Unification Theory

- **$\mathcal{E}$-unification problem**: $\mathcal{E} \cup \mathcal{E}_\prec \models \exists s \approx t$.
- **$\mathcal{E}$-unifier** $\theta$ is a solution of the $\mathcal{E}$-unification problem iff $s\theta \approx \mathcal{E} t\theta$.
- $\eta$ and $\theta$ are $\mathcal{E}$-equal on set $V$ of variables ($\theta \equiv_\mathcal{E} \eta[V]$) iff $X\eta \equiv_\mathcal{E} X\theta$ for all $X \in V$.
- $\eta$ is an $\mathcal{E}$-instance of $\theta$ on set $V$ of variables ($\theta \succeq_\mathcal{E} \eta[V]$) iff there exists a substitution $\tau$ such that $X\eta \equiv_\mathcal{E} X\theta\tau$ for all $X \in V$.
- $\theta \succ_\mathcal{E} \eta[V]$ iff $\theta \succeq_\mathcal{E} \eta[V]$ and not $\theta \equiv_\mathcal{E} \eta[V]$.
- If neither $\theta \succeq_\mathcal{E} \eta[V]$ nor $\eta \succeq_\mathcal{E} \theta[V]$ then $\theta$ and $\eta$ are said to be *incomparable*. 
Example: $\mathcal{E} \cup \mathcal{E}_\approx \models (\exists X, Y) \ f(X, g(a, b)) \approx f(g(Y, b), X)$

- $\mathcal{E} = \emptyset$
  - Decision problem is decidable.
  - Most general unifier is unique modulo variable renaming:
    $\theta_1 = \{X \mapsto g(a, b), \ Y \mapsto a\}$.
- $\mathcal{E} = \{\forall f(X, Y) \approx f(Y, X)\}$
  - $\theta_1$ is a solution.
  - So is $\theta_2 = \{Y \mapsto a\}$:
    $f(X, g(a, b))\theta_2 = f(X, g(a, b)) \approx_\mathcal{E} f(g(a, b), X) = f(g(Y, b), X)\theta_2$.
  - $\theta_2 \geq_\mathcal{E} \theta_1[\{X, Y\}]$.
  - There are at most finitely many most general unifiers.
Example: $\mathcal{E} \cup \mathcal{E} \approx \models (\exists X, Y) \ f(X, g(a, b)) \approx f(g(Y, b), X)$

$\mathcal{E} = \{ \forall f(X, f(Y, Z)) \approx f(f(X, Y), Z) \}$

$\theta_1 = \{ X \mapsto g(a, b), \ Y \mapsto a \}$ is a solution.

So is $\theta_3 = \{ X \mapsto f(g(a, b), g(a, b)), \ Y \mapsto a \}$:

$$f(X, g(a, b))\theta_3 = f(f(g(a, b), g(a, b)), g(a, b)) \approx_{\mathcal{E}} f(g(a, b), f(g(a, b), g(a, b))) = f(g(Y, b), X)\theta_3.$$

$\theta_1$ and $\theta_3$ are incomparable.

$\theta_4 = \{ X \mapsto f(g(a, b), f(g(a, b), g(a, b))), \ Y \mapsto a \}$ is yet another solution incomparable to $\theta_1$ and $\theta_2$.

In general, there may be infinitely many most general unifiers.

$\mathcal{E} = \{ \forall f(X, f(Y, Z)) \approx f(f(X, Y), Z)), \ \forall f(X, Y) \approx f(Y, X) \}$

There are at most finitely many most general unifiers.
Sets of $E$-Unifiers

- Given an $E$-unification problem $E \cup E \approx ? \models \exists s \approx t$.
- $U_E(s, t)$ denotes the set of all $E$-unifiers of $s$ and $t$.
- Complete set $cU_E(s, t)$ of $E$-unifiers of $s$ and $t$:
  - $cU_E(s, t) \subseteq U_E(s, t)$,
  - for all $\eta \in U_E(s, t)$ there exists $\theta \in cU_E(s, t)$ such that $\theta \geq E \eta[V]$, where $V = \text{VAR}(s) \cup \text{VAR}(t)$.
- Minimal complete set $\mu U_E(s, t)$ of $E$-unifiers for $s$ and $t$:
  - complete set,
  - for all $\theta, \eta \in \mu U_E(s, t)$ we find $\theta \geq E \eta[V]$ implies $\theta = \eta$.
- If $cU_E(s, t)$ is finite and $\geq E$ is decidable then there exists $\mu U_E(s, t)$.
- Let $\theta \equiv E \eta[V]$ iff $\theta \geq E \eta[V]$ and $\eta \geq E \theta[V]$. Then, $\mu U_E(s, t)$ is unique up to $\equiv E [V]$, if it exists.
Another Example

\[ \mathcal{F} = \{a/0, f/2\}, \mathcal{E} = \{\forall f(X, f(Y, Z)) \approx f(f(X, Y), Z)\}, \]

\[ \mathcal{E} \cup \mathcal{E} \approx \models (\exists X) f(X, a) \approx f(a, Y). \]

\[ \theta = \{X \mapsto a, Y \mapsto a\} \text{ is a solution.} \]
\[ \eta = \{X \mapsto f(a, Z), Y \mapsto f(Z, a)\} \text{ is another solution.} \]
\[ \{\theta, \eta\} \text{ is a complete set of } \mathcal{E}\text{-unifiers} \rightsquigarrow \text{Exercise}. \]
\[ \theta \text{ and } \eta \text{ are incomparable under } \geq \mathcal{E}. \]
\[ \{\theta, \eta\} \text{ is minimal.} \]
A Note on Minimal Complete Sets of $\mathcal{E}$-Unifiers

- Consider $\mathcal{E} = \{\forall f(0, X) \approx X, \forall g(f(X, Y)) \approx g(Y)\}$.
- Claim There does not exist $\mu U_\mathcal{E}(g(X), g(0))$.
- Proof Let $\mathcal{R} = \{f(0, X) \rightarrow X, g(f(X, Y)) \rightarrow g(Y)\}$.

  - $\mathcal{R}$ is canonical $\Rightarrow$ Exercise.
  - Define $\sigma_0 = \{X \mapsto 0\}$,
    $\sigma_1 = \{X \mapsto f(X_1, 0)\} = \{X \mapsto f(X_1, X\sigma_0)\}$,
    $\vdots$
    $\sigma_i = \{X \mapsto f(X_i, X\sigma_{i-1})\}$.
  - Let $\mathcal{S} = \{\sigma_i \mid i \geq 0\}$ and $V = \{X\}$
  - $\mathcal{S}$ is a $cU_\mathcal{E}(g(X), g(0)) \Rightarrow$ Exercise.
  - With $\rho_i = \{X_i \mapsto 0\}$ we find $X\sigma_i\rho_i = f(0, X\sigma_{i-1}) \approx_\mathcal{E} X\sigma_{i-1}$.
  - Hence, $\sigma_i \geq_\mathcal{E} \sigma_{i-1}$.
  - Because $X\sigma_i = f(X_i, X\sigma_{i-1}) \not\approx_\mathcal{E} X\sigma_{i-1}$ we find $\sigma_i \not\approx_\mathcal{E} \sigma_{i-1}$.
  - Thus, $\sigma_i > \sigma_{i-1}[V]$. 

Equational Logic (14th December 2007)
A Note on Minimal Complete Sets of $\mathcal{E}$-Unifiers (Continued)

▶ Remember $\mathcal{E} = \{\forall f(0, X) \approx X, \forall g(f(X, Y)) \approx g(Y)\}$.

▷ Assume $S'$ is a $\mu U_{\mathcal{E}}(g(X), g(0))$.

▷ Because $S$ is complete we find that for all $\rho \in S'$ there exists $\sigma_i \in S$ such that $\sigma_i \geq_{\mathcal{E}} \rho[V]$.

▷ Because $\sigma_{i+1} >_{\mathcal{E}} \sigma_i[V]$ we obtain $\sigma_{i+1} >_{\mathcal{E}} \rho[V]$.

▷ Because $S'$ is complete we find that there exists $\sigma \in S'$ such that $\sigma \geq_{\mathcal{E}} \sigma_{i+1}[V]$.

▷ Hence, $\sigma >_{\mathcal{E}} \rho[V]$.

▷ Thus, $S'$ is not minimal $\leadsto$ Contradiction.
Unification Types

- The unification type of $\mathcal{E}$ is
  - unitary iff a set $\mu U_\mathcal{E}(s, t)$ exists for all $s, t$ and has cardinality 0 or 1.
  - finitary iff a set $\mu U_\mathcal{E}(s, t)$ exists for all $s, t$ and is finite.
  - infinitary iff a set $\mu U_\mathcal{E}(s, t)$ exists for all $s, t$, and there are $u$ and $v$ such that $\mu U_\mathcal{E}(u, v)$ is infinite.
  - zero iff there are $s, t$ such that $\mu U_\mathcal{E}(s, t)$ does not exist.
Unification procedures

▶ $\mathcal{E}$-unification procedure:

- input: $s \approx t$.
- output: subset of $U_\mathcal{E}(s, t)$.
- is complete iff for all $s, t$ the output is a $cU_\mathcal{E}(s, t)$.
- is minimal iff for all $s, t$ the output is a $\mu U_\mathcal{E}(s, t)$.

▶ Universal $\mathcal{E}$-unification procedure:

- input: $\mathcal{E}$ and $s \approx t$.
- output: subset of $U_\mathcal{E}(s, t)$.
- is complete iff for all $\mathcal{E}$ and $s, t$ the output is a $cU_\mathcal{E}(s, t)$.
- is minimal iff for all $\mathcal{E}$ and $s, t$ the output is a $\mu U_\mathcal{E}(s, t)$.
Typical Questions related to \( \mathcal{E} \)

- Is it decidable whether an \( \mathcal{E} \)-unification problem is solvable?
- What is the unification type of \( \mathcal{E} \)?
- How can we obtain an efficient \( \mathcal{E} \)-unification algorithm or, preferably, a minimal \( \mathcal{E} \)-unification procedure?
Classes of $\mathcal{E}$-Unification Problems

The class of an $\mathcal{E}$-unification problem $\mathcal{E} \cup \mathcal{E} \approx ? \models \exists s \approx t$ is called

- **elementary** iff $s$ and $t$ contain only symbols occurring in $\mathcal{E}$.
- **with constants** iff $s$ and $t$ may contain additional so-called free constants.
- **general** iff $s$ and $t$ may contain additional free function symbols of arbitrary arity.
### Unification with Constants: Some Examples

<table>
<thead>
<tr>
<th>Equational System</th>
<th>Unification Type</th>
<th>Unification decidable?</th>
<th>Complexity of the decision problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{E}_A$</td>
<td>infinitary</td>
<td>yes</td>
<td>NP-hard</td>
</tr>
<tr>
<td>$\mathcal{E}_C$</td>
<td>finitary</td>
<td>yes</td>
<td>NP-complete</td>
</tr>
<tr>
<td>$\mathcal{E}_{AC}$</td>
<td>finitary</td>
<td>yes</td>
<td>NP-complete</td>
</tr>
<tr>
<td>$\mathcal{E}_{AG}$</td>
<td>unitary</td>
<td>yes</td>
<td>polynomial</td>
</tr>
<tr>
<td>$\mathcal{E}_{AI}$</td>
<td>zero</td>
<td>yes</td>
<td>NP-hard</td>
</tr>
<tr>
<td>$\mathcal{E}_{CR1}$</td>
<td>zero</td>
<td>no</td>
<td>–</td>
</tr>
<tr>
<td>$\mathcal{E}<em>{DL}, \mathcal{E}</em>{DR}$</td>
<td>unitary</td>
<td>yes</td>
<td>polynomial</td>
</tr>
<tr>
<td>$\mathcal{E}_{D}$</td>
<td>infinitary</td>
<td>yes</td>
<td>NP-hard</td>
</tr>
<tr>
<td>$\mathcal{E}_{DA}$</td>
<td>infinitary</td>
<td>no</td>
<td>–</td>
</tr>
<tr>
<td>$\mathcal{E}_{BR}$</td>
<td>unitary</td>
<td>yes</td>
<td>NP-complete</td>
</tr>
</tbody>
</table>
Additional Remarks

- **E-matching problem**: $E \cup E \approx ? \vdash \exists \theta \ s \approx_E t \theta$.

- **Combination problem**: Can the results and unification algorithms for $E_1$ and $E_2$ be combined to $E_1 \cup E_2$?

- **Universal $E$-unification problem**: $E$-unification problem, where the equational system is part of the input.
Canonical Term Rewriting Systems Revisited

- Let $R$ be a canonical term rewriting system.
- So far, we were able to answer questions of the form $E_R \cup E_\approx \models \forall s \approx t$.
  - **Rewriting**: $s \rightarrow_R t$ iff there are $l \rightarrow r \in R$, $\pi \in P(s)$ and $\theta$ such that $s[\pi] = l\theta$ and $t = s[\pi \mapsto r\theta]$.
- Compare narrowing to rewriting and paramodulation!
  - **Narrowing**: $s \Rightarrow_R t$ iff there are $l \rightarrow r \in R$, $\pi \in P(s)$ and $\theta$ such that $s[\pi] \notin \mathcal{V}$, $s[\pi] \theta = l\theta$ and $t = (s[\pi \mapsto r])\theta$.

**Theorem**

Let $R$ be a canonical term rewriting system with $\text{VAR}(l) \supseteq \text{VAR}(r)$ for all $l \rightarrow r \in R$. Then narrowing and resolution is sound and complete.

- A complete universal $E$-unification procedure for canonical theories $E$ can be built upon narrowing and resolution.
Applications

- databases
- information retrieval
- computer vision
- natural language processing
- knowledge based systems
- text manipulation systems
- planning and scheduling systems
- pattern directed programming languages
- logic programming systems
- computer algebra systems
- deduction systems
- non-classical reasoning systems
Multisets

- \( \{e_1, e_2, \ldots \} \), \( \emptyset \).
- \( X \in_k \mathcal{M} \) iff \( X \) occurs precisely \( k \) times in \( \mathcal{M} \).
- \( \mathcal{M}_1 \equiv \mathcal{M}_2 \) iff for all \( X \) we find \( X \in_k \mathcal{M}_1 \) iff \( X \in_k \mathcal{M}_2 \).
- \( X \in_m \mathcal{M}_1 \cup \mathcal{M}_2 \) iff there exist \( k, l \geq 0 \) such that \( X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2 \) and \( k + l = m \).
- \( X \in_m \mathcal{M}_1 \setminus \mathcal{M}_2 \) iff there exist \( k, l \geq 0 \) such that either \( X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2, k > l \) and \( m = k - l \) or \( X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2, k \leq l \) and \( m = 0 \).
- \( X \in_m \mathcal{M}_1 \cap \mathcal{M}_2 \) iff there exist \( k, l \geq 0 \) such that \( X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2 \) and \( m = \min \{k, l\} \).
- \( \mathcal{M}_1 \subseteq \mathcal{M}_2 \) iff \( \mathcal{M}_1 \cap \mathcal{M}_2 \equiv \mathcal{M}_1 \).
Fluent Terms

- Alphabet with variables $\mathcal{V}$ and function symbols $\mathcal{F} \supseteq \{o/2, 1/0\}$.
- $\mathcal{F}^- = \mathcal{F} \setminus \{o/2, 1/0\}$
- $\mathcal{T}(\mathcal{F}^-, \mathcal{V})$: terms built over $\mathcal{V}$ and $\mathcal{F}$ without using $o/2$ and $1/0$.
- Fluents: nonvariable elements of $\mathcal{T}(\mathcal{F}^-, \mathcal{V})$.
- Fluent terms:
  - Each fluent is a fluent term.
  - 1 is a fluent term.
  - If $s$ and $t$ are fluent terms then $s \circ t$ is a fluent term as well.

- $\mathcal{E}_{AC1} = \{ \begin{align*}
(\forall X, Y, Z) \ X \circ (Y \circ Z) & \approx (X \circ Y) \circ Z \\
(\forall X, Y) \ X \circ Y & \approx Y \circ X \\
(\forall X) \ X \circ 1 & \approx X
\end{align*} \}
Multisets vs. Fluent Terms

$I$:

\[ t^I = \begin{cases} 
\emptyset & \text{if } t = 1 \\
\{t\} & \text{if } t \text{ is a fluent} \\
u^I \cup v^I & \text{if } t = u \circ v
\end{cases} \]

$-I$:

\[ M^{-I} = \begin{cases} 
1 & \text{if } M \vdash \emptyset \\
s \circ N^{-I} & \text{if } M \vdash \{s\} \cup N
\end{cases} \]
Matching and Unification Problems

- **Submultiset matching problem:**
  Does there exist a $\theta$ such that $M\theta \subseteq N\theta$, where $N\theta$ is ground?

- **Submultiset unification problem:**
  Does there exist a $\theta$ such that $M\theta \subseteq N\theta$?

- **Fluent matching problem:**
  Does there exist a $\theta$ such that $(s \circ X)\theta \approx_{AC1} t$, where $t$ is ground and $X$ does not occur in $s$?

- **Fluent unification problem:**
  Does there exist a $\theta$ such that $(s \circ X)\theta \approx_{AC1} t\theta$, where $X$ does not occur in $s$ or $t$?
Submultiset vs. Fluent Unification Problems

▶ Equivalence of matching problems:

\[(s \circ X)\theta \approx_{AC1} t \iff (s\theta)^I \subseteq t^I \text{ and } (X\theta)^I = t^I \setminus (s\theta)^I\]

▶ Equivalence of unification problems:

\[(s \circ X)\theta \approx_{AC1} t\theta \iff (s\theta)^I \subseteq (t\theta)^I \text{ and } (X\theta)^I = (t\theta)^I \setminus (s\theta)^I\]

▶ Fluent matching and fluent unification problems are

▷ decidable,
▷ finitary and
▷ there always exists a minimal complete set of matchers and unifiers.
**Fluent Matching Algorithm**

**Input:** A fluent matching problem \( \exists \theta (s \circ X) \theta \approx_{AC1} t? \)
(where \( t \) is ground and \( X \) does not occur in \( s \)).

**Output:** A solution \( \theta \) of the fluent matching problem, if it is solvable; **failure**, otherwise.

1. \( \theta = \varepsilon; \)
2. if \( s \approx_{AC1} t \) then return \( \theta \{ X \mapsto t \} \); 
3. don’t-care non-deterministically select a fluent \( f \) from \( s \) and remove \( f \) from \( s \); 
4. don’t-know non-deterministically select a fluent \( g \) from \( t \) such that there exists a substitution \( \eta \) with \( f\eta = g \); 
5. if such a fluent exists then apply \( \eta \) to \( s \), delete \( g \) from \( t \) and let \( \theta := \theta\eta \), otherwise stop with failure; 
6. goto 2.
States, Actions and Causality

- Rational Agents, Cognitive Robotics.
- **Situation Calculus** (John McCarthy 1963)
- **Core idea** a state is a snapshot of the world and can be changed by actions only.
- **Problem** Each state and each action is only partially known!
General Problems

- **Frame problem:**
  Which fluents are unaffected by the execution of an action?

- **Ramification problem:**
  Which fluents are really present after the execution of an action?

- **Qualification problem:**
  Which preconditions have to be satisfied such that an action is executable?

- **Prediction problem:**
  How long are fluents present in certain situations?

- All problems have a cognitive as well as a technical aspect.
Requirements

▶ (McCarthy 1963):
▶ General properties of causality and facts about the possibility and results of actions are given as formulas.
▶ It is a logical consequence of the facts of a state and the general axioms that goals can be achieved by performing certain actions.
▶ The formal descriptions of states should correspond as closely as possible to what people may reasonably be presumed to know about them when deciding what to do.
Conjunctive Planning Problems

- **Initial state** $I : \{i_1, \ldots, i_m\}$ of ground fluents.
- **Goal state** $G : \{g_1, \ldots, g_n\}$ of ground fluents
- **Finite set** $\mathcal{A}$ of actions of the form
  \[ \{c_1, \ldots, c_l\} \Rightarrow \{e_1, \ldots, e_k\} , \]
  where $\{c_1, \ldots, c_l\}$ and $\{e_1, \ldots, e_k\}$ are multisets of fluents called conditions and effects respectively.
- **Assumption** each variable occurring in the effects of an action occurs also in its conditions.
- A conjunctive planning problem is the question of whether there exists a sequence of actions such that its execution transforms the initial state into the goal state.
Actions and Plans

- $C \Rightarrow E$ is applicable in $S$ iff there exists $\theta$ such that $C\theta \subseteq S$.
- The application of $C \Rightarrow E$ in $S$ leads to $S' = (S \setminus C\theta) \cup E\theta$.
  - If $S$ is ground then $S'$ is ground as well.

- A plan is a list of actions.
- A goal $G$ is satisfied iff there exists a plan $p$ which transforms $I$ into $S$ and $G \subseteq S$.
- Such a plan is called solution for the planning problem.
Blocks World

- The pickup action:

  \[
  \text{pickup}(V) : \{\text{clear}(V), \text{ontable}(V), \text{empty}\} \Rightarrow \{\text{holding}(V)\}
  \]

- The unstack action:

  \[
  \text{unstack}(V,W) : \{\text{clear}(V), \text{on}(V,W), \text{empty}\} \Rightarrow \{\text{holding}(V), \text{clear}(W)\}
  \]

- The putdown action:

  \[
  \text{putdown}(V) : \{\text{holding}(V)\} \Rightarrow \{\text{clear}(V), \text{ontable}(V), \text{empty}\}
  \]

- The stack action:

  \[
  \text{stack}(V,W) : \{\text{holding}(V), \text{clear}(W)\} \Rightarrow \{\text{on}(V,W), \text{clear}(V), \text{empty}\}
  \]
Sussman’s Anomaly

- $\mathcal{I} = \{ \text{ontable}(a), \text{ontable}(b), \text{on}(c, a), \text{clear}(b), \text{clear}(c), \text{empty} \}$
- $\mathcal{G} = \{ \text{ontable}(c), \text{on}(b, c), \text{on}(a, b), \text{clear}(a), \text{empty} \}$
- **Solution**
  - $p = [\text{unstack}(c, a), \text{putdown}(c), \text{pickup}(b), \text{stack}(b, c), \text{pickup}(a), \text{stack}(a, b)].$
A Fluent Calculus Implementation

- An action $C \Rightarrow E$ is represented by $\text{action}(C^{-I}, \text{name}, E^{-I})$:
  
  $\text{action}(\text{clear}(V) \circ \text{ontable}(V) \circ \text{empty}, \text{pickup}(V), \text{holding}(V))$
  $\text{action}(\text{clear}(V) \circ \text{on}(V, W) \circ \text{empty}, \text{unstack}(V, W), \text{holding}(V) \circ \text{clear}(W))$
  $\text{action}(\text{holding}(V), \text{putdown}(V), \text{clear}(V) \circ \text{ontable}(V) \circ \text{empty})$
  $\text{action}(\text{holding}(V) \circ \text{clear}(W), \text{stack}(V, W), \text{on}(V, W) \circ \text{clear}(V) \circ \text{empty})$

  Let $F_A$ be the set of these facts.

- Causality is expressed by $\text{causes}(s, p, s')$:
  
  $\text{causes}(X, [], Y) \quad \leftarrow \quad X \approx Y \circ Z$
  $\text{causes}(X, [V|W], Y) \quad \leftarrow \quad \text{action}(P, V, Q) \land P \circ Z \approx X$
  \hspace{1cm} $\land \text{causes}(Z \circ Q, W, Y)$

  $X \approx X$

  Let $F_C$ be the set of these clauses.

- The planning problem is the problem whether
  
  $F_A \cup F_C \cup E_{AC1} \models (\exists P) \text{causes}(I^{-I}, P, G^{-I})$ holds.
SLDE-Resolution

Let

- $\mathcal{F}$ be a set of definite clauses not containing $\approx / 2$ in their head plus $X \approx X$ and
- $\mathcal{E}$ be an equational system.
- Does $\mathcal{F} \cup \mathcal{E} \models G$ hold?.

Let $\text{up}_\mathcal{E}$ be an $\mathcal{E}$-unification procedure, $C$ a new variant $H \leftarrow A_1 \land \ldots \land A_m$ of a clause in $\mathcal{F}$ and $G$ be the goal clause $\leftarrow B_1 \land \ldots \land B_n$. If $H$ and $B_i$, $i \in [1, n]$, are $\mathcal{E}$-unifiable with $\theta \in \text{up}_\mathcal{E}(H, B_i)$, then

$$\leftarrow (B_1 \land \ldots \land B_{i-1} \land A_1 \land \ldots \land A_m \land B_{i+1} \land \ldots \land B_n)\theta$$

is called SLDE-resolvent of $C$ and $G$.

Theorem

- SLDE-resolution is sound if $\text{up}_\mathcal{E}$ is sound.
- SLDE-resolution is complete if $\text{up}_\mathcal{E}$ is complete.
- The selection of the literal $B_i$ is don’t care non–deterministic.
A Solution to Sussman’s Anomaly (1)

(1) \[ \leftarrow causes(ontable(a) \circ ontable(b) \circ on(c, a) \circ clear(b) \circ clear(c) \circ empty, \right. \]
\[ \left. W, \right. \]
\[ \left. ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \right. \]

(2) \[ \leftarrow action(P_1, V_1, Q_1) \land \]
\[ P_1 \circ Z_1 \approx ontable(a) \circ ontable(b) \circ on(c, a) \circ clear(b) \circ clear(c) \circ empty \land \]
\[ causes(Z_1 \circ Q_1, W_1, ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \]

(3) \[ \leftarrow clear(V_2) \circ on(V_2, W_2) \circ empty \circ Z_1 \approx \]
\[ ontable(a) \circ ontable(b) \circ on(c, a) \circ clear(b) \circ clear(c) \circ empty \land \]
\[ causes(Z_1 \circ holding(V_2) \circ clear(W_2), \right. \]
\[ \left. W_1, \right. \]
\[ ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \right. \]

(4) \[ \leftarrow causes(ontable(a) \circ ontable(b) \circ clear(b) \circ clear(a) \circ holding(c), \right. \]
\[ \left. W_1, \right. \]
\[ ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \right. \]
A Solution to Sussman’s Anomaly (2)

(7) \[ \leftarrow causes(ontable(a) \circ ontable(b) \circ clear(b) \circ clear(a) \circ clear(c) \circ ontable(c) \circ empty, \ W_4, \right. \]
\[ \left. ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \ \right. \]
\[ \vdots \]

(10) \[ \leftarrow causes(ontable(a) \circ clear(c) \circ ontable(c) \circ clear(a) \circ holding(b), \ W_7, \right. \]
\[ \left. ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \ \right. \]
\[ \vdots \]

(13) \[ \leftarrow causes(ontable(a) \circ ontable(c) \circ clear(a) \circ on(b, c) \circ clear(b) \circ empty, \ W_{10}, \right. \]
\[ \left. ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \ \right. \]
\[ \vdots \]

(16) \[ \leftarrow causes(ontable(c) \circ on(b, c) \circ clear(b) \circ holding(a), \ W_{13}, \right. \]
\[ \left. ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \ \right. \]
\[ \vdots \]

(19) \[ \leftarrow causes(ontable(c) \circ on(b, c) \circ clear(a) \circ on(a, b) \circ empty, \ W_{16}, \right. \]
\[ \left. ontable(c) \circ on(b, c) \circ on(a, b) \circ clear(a) \circ empty). \ \right. \]

(20) [ ]
Solving the Frame Problem

- In the fluent calculus the frame problem is mapped onto fluent matching and fluent unification problems.
- For example, let

\[
 s = \text{ontable}(a) \circ \text{ontable}(b) \circ \text{on}(c, a) \circ \text{clear}(b) \circ \text{clear}(c) \circ \text{empty}
\]

\[
 t = \text{clear}(c) \circ \text{on}(c, a) \circ \text{empty},
\]

then

\[
 \theta = \{ Z \mapsto \text{ontable}(a) \circ \text{ontable}(b) \circ \text{clear}(b) \}
\]

is a most general \(E\)-matcher for the \(E\)-matching problem

\[
 \mathcal{F}_{AC1} \models (\exists Z) s \approx t \circ Z.
\]

- Consequently, \(\text{unstack}(c, a)\) can be applied to \(s\) yielding

\[
 s' = \text{ontable}(a) \circ \text{ontable}(b) \circ \text{clear}(b) \circ \text{clear}(a) \circ \text{holding}(c).
\]
Why are Situations not Modelled by Sets?

Let $\mathcal{E}_{ACI_1} = \mathcal{E}_{AC_1} \cup \{(\forall X) X \circ X \approx X\}$.

In this case the $\mathcal{E}$–matching problem

$$\mathcal{E}_{ACI_1} \models (\exists Z) s \approx t \circ Z$$

has an additional solution, viz.

$$\eta = \{Z_1 \mapsto ontable(a) \circ ontable(b) \circ clear(b) \circ empty\}.$$

$\theta$ and $\eta$ are independent wrt $F_{ACI_1}$.

Computing the successor state in this case yields

$$s'' = ontable(a) \circ ontable(b) \circ clear(b) \circ clear(a) \circ holding(c) \circ empty.$$

which is not intended because the arm of the robot cannot be empty and holding an object at the same time.
Remarks

- Some people even believed that the frame problem cannot be solved within first order logic.
- Forward vs. backward planning.
- Incomplete specifications of initial situation, e.g.

\[(\exists X, P, Y)\]
\[\text{causes} (\text{ontable}(b) \circ Y, P, \text{ontable}(c) \circ \text{on}(b, c) \circ \text{on}(a, b) \circ \text{clear}(a) \circ \text{empty} \circ X).\]

- Indeterminate effects
- Consistency constraints
- etc; \(\rightsquigarrow\) Rational Agents
Sorts

- \( (\forall X, Y) \ (\text{number}(X) \land \text{number}(Y) \rightarrow \text{plus}(X, Y) \approx \text{plus}(Y, X)) \)
  - \( (\forall X, Y : \text{number}) \ \text{plus}(X, Y) \approx \text{plus}(Y, X). \)

- First order language with sorts:
  - first order language,
  - \( \text{sort} : \mathcal{V} \rightarrow 2^{\mathcal{R}_S}, \) where \( \mathcal{R}_S \subseteq \mathcal{R} \) is a finite set of unary predicate symbols called base sorts.

- Sort \( s \in 2^{\mathcal{R}_S}; \)
- \( \emptyset \in 2^{\mathcal{R}_S} \) is called top sort.
- We write \( X : s \) if \( \text{sort}(X) = s. \)
- We assume that for every sort \( s \) there are countably many variables \( X : s. \)
Sorts – Semantics

- Let $I$ be an interpretation with domain $D$,

$$I : \mathcal{S} = \{p_1, \ldots, p_n\} \mapsto s^I = D \cap p_1^I \cap \ldots \cap p_n^I.$$

- $\triangleright I : \emptyset \mapsto D$.

- A variable assignment $\mathcal{Z}$ is sorted iff for all $X : \mathcal{S}$ we find $X^\mathcal{Z} \in s^I$.

- We assume that all sorts are non-empty.

- $F^I, \mathcal{Z}$ is defined as usual except for

$$[(\exists X : \mathcal{S})\ F]^I, \mathcal{Z} = \top \text{ iff there exists } d \in s^I \text{ such that } F^I,\{X \mapsto d\}, \mathcal{Z} = \top.$$

$$[(\forall X : \mathcal{S})\ F]^I, \mathcal{Z} = \top \text{ iff for all } d \in s^I \text{ we find } F^I,\{X \mapsto d\}, \mathcal{Z} = \top.$$

Equational Logic (14th December 2007)
Relativization

Sorted formulas can be mapped onto unsorted ones by means of a relativization function $r$:

- $r(p(t_1, \ldots, t_n)) = p(t_1, \ldots, t_n)$
- $r(\neg F) = \neg r(F)$
- $r(F_1 \land F_2) = r(F_1) \land r(F_2)$
- $r(F_1 \lor F_2) = r(F_1) \lor r(F_2)$
- $r(F_1 \rightarrow F_2) = r(F_1) \rightarrow r(F_2)$
- $r(F_1 \leftrightarrow F_2) = r(F_1) \leftrightarrow r(F_2)$
- $r((\forall X : s) F) = (\forall Y) \ (p_1(Y) \land \ldots \land p_n(Y) \rightarrow r(F\{X/Y\}))$
  if $\text{sort}(X) = s = \{p_1, \ldots, p_n\}$ and $Y$ is a new variable
- $r((\exists X : s) F) = (\exists Y) \ (p_1(Y) \land \ldots \land p_n(Y) \land r(F\{X/Y\}))$
  if $\text{sort}(X) = s = \{p_1, \ldots, p_n\}$ and $Y$ is a new variable
Sorting Function and Relation Symbols

- Each atom of the form $p(t_1, \ldots, t_n)$ can be equivalently replaced by
  \[(\forall X_1 \ldots X_n) (p(X_1, \ldots, X_n) \leftarrow X_1 \approx t_1 \land \ldots \land X_n \approx t_n).\]

- Each atom $A$ with $A[\pi] = f(t_1, \ldots, t_n)$ can be equivalently replaced by
  \[(\forall X_1 \ldots X_n) A[\pi \mapsto f(X_1, \ldots, X_n)] \leftarrow X_1 \approx t_1 \land \ldots \land X_n \approx t_n.\]

- Each formula $F$ can be transformed into an equivalent formula $F'$, in which
  - all arguments of function and relation symbols different from $\approx / 2$
    are variables and
  - all equations are of the form $d_1 \approx d_2$ or $f(X_1, \ldots, X_n) \approx d$, where
    $X_1, \ldots, X_n$ are variables and $d$, $d_1$ and $d_2$ are variables or constants.

- Sorting the variables occurring in $F'$ effectively sorts the function and relation
  symbols.
Sort Declaration

- $F'$ is usually quite lengthy and cumbersome to read.
- If $\text{sort}(X) = s$ then the sort declaration for the variable $X$ is $X : s$.

Let $s_i, 1 \leq i \leq n$, and $s$ be sorts, $f/n$ a function and $p/n$ a relation symbol. Then

$$f : s_1 \times \ldots \times s_n \rightarrow s$$

and

$$p : s_1 \times \ldots \times s_n$$

are sort declarations for $f/n$ and $p/n$ respectively.