Part 3: First-Order Logic with Equality

Equality is the most important relation in mathematics and functional programming.

In principle, problems in first-order logic with equality can be handled by, e.g., resolution theorem provers.

Equality is theoretically difficult:
First-order functional programming is Turing-complete.

But: resolution theorem provers cannot even solve problems that are intuitively easy.

Consequence: to handle equality efficiently, knowledge must be integrated into the theorem prover.

3.1 Handling Equality Naively

Proposition 3.1:

Let F be a closed first-order formula with equality. Let $\sim \notin \Pi$ be a new predicate symbol. The set $Eq(\Sigma)$ contains the formulas

$$\forall x\,(x \sim x)$$
$$\forall x, y\,(x \sim y \rightarrow y \sim x)$$
$$\forall x, y, z\,(x \sim y \wedge y \sim z \rightarrow x \sim z)$$
$$\forall \vec{x}, \vec{y}\,(x_1 \sim y_1 \wedge \cdots \wedge x_n \sim y_n \rightarrow f(x_1, \dots, x_n) \sim f(y_1, \dots, y_n))$$
$$\forall \vec{x}, \vec{y}\,(x_1 \sim y_1 \wedge \cdots \wedge x_n \sim y_n \wedge p(x_1, \dots, x_n) \rightarrow p(y_1, \dots, y_n))$$

for every $f/n \in \Omega$ and $p/n \in \Pi$. Let $\tilde{F}$ be the formula that one obtains from F if every occurrence of $\approx$ is replaced by $\sim$. Then F is satisfiable if and only if $Eq(\Sigma) \cup \{\tilde{F}\}$ is satisfiable.

Handling Equality Naively

By giving the equality axioms explicitly, first-order problems with equality can in principle be solved by a standard resolution or tableaux prover.

But this is unfortunately not efficient
(mainly due to the transitivity and congruence axioms).

Roadmap

How to proceed:

- Arbitrary binary relations.
- Equations (unit clauses with equality):

  Term rewrite systems.

  Expressing semantic consequence syntactically.

  Entailment for equations.

- Equational clauses:

  Entailment for clauses with equality.

3.2 Abstract Reduction Systems

Abstract reduction system: $(A, \rightarrow)$, where

A is a set,

$\rightarrow \subseteq A \times A$ is a binary relation on A.

Abstract Reduction Systems

$\rightarrow^0$	$= \{\, (x, x) \mid x \in A \,\}$	identity
$\rightarrow^{i+1}$	$= \rightarrow^i \circ \rightarrow$	$i + 1$-fold composition
$\rightarrow^+$	$= \bigcup_{i>0} \rightarrow^i$	transitive closure
$\rightarrow^*$	$= \bigcup_{i\geq 0} \rightarrow^i \;=\; \rightarrow^+ \cup \rightarrow^0$	reflexive transitive closure
$\rightarrow^=$	$= \rightarrow \cup \rightarrow^0$	reflexive closure
$\rightarrow^{-1}$	$= \leftarrow \;=\; \{\, (x, y) \mid y \rightarrow x \,\}$	inverse
$\leftrightarrow$	$= \rightarrow \cup \leftarrow$	symmetric closure
$\leftrightarrow^+$	$= (\leftrightarrow)^+$	transitive symmetric closure
$\leftrightarrow^*$	$= (\leftrightarrow)^*$	refl. trans. symmetric closure

Abstract Reduction Systems

$x \in A$ is reducible, if there is a y such that $x \to y$.

x is in normal form (irreducible), if it is not reducible.

y is a normal form of x, if $x \to^* y$ and y is in normal form.
Notation: $y = x{\downarrow}$ (if the normal form of x is unique).

x and y are joinable, if there is a z such that $x \to^* z \leftarrow^* y$.
Notation: $x \downarrow y$.

Abstract Reduction Systems

A relation $\rightarrow$ is called

Church-Rosser, if $x \leftrightarrow^* y$ implies $x \downarrow y$.

confluent, if $x \leftarrow^* z \rightarrow^* y$ implies $x \downarrow y$.

locally confluent, if $x \leftarrow z \rightarrow y$ implies $x \downarrow y$.

terminating, if there is no infinite decreasing chain $x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \ldots$.

normalizing, if every $x \in A$ has a normal form.

convergent, if it is confluent and terminating.

Abstract Reduction Systems

Lemma 3.2:

If $\rightarrow$ is terminating, then it is normalizing.

Note: The reverse implication does not hold.

Abstract Reduction Systems

Theorem 3.3:

The following properties are equivalent:

(i) $\rightarrow$ has the Church-Rosser property.

(ii) $\rightarrow$ is confluent.

Proof:

(i)$\Rightarrow$(ii): trivial.

(ii)$\Rightarrow$(i): by induction on the number of peaks in the derivation $x \leftrightarrow^* y$.

Abstract Reduction Systems

Lemma 3.4:
If $\rightarrow$ is confluent, then every element has at most one normal form.

Corollary 3.5:
If $\rightarrow$ is normalizing and confluent, then every element x has a unique normal form.

Proposition 3.6:
If $\rightarrow$ is normalizing and confluent, then $x \leftrightarrow^* y$ if and only if $x\!\downarrow = y\!\downarrow$.

Well-Founded Orderings

Lemma 3.7:
If $\rightarrow$ is a terminating binary relation over A,
then $\rightarrow^+$ is a well-founded partial ordering.

Lemma 3.8:
If $>$ is a well-founded partial ordering and $\rightarrow \subseteq >$,
then $\rightarrow$ is terminating.

Proving Confluence

Theorem 3.9 ("Newman's Lemma"):

If a terminating relation $\longrightarrow$ is locally confluent, then it is confluent.

Proof:

Let $\longrightarrow$ be a terminating and locally confluent relation.

Then $\longrightarrow^+$ is a well-founded ordering.

Define $P(z) \;\Leftrightarrow\; \big(\forall x, y : x \longleftarrow^* z \longrightarrow^* y \Rightarrow x \downarrow y\big)$.

Prove $P(z)$ for all $x \in A$ by well-founded induction over $\longrightarrow^+$:

Case 1: $x \longleftarrow^0 z \longrightarrow^* y$: trivial.

Case 2: $x \longleftarrow^* z \longrightarrow^0 y$: trivial.

Case 3: $x \longleftarrow^* x' \longleftarrow z \longrightarrow y' \longrightarrow^* y$: use local confluence, then use the induction hypothesis.

Proving Termination: Monotone Mappings

Let $(A, >_A)$ and $(B, >_B)$ be partial orderings.
A mapping $\varphi : A \to B$ is called monotone,
if $x >_A y$ implies $\varphi(x) >_B \varphi(y)$ for all $x, y \in A$.

Lemma 3.10:
If $\varphi : A \to B$ is a monotone mapping from $(A, >_A)$ to $(B, >_B)$
and $(B, >_B)$ is well-founded, then $(A, >_A)$ is well-founded.

3.3 Rewrite Systems

Some notation:

Positions of a term s:

$$\text{pos}(x) = \{\varepsilon\},$$
$$\text{pos}(f(s_1, \dots, s_n)) = \{\varepsilon\} \cup \bigcup_{i=1}^{n} \{\, i\,p \mid p \in \text{pos}(s_i)\,\}.$$

Size of a term s:

$|s| =$ cardinality of $\text{pos}(s)$.

Prefix order for $p, q \in \text{pos}(s)$:

p above q: $p \le q$ if $p\,p' = q$ for some p',
p strictly above q: $p < q$ if $p \le q$ and not $q \le p$,
p and q parallel: $p \parallel q$ if neither $p \le q$ nor $q \le p$.

Rewrite Systems

Some notation:

Subterm of s at a position $p \in \mathrm{pos}(s)$:

$$s/\varepsilon = s,$$
$$f(s_1, \ldots, s_n)/i\,p = s_i/p.$$

Replacement of the subterm at position $p \in \mathrm{pos}(s)$ by t:

$$s[t]_\varepsilon = t,$$
$$f(s_1, \ldots, s_n)[t]_{ip} = f(s_1, \ldots, s_i[t]_p, \ldots, s_n).$$

Rewrite Relations

Let E be a set of equations.

The rewrite relation $\rightarrow_E \subseteq \mathsf{T}_\Sigma(X) \times \mathsf{T}_\Sigma(X)$ is defined by

$$
\begin{aligned}
s \rightarrow_E t \quad \text{iff} \quad & \text{there exist } (l \approx r) \in E,\ p \in \mathrm{pos}(s), \\
& \text{and } \sigma : X \rightarrow \mathsf{T}_\Sigma(X), \\
& \text{such that } s/p = l\sigma \text{ and } t = s[r\sigma]_p.
\end{aligned}
$$

An instance of the lhs (left-hand side) of an equation is called a redex (reducible expression).
Contracting a redex means replacing it with the corresponding instance of the rhs (right-hand side) of the rule.

Rewrite Relations

An equation $l \approx r$ is also called a rewrite rule, if l is not a variable and $\text{var}(l) \supseteq \text{var}(r)$.

Notation: $l \longrightarrow r$.

A set of rewrite rules is called a term rewrite system (TRS).

Rewrite Relations

We say that a set of equations E or a TRS R is terminating, if the rewrite relation $\longrightarrow_E$ or $\longrightarrow_R$ has this property.

(Analogously for other properties of abstract reduction systems).

Note: If E is terminating, then it is a TRS.

E-Algebras

Let E be a set of closed equations. A Σ-algebra $\mathcal{A}$ is called an E-algebra, if $\mathcal{A} \models \forall \vec{x}(s \approx t)$ for all $\forall \vec{x}(s \approx t) \in E$.

If $E \models \forall \vec{x}(s \approx t)$ (i.e., $\forall \vec{x}(s \approx t)$ is valid in all E-algebras), we write this also as $s \approx_E t$.

Goal:
Use the rewrite relation $\rightarrow_E$ to express the semantic consequence relation syntactically:

$s \approx_E t$ if and only if $s \leftrightarrow_E^* t$.

E-Algebras

Let E be a set of equations over $\mathrm{T}_\Sigma(X)$. The following inference system allows to derive consequences of E:

E-Algebras

$$E \vdash t \approx t \qquad \text{(Reflexivity)}$$

$$\frac{E \vdash t \approx t'}{E \vdash t' \approx t} \qquad \text{(Symmetry)}$$

$$\frac{E \vdash t \approx t' \qquad E \vdash t' \approx t''}{E \vdash t \approx t''} \qquad \text{(Transitivity)}$$

$$\frac{E \vdash t_1 \approx t'_1 \quad \dots \quad E \vdash t_n \approx t'_n}{E \vdash f(t_1, \dots, t_n) \approx f(t'_1, \dots, t'_n)} \qquad \text{(Congruence)}$$

$$E \vdash t\sigma \approx t'\sigma \qquad \text{(Instance)}$$

if $(t \approx t') \in E$ and $\sigma : X \to \mathrm{T}_\Sigma(X)$

E-Algebras

Lemma 3.11:

The following properties are equivalent:

(i) $s \leftrightarrow_E^* t$

(ii) $E \vdash s \approx t$ is derivable.

Proof:

(i)$\Rightarrow$(ii): $s \leftrightarrow_E t$ implies $E \vdash s \approx t$ by induction on the depth of the position where the rewrite rule is applied;
then $s \leftrightarrow_E^* t$ implies $E \vdash s \approx t$ by induction on the number of rewrite steps in $s \leftrightarrow_E^* t$.

(ii)$\Rightarrow$(i): By induction on the size of the derivation for $E \vdash s \approx t$.

E-Algebras

Constructing a quotient algebra:

Let X be a set of variables.

For $t \in \mathrm{T}_\Sigma(X)$ let $[t] = \{\, t' \in \mathrm{T}_\Sigma(X) \mid E \vdash t \approx t' \,\}$ be the congruence class of t.

Define a Σ-algebra $\mathrm{T}_\Sigma(X)/E$ (abbreviated by $\mathcal{T}$) as follows:

$U_\mathcal{T} = \{\, [t] \mid t \in \mathrm{T}_\Sigma(X) \,\}$.

$f_\mathcal{T}([t_1], \ldots, [t_n]) = [f(t_1, \ldots, t_n)]$ for $f/n \in \Omega$.

E-Algebras

Lemma 3.12:

$f_{\mathcal{T}}$ is well-defined:

If $[t_i] = [t_i']$, then $[f(t_1, \ldots, t_n)] = [f(t_1', \ldots, t_n')]$.

Proof:

Follows directly from the *Congruence* rule for $\vdash$.

E-Algebras

Lemma 3.13:
$\mathcal{T} = \mathrm{T}_\Sigma(X)/E$ is an E-algebra.

Proof:
Let $\forall x_1 \dots x_n (s \approx t)$ be an equation in E; let β be an arbitrary assignment.

We have to show that $\mathcal{T}(\beta)(\forall \vec{x}(s \approx t)) = 1$, or equivalently, that $\mathcal{T}(\gamma)(s) = \mathcal{T}(\gamma)(t)$ for all $\gamma = \beta[x_i \mapsto [t_i] \mid 1 \leq i \leq n]$ with $[t_i] \in U_{\mathcal{T}}$.

Let $\sigma = [t_1/x_1, \dots, t_n/x_n]$, then $s\sigma \in \mathcal{T}(\gamma)(s)$ and $t\sigma \in \mathcal{T}(\gamma)(t)$.

By the *Instance* rule, $E \vdash s\sigma \approx t\sigma$ is derivable,
hence $\mathcal{T}(\gamma)(s) = [s\sigma] = [t\sigma] = \mathcal{T}(\gamma)(t)$.

E-Algebras

Lemma 3.14:

Let X be a countably infinite set of variables; let $s, t \in \mathrm{T}_\Sigma(X)$. If $\mathrm{T}_\Sigma(X)/E \models \forall\vec{x}(s \approx t)$, then $E \vdash s \approx t$ is derivable.

Proof:

Assume that $\mathcal{T} \models \forall\vec{x}(s \approx t)$, i.e., $\mathcal{T}(\beta)(\forall\vec{x}(s \approx t)) = 1$. Consequently, $\mathcal{T}(\gamma)(s) = \mathcal{T}(\gamma)(t)$ for all $\gamma = \beta[\,x_i \mapsto [t_i] \mid i \in I\,]$ with $[t_i] \in U_{\mathcal{T}}$.

Choose $t_i = x_i$, then $[s] = \mathcal{T}(\gamma)(s) = \mathcal{T}(\gamma)(t) = [t]$, so $E \vdash s \approx t$ is derivable by definition of $\mathcal{T}$.

E-Algebras

Theorem 3.15 ("Birkhoff's Theorem"):
Let X be a countably infinite set of variables, let E be a set of (universally quantified) equations. Then the following properties are equivalent for all $s, t \in \mathrm{T}_\Sigma(X)$:

(i) $s \leftrightarrow_E^* t$.

(ii) $E \vdash s \approx t$ is derivable.

(iii) $s \approx_E t$, i.e., $E \models \forall \vec{x}(s \approx t)$.

(iv) $\mathrm{T}_\Sigma(X)/E \models \forall \vec{x}(s \approx t)$.

E-Algebras

Proof:

(i)$\Leftrightarrow$(ii): See above (slide 23).

(ii)$\Rightarrow$(iii): By induction on the size of the derivation for $E \vdash s \approx t$.

(iii)$\Rightarrow$(iv): Obvious, since $\mathcal{T} = \mathcal{T}_E(X)$ is an E-algebra.

(iv)$\Rightarrow$(ii): See above (slide 27).

Universal Algebra

$\mathsf{T}_\Sigma(X)/E = \mathsf{T}_\Sigma(X)/{\approx_E} = \mathsf{T}_\Sigma(X)/{\leftrightarrow_E^*}$ is called the free E-algebra with generating set $X/{\approx_E} = \{\, [x] \mid x \in X \,\}$:

Every mapping $\varphi : X/{\approx_E} \to \mathcal{B}$ for some E-algebra $\mathcal{B}$ can be extended to a homomorphism $\hat{\varphi} : \mathsf{T}_\Sigma(X)/E \to \mathcal{B}$.

$\mathsf{T}_\Sigma(\emptyset)/E = \mathsf{T}_\Sigma(\emptyset)/{\approx_E} = \mathsf{T}_\Sigma(\emptyset)/{\leftrightarrow_E^*}$ is called the initial E-algebra.

Universal Algebra

$\approx_E = \{ (s, t) \mid E \models s \approx t \}$
is called the equational theory of E.

$\approx_E^I = \{ (s, t) \mid \mathrm{T}_\Sigma(\emptyset)/E \models s \approx t \}$
is called the inductive theory of E.

Example:

Let $E = \{\forall x(x + 0 \approx x),\ \forall x \forall y(x + s(y) \approx s(x + y))\}$.
Then $x + y \approx_E^I y + x$, but $x + y \not\approx_E y + x$.

Rewrite Relations

Corollary 3.16:
If E is convergent (i.e., terminating and confluent),
then $s \approx_E t$ if and only if $s \leftrightarrow^*_E t$ if and only if $s{\downarrow_E} = t{\downarrow_E}$.

Corollary 3.17:
If E is finite and convergent, then $\approx_E$ is decidable.

Reminder:
If E is terminating, then it is confluent if and only if
it is locally confluent.

Rewrite Relations

Problems:

Show local confluence of E.

Show termination of E.

Transform E into an equivalent set of equations that is locally confluent and terminating.