Programming Languages

First-order logic

Introduction

Syntax of first-order logic

Natural deduction for first-order logic

Semantics of first-order logic

Unification of terms

Introduction

Propositional logic

Allows reasoning about propositions.
Example: $\textbf{ItRains} \vee \neg \textbf{ItRains}$

First-order logic

Allows reasoning about elements about which one predicates.
Example:

$$\forall X. (\textbf{IsEven}(X) \Rightarrow \neg \textbf{IsEven}(\texttt{succ}(X)))$$

Extends propositional logic with terms and quantifiers.

What's all this logic for? I signed up for computer science...

Close connection between first-order logic and computing.

Historically

- Hilbert's decision problem.

Nowadays

- Computability and descriptive complexity.
- Knowledge representation, multi-agent systems.
- Artificial intelligence, automated reasoning.
- Formal methods, automated verification.
- Relational databases, query languages.
- Hardware verification.
- ...
- **Foundation of logic programming**.

Logic programming

Ideal of declarative programming

Programs should resemble specifications.

In particular: logic programming

- The user writes a formula:

$$\exists X.\, \mathbf{P}(X)$$

- The system tries to satisfy or refute the formula.
- If it succeeds in satisfying it, the system produces an output that verifies the desired property P.

First-order languages

Definition

A **first-order language** $\mathcal{L}$ is given by:

1. A set of **function symbols** $\mathcal{F} = \{\mathtt{f}, \mathtt{g}, \mathtt{h}, \ldots\}$.
   Each function symbol has an associated arity (≥ 0).
2. A set of **predicate symbols** $\mathcal{P} = \{\mathbf{P}, \mathbf{Q}, \mathbf{R}, \ldots\}$.
   Each predicate symbol has an associated arity (≥ 0).

First-order terms

Assume a first-order language $\mathcal{L}$ is fixed
and a countably infinite set of **variables** $\mathcal{X} = \{X, Y, Z, \ldots\}$.

Definition

The set $\mathcal{T}$ of **terms** is defined by the following grammar:

$$t \quad ::= \quad X \quad | \quad \mathtt{f}(t_1, \ldots, t_n)$$

where:

- X denotes a variable
- $\mathtt{f}$ denotes a function symbol of arity n

First-order terms

Example — the language $\mathcal{L}_{\text{arithmetic}}$

$$\underbrace{0^0 \quad \texttt{succ}^1 \quad +^2 \quad *^2}_{\text{function symbols}} \qquad \underbrace{=^2 \quad <^2}_{\text{predicate symbols}}$$

Example — terms over the language $\mathcal{L}_{\text{arithmetic}}$

$$+(0, \texttt{succ}(X)) \qquad *(+(X, Y), Z)$$

Function symbols of arity 0 are called constants.

Note. We use infix notation as a convenience.

$$0 + \texttt{succ}(X) \qquad (X + Y) * Z$$

First-order formulas

Recall the grammar of formulas in propositional logic and let's extend it to first-order logic.

$$
\begin{array}{rcll}
\sigma & ::= & \mathbf{P}(t_1, \ldots, t_n) & \textbf{atomic formula} \\
 & | & \bot & \text{contradiction} \\
 & | & \sigma \Rightarrow \sigma & \text{implication} \\
 & | & \sigma \wedge \sigma & \text{conjunction} \\
 & | & \sigma \vee \sigma & \text{disjunction} \\
 & | & \neg\sigma & \text{negation} \\
 & | & \forall X.\, \sigma & \textbf{universal quantification} \\
 & | & \exists X.\, \sigma & \textbf{existential quantification}
\end{array}
$$

P denotes a predicate symbol of arity n.

Quantifiers bind a variable X.

First-order formulas

Recall — the language $\mathcal{L}_{\text{arithmetic}}$

$$\mathtt{0}^0 \quad \mathtt{succ}^1 \quad +^2 \quad *^2 \qquad\qquad =^2 \quad <^2$$

Example — formulas over $\mathcal{L}_{\text{arithmetic}}$

$$\forall X.\, \exists Y.\, = (+(X, Y), \mathtt{0})$$

$$\forall X.\, \forall Y.\, (\mathtt{succ}(X) = \mathtt{succ}(Y) \Rightarrow X = Y)$$

$$\forall X.\, (X < \mathtt{0} \ \vee \ X = \mathtt{0} \ \vee \ \mathtt{0} < X)$$

First-order formulas

An occurrence of a variable X in a formula is:

bound	if it is within the scope of a quantifier $\forall X / \exists X$,
free	otherwise.

Two formulas that differ only in the names of bound variables are considered equal.

Example

$$\forall X.\, \exists Y.\, \mathbf{P}(X, Y) \;\equiv\; \forall Y.\, \exists X.\, \mathbf{P}(Y, X) \;\equiv\; \forall A.\, \exists B.\, \mathbf{P}(A, B)$$

First-order formulas

We denote $\sigma\{X := t\}$ as the substitution of the free occurrences of X in the formula σ by the term t, avoiding variable capture.

Example

Let:

$$\sigma \quad :\equiv \quad \mathtt{succ}(X) = Y \implies \exists Z.\, X + Z = Y$$

then:

$$\sigma\{X := Z * Z\} \;\equiv\; \mathtt{succ}(Z * Z) = Y \implies \exists Z'.\, (Z * Z) + Z' = Y$$

Natural deduction

Propositional natural deduction is extended to first order.

Same as before:

1. A **context** Γ is a finite set of formulas.
2. A **sequent** is of the form $\Gamma \vdash \sigma$.

All propositional natural deduction rules remain valid.
Introduction and elimination rules for $\forall$ and $\exists$ are added.

Axiom	AX		
Conjunction	$\wedge$I	$\wedge$E$_1$	$\wedge$E$_2$
Disjunction	$\vee$I$_1$	$\vee$I$_2$	$\vee$E
Implication	$\Rightarrow$I	$\Rightarrow$E	
Negation	$\neg$I	$\neg$E	
Contradiction	$\bot$E		
Classical logic	$\neg\neg$E		
Universal quantification	$\forall$I	$\forall$E	
Existential quantification	$\exists$I	$\exists$E	

Universal quantification

Elimination rule

$$\frac{\Gamma \vdash \forall X.\, \sigma}{\Gamma \vdash \sigma\{X := t\}}\ \forall\mathrm{E}$$

Introduction rule

$$\frac{\Gamma \vdash \sigma \qquad X \notin \mathsf{fv}(\Gamma)}{\Gamma \vdash \forall X.\, \sigma}\ \forall\mathrm{I}$$

Universal quantification

Example

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\forall X.\,(\mathbf{P}(X) \wedge \mathbf{Q}(X)) \vdash \forall X.\,(\mathbf{P}(X) \wedge \mathbf{Q}(X))}\,\text{AX}}{\forall X.\,(\mathbf{P}(X) \wedge \mathbf{Q}(X)) \vdash \mathbf{P}(\mathsf{cos}(X)) \wedge \mathbf{Q}(\mathsf{cos}(X))}\,\forall\text{E}}{\forall X.\,(\mathbf{P}(X) \wedge \mathbf{Q}(X)) \vdash \mathbf{P}(\mathsf{cos}(X))}\,\wedge\text{E}_1}{\forall X.\,(\mathbf{P}(X) \wedge \mathbf{Q}(X)) \vdash \forall X.\,\mathbf{P}(\mathsf{cos}(X))}\,\forall\text{I}}{\vdash \forall X.\,(\mathbf{P}(X) \wedge \mathbf{Q}(X)) \Rightarrow \forall X.\,\mathbf{P}(\mathsf{cos}(X))}\,\Rightarrow\text{I}
$$

Universal quantification

Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbf{P}(X), \forall X.\, \forall Y.\, \mathbf{Q}(X,Y) \vdash \forall Z.\, \forall Y.\, \mathbf{Q}(Z,Y)}\ \text{AX}}{\mathbf{P}(X), \forall X.\, \forall Y.\, \mathbf{Q}(X,Y) \vdash \forall Y.\, \mathbf{Q}(Z,Y)}\ \forall\text{E}}{\mathbf{P}(X), \forall X.\, \forall Y.\, \mathbf{Q}(X,Y) \vdash \mathbf{Q}(Z,Y)}\ \forall\text{E}}{\mathbf{P}(X), \forall X.\, \forall Y.\, \mathbf{Q}(X,Y) \vdash \forall \quad Z.\, \mathbf{Q}(\quad Z, Y)}\ \forall\text{I}}{\mathbf{P}(X), \forall X.\, \forall Y.\, \mathbf{Q}(X,Y) \vdash \forall Y.\, \forall X.\, \mathbf{Q}(X,Y)}\ \forall\text{I}$$

Universal quantification

Why is it required that $X \notin \text{fv}(\Gamma)$ in the rule $\forall\textsc{i}$?

Example — incorrect application of the rule $\forall\textsc{i}$

$$\frac{\textbf{IsEven}(N) \vdash \textbf{IsEven}(N)}{\textbf{IsEven}(N) \vdash \forall N.\, \textbf{IsEven}(N)} \Leftarrow \text{Invalid reasoning step}$$

Existential quantification

Introduction rule

$$\frac{\Gamma \vdash \sigma\{X := t\}}{\Gamma \vdash \exists X.\, \sigma}\,\exists\mathrm{I}$$

Elimination rule

$$\frac{\Gamma \vdash \exists X.\, \sigma \qquad \Gamma, \sigma \vdash \tau \qquad X \notin \mathsf{fv}(\Gamma, \tau)}{\Gamma \vdash \tau}\,\exists\mathrm{E}$$

Existential quantification

Example

$$\dfrac{\dfrac{}{\sigma \vdash \sigma}\,\text{AX} \qquad \dfrac{\dfrac{\dfrac{}{\sigma, \mathbf{P}(\cos(X)) \vdash \mathbf{P}(\cos(X))}\,\text{AX}}{\sigma, \mathbf{P}(\cos(X)) \vdash \mathbf{P}(\cos(X)) \vee \mathbf{Q}(\cos(X))}\,\vee\text{I}_1}{\sigma, \mathbf{P}(\cos(X)) \vdash \exists X.\,(\mathbf{P}(X) \vee \mathbf{Q}(X))}\,\exists\text{I}}{\dfrac{\sigma \vdash \exists X.\,(\mathbf{P}(X) \vee \mathbf{Q}(X))}{\vdash \exists X.\,\mathbf{P}(\cos(X)) \Rightarrow \exists X.\,(\mathbf{P}(X) \vee \mathbf{Q}(X))}\,\Rightarrow\text{I}}\,\exists\text{E}$$

$\sigma :\equiv \exists X.\,\mathbf{P}(\cos(X))$

Existential quantification

Example

$$
\dfrac{
  \dfrac{}{\sigma \vdash \sigma}\,\textrm{AX}
  \qquad
  \dfrac{
    \dfrac{
      \dfrac{
        \dfrac{}{\sigma, \mathbf{P}(W, W), \mathbf{Q}(X) \vdash \mathbf{P}(W, W)}\,\textrm{AX}
      }{\sigma, \mathbf{P}(W, W) \vdash \mathbf{Q}(X) \Rightarrow \mathbf{P}(W, W)}\,{\Rightarrow}\textrm{I}
    }{\sigma, \mathbf{P}(W, W) \vdash \exists Z.\,(\mathbf{Q}(X) \Rightarrow \mathbf{P}(W, Z))}\,\exists\textrm{I}
  }{\sigma, \mathbf{P}(W, W) \vdash \exists Y.\,\exists Z.\,(\mathbf{Q}(X) \Rightarrow \mathbf{P}(Y, Z))}\,\exists\textrm{I}
}{\exists\ W.\,\mathbf{P}(\ W,\ W) \vdash \exists Y.\,\exists Z.\,(\mathbf{Q}(X) \Rightarrow \mathbf{P}(Y, Z))}\,\exists\textrm{E}
$$

$\sigma :\equiv \exists W.\,\mathbf{P}(W, W)$

Existential quantification

To think about

Why is it required that $X \notin \mathsf{fv}(\Gamma, \tau)$ in the rule $\exists\mathrm{E}$?

First-order structures

Assume a first-order language $\mathcal{L}$ is fixed.

Definition

A **first-order structure** is a pair $\mathcal{M} = (M, I)$ where:

- M is a **non-empty** set, called the *universe*.
- I is a function that gives an interpretation to each symbol.
- For each function symbol $\texttt{f}$ of arity n:

$$I(\texttt{f}) : M^n \to M$$

- For each predicate symbol **P** of arity n:

$$I(\mathbf{P}) \subseteq M^n$$

First-order structures

Recall — the language $\mathcal{L}_{\text{arithmetic}}$

$$\mathtt{0}^0 \quad \mathtt{succ}^1 \quad +^2 \quad *^2 \qquad\qquad =^2 \quad <^2$$

Example — a structure over $\mathcal{L}_{\text{arithmetic}}$

$M := \mathbb{N}$ (elements are natural numbers)

$$\begin{aligned} I(\mathtt{0}) &= 0 \\ I(\mathtt{succ})(n) &= n+1 \\ I(+)(n,m) &= n+m \\ I(*)(n,m) &= n \cdot m \end{aligned} \qquad \begin{aligned} (n,m) \in I(=) &\iff n = m \\ (n,m) \in I(<) &\iff n < m \end{aligned}$$

Under this structure, the formula $\forall X.\, X = X + X$ is false.

First-order structures

Recall — the language $\mathcal{L}_{\text{arithmetic}}$

$$\texttt{0}^0 \quad \texttt{succ}^1 \quad +^2 \quad *^2 \qquad\qquad =^2 \quad <^2$$

Example — another structure over $\mathcal{L}_{\text{arithmetic}}$

$M := \mathcal{P}(\mathbb{R})$ (elements are sets of real numbers)

$$\begin{aligned} I(\texttt{0}) &= \varnothing \\ I(\texttt{succ})(A) &= \{1 + x \mid x \in A\} \\ I(+)(A, B) &= A \cup B \\ I(*)(A, B) &= A \cap B \end{aligned}$$

$$(A, B) \in I(=) \iff A = B$$

$$(A, B) \in I(<) \iff A \subseteq B$$

Under this structure, the formula $\forall X.\, X = X + X$ is true.

Interpretation of terms

Assume a first-order structure $\mathcal{M} = (M, I)$ is fixed.

Definition

An **assignment** is a function that assigns an element of the universe to each variable:

$$\mathbf{a} : \mathcal{X} \to M$$

Definition – interpretation of terms

Each term $t \in \mathcal{T}$ is interpreted as an element $\mathbf{a}(t) \in M$, extending the definition of $\mathbf{a}$ to terms:

$$\mathbf{a}(\mathtt{f}(t_1, \ldots, t_n)) \quad = \quad I(\mathtt{f})(\mathbf{a}(t_1), \ldots, \mathbf{a}(t_n))$$

Interpretation of formulas

Assume a first-order structure $\mathcal{M} = (M, I)$ is fixed.

We define a **satisfaction** relation $\mathbf{a} \models_{\mathcal{M}} \sigma$.
"The assignment **a** (under the structure $\mathcal{M}$) satisfies the formula σ".

$$
\begin{array}{lll}
\mathbf{a} \models_{\mathcal{M}} \mathbf{P}(t_1, \ldots, t_n) & \text{iff} & (\mathbf{a}(t_1), \ldots, \mathbf{a}(t_n)) \in I(\mathbf{P}) \\
\mathbf{a} \models_{\mathcal{M}} \sigma \wedge \tau & \text{iff} & \mathbf{a} \models_{\mathcal{M}} \sigma \ \text{ and } \ \mathbf{a} \models_{\mathcal{M}} \tau \\
\mathbf{a} \models_{\mathcal{M}} \sigma \vee \tau & \text{iff} & \mathbf{a} \models_{\mathcal{M}} \sigma \ \text{ or } \ \mathbf{a} \models_{\mathcal{M}} \tau \\
\mathbf{a} \models_{\mathcal{M}} \sigma \Rightarrow \tau & \text{iff} & \mathbf{a} \not\models_{\mathcal{M}} \sigma \ \text{ or } \ \mathbf{a} \models_{\mathcal{M}} \tau \\
\mathbf{a} \models_{\mathcal{M}} \neg\sigma & \text{iff} & \mathbf{a} \not\models_{\mathcal{M}} \sigma \\
\mathbf{a} \not\models_{\mathcal{M}} \bot & & \\
\mathbf{a} \models_{\mathcal{M}} \forall X.\, \sigma & \text{iff} & \mathbf{a}[X \mapsto m] \models_{\mathcal{M}} \sigma \text{ for all } m \in M \\
\mathbf{a} \models_{\mathcal{M}} \exists X.\, \sigma & \text{iff} & \mathbf{a}[X \mapsto m] \models_{\mathcal{M}} \sigma \text{ for some } m \in M \\
 & & \\
\mathbf{a} \models_{\mathcal{M}} \sigma \clubsuit \tau & \text{iff} & \mathbf{a} \models_{\mathcal{M}} \sigma \text{ broccoli } \mathbf{a} \models_{\mathcal{M}} \tau
\end{array}
$$

(A "joke" by J.-Y. Girard)

Validity and satisfiability

We say that a formula σ is:

VALID if $\mathbf{a} \models_{\mathcal{M}} \sigma$ for all $\mathcal{M}, \mathbf{a}$	SATISFIABLE if $\mathbf{a} \models_{\mathcal{M}} \sigma$ for some $\mathcal{M}, \mathbf{a}$
INVALID if $\mathbf{a} \not\models_{\mathcal{M}} \sigma$ for some $\mathcal{M}, \mathbf{a}$	UNSATISFIABLE if $\mathbf{a} \not\models_{\mathcal{M}} \sigma$ for all $\mathcal{M}, \mathbf{a}$

Observations

σ is VALID iff σ is not INVALID

σ is SATISFIABLE iff σ is not UNSATISFIABLE

σ is VALID iff $\neg\sigma$ is UNSATISFIABLE

σ is SATISFIABLE iff $\neg\sigma$ is INVALID

Models

A *sentence* is a formula σ with no free variables.
A *first-order theory* is a set of sentences.

Definition — consistency

A theory $\mathcal{T}$ is *consistent* if $\mathcal{T} \nvdash \bot$.

Definition — model

A structure $\mathcal{M} = (M, I)$ is a *model* of a theory $\mathcal{T}$ if $\vDash_{\mathcal{M}} \sigma$ holds for every formula $\sigma \in \mathcal{T}$.
(The assignment is irrelevant since σ is closed).

Soundness and completeness

Theorem (Gödel, 1929)

Given a theory $\mathcal{T}$, the following are equivalent:

1. $\mathcal{T}$ is consistent.
2. $\mathcal{T}$ has (at least) one model.

Corollary

Given a formula σ, the following are equivalent:

1. $\vdash \sigma$ is derivable.
2. σ is valid.

Corollary

Given a formula σ, the following are equivalent:

1. $\vdash \neg\sigma$ is derivable.
2. σ is unsatisfiable.

Examples of validity and satisfiability

Example

Determine whether they are (in)valid/(un)satisfiable:

1. $\forall X.\, X = X$ — satisfiable and invalid
2. $\forall X.\, \mathbf{P}(X) \Rightarrow \forall X.\, \mathbf{P}(\mathtt{f}(X))$ — valid ($\therefore$ satisfiable)
3. $\forall X.\, \neg\mathbf{P}(X) \wedge \exists X.\, \mathbf{P}(X)$ — unsatisfiable ($\therefore$ invalid)
4. $\forall X.\, \exists Y.\, \mathbf{P}(X, Y) \Rightarrow \exists Y.\, \forall X.\, \mathbf{P}(X, Y)$ — satisfiable and invalid
5. $\forall X.\, (\mathbf{P}(X) \Rightarrow \sigma) \;\Rightarrow\; (\exists X.\, \mathbf{P}(X)) \Rightarrow \sigma$ with $X \notin \mathrm{fv}(\sigma)$ — valid

The decision problem

We would like an algorithm that solves the following problem:

Input:	a formula σ.
Output:	a boolean indicating whether σ is valid.

It is **not** possible to give an algorithm that meets this specification.

Introduction

Syntax of first-order logic

Natural deduction for first-order logic

Semantics of first-order logic

Unification of terms

Unification algorithm

The unification algorithm we knew adapts to first-order terms just by changing the notation:

$$\{X \overset{?}{=} X\} \cup E \xrightarrow{\texttt{Delete}} E$$

$$\{\texttt{f}(t_1, \ldots, t_n) \overset{?}{=} \texttt{f}(s_1, \ldots, s_n)\} \cup E \xrightarrow{\texttt{Decompose}} \{t_1 \overset{?}{=} s_1, \ldots, t_n \overset{?}{=} s_n\} \cup E$$

$$\{t \overset{?}{=} X\} \cup E \xrightarrow{\texttt{Swap}} \{X \overset{?}{=} t\} \cup E$$

if t is not a variable

$$\{X \overset{?}{=} t\} \cup E \xrightarrow{\texttt{Elim}}_{\{X:=t\}} E\{X:=t\}$$

if $X \notin \mathsf{fv}(t)$

$$\{\texttt{f}(t_1, \ldots, t_n) \overset{?}{=} \mathsf{g}(s_1, \ldots, s_m)\} \cup E \xrightarrow{\texttt{Clash}} \texttt{failure}$$

if $\texttt{f} \neq \mathsf{g}$

$$\{X \overset{?}{=} t\} \cup E \xrightarrow{\texttt{Occurs-Check}} \texttt{failure}$$

if $X \neq t$ and $X \in \mathsf{fv}(t)$

Termination of the unification algorithm

Given a set of unification equations E, we define:

- n_1: number of distinct variables in E
- n_2: size of E, calculated as $\sum_{(t \stackrel{?}{=} s) \in E} |t| + |s|$
- n_3: number of equations of the form $t \stackrel{?}{=} X$ in E

We can observe that the rules that do not produce failure decrease the triple (n_1, n_2, n_3), according to the *lexicographic order*:

	n_1	n_2	n_3
`Elim`	$>$		
`Decompose`	$=$	$>$	
`Delete`	$\geq$	$>$	
`Swap`	$=$	$=$	$>$

Correctness of the unification algorithm

Recall

1. A **substitution** is a function $\mathbf{S}$
   that associates a term $\mathbf{S}(X)$ with each variable X.
2. $\mathbf{S}$ is a **unifier** of E
   if for each $(t \overset{?}{=} s) \in E$ we have $\mathbf{S}(t) = \mathbf{S}(s)$.
3. $\mathbf{S}$ is **more general** than $\mathbf{S}'$
   if there exists $\mathbf{T}$ such that $\mathbf{S}' = \mathbf{T} \circ \mathbf{S}$.
4. $\mathbf{S}$ is an **m.g.u.** of E if $\mathbf{S}$ is a unifier of E
   and for every unifier $\mathbf{S}'$ of E
   we have that $\mathbf{S}$ is more general than $\mathbf{S}'$.
   Technically, we are interested in **idempotent** m.g.u.'s,
   i.e., $\mathbf{S}(\mathbf{S}(t)) = \mathbf{S}(t)$ for every term t.

Correctness of the unification algorithm

Lemma — correctness of the $\texttt{Delete}$ rule

$\mathbf{S}$ m.g.u. of $E \implies \mathbf{S}$ m.g.u. of $\{X \stackrel{?}{=} X\} \cup E$.

Lemma — correctness of the $\texttt{Swap}$ rule

$\mathbf{S}$ m.g.u. of $\{t \stackrel{?}{=} s\} \cup E \implies \mathbf{S}$ m.g.u. of $\{s \stackrel{?}{=} t\} \cup E$.

Lemma — correctness of the $\texttt{Decompose}$ rule

$\mathbf{S}$ m.g.u. of $\{t_1 \stackrel{?}{=} s_1, \ldots, t_n \stackrel{?}{=} s_n\} \cup E$

$\implies \mathbf{S}$ m.g.u. of $\{\mathtt{f}(t_1, \ldots, t_n) \stackrel{?}{=} \mathtt{f}(s_1, \ldots, s_n)\} \cup E$.

Lemma — correctness of the $\texttt{Elim}$ rule

$\mathbf{S}$ m.g.u. of $E\{X := t\}$ and $X \notin \mathrm{fv}(t)$

$\implies \mathbf{S} \circ \{X := t\}$ m.g.u. of E.

Use the fact that if $\mathbf{S}(X) = t$ then $\mathbf{S}(s\{X := t\}) = \mathbf{S}(s)$.

Correctness of the unification algorithm

Let's prove correctness in case of success.

Let $E_0 \rightarrow_{\mathbf{S}_1} E_1 \rightarrow_{\mathbf{S}_n} E_2 \rightarrow \ldots \rightarrow_{\mathbf{S}_n} E_n = \varnothing$.
We need to show that $\mathbf{S}_n \circ \ldots \circ \mathbf{S}_1$ is an m.g.u. of E.
By induction on n:

1. If $n = 0$, the identity substitution is an m.g.u. of $\varnothing$.
2. If $n > 0$, we have:

$$E_0 \rightarrow_{\mathbf{S}_1} E_1 \qquad E_1 \rightarrow_{\mathbf{S}_2} \ldots \rightarrow_{\mathbf{S}_n} E_n = \varnothing$$

   By IH, $\mathbf{S}_n \circ \ldots \circ \mathbf{S}_2$ is an m.g.u. of E_1.
   Applying one of the previous lemmas, we conclude that
   $\mathbf{S}_n \circ \ldots \circ \mathbf{S}_2 \circ \mathbf{S}_1$ is an m.g.u. of E_0.

Correctness of the unification algorithm

Correctness in case of failure is proved similarly,
with lemmas that go “forward” instead of “backward”.

¿ ¿ ¿ ¿ ¿ ¿ ¿ ¿ ¿ ? ? ? ? ? ? ? ? ?

Recommended reading

Chapter 2 of van Dalen's book.

Dirk van Dalen. *Logic and Structure* (Fifth Edition). Springer-Verlag, 2013.