Chapter 3

First-Order Logic

First-Order logic is a generalization of propositional logic. Propositional logic can represent propositions, whereas first-order logic can represent individuals and propositions about individuals. For example, in propositional logic from "Socrates is a man" and "If Socrates is a man then Socrates is mortal" the conclusion "Socrates is mortal" can be drawn. In first-order logic this can be represented much more fine-grained. From "Socrates is a man" and "All man are mortal" the conclusion "Socrates is mortal" can be drawn.

This chapter introduces first-order logic with equality. However, all calculi presented here, namely Tableau and Free-Variable Tableau (Sections 3.6, 3.8), Resolution (Section 3.10), and Superposition (Section 3.12) are presented only for its restriction without equality. Purely equational logic and first-order logic with equality are presented separately in Chapter 4 and Chapter 5, respectively.

3.1 Syntax

Most textbooks introduce first-order logic in an unsorted way. Like in programming languages, sorts support distinguishing "apples from oranges" and therefore move part of the reasoning to a more complex syntax of formulas. Many-sorted logic is a generalization of unsorted first-order logic where the universe is separated into disjoint sets of objects, called *sorts*. Functions and predicates are defined with respect to these sorts in a unique way. The resulting language: many-sorted first-order logic has a very simple, but already useful sort structure, sometimes also called *type* structure. It can distinguish apples from oranges by providing two different, respective sorts, but it cannot express relationships between sorts. For example, it cannot express the integers to be a subsort of the reals, because all sorts are assumed to be disjoint. On the other hand, the simple many-sorted language comes at no extra cost when considering inference or simplification rules, whereas more expressive sort languages need extra and sometimes costly reasoning.

Definition 3.1.1 (Many-Sorted Signature)**.** A *many-sorted signature* $\Sigma = (\mathcal{S}, \Omega, \Pi)$ is a triple consisting of a finite non-empty set $\mathcal{S}$ of *sort symbols*, a non-empty set Ω of *operator symbols* (also called *function symbols*) over $\mathcal{S}$ and a set Π of *predicate symbols*. Every operator symbol $f \in \Omega$ has a unique sort declaration $f : S_1 \times \ldots \times S_n \to S$, indicating the sorts of arguments (also called *domain sorts*) and the *range sort* of f, respectively, for some $S_1, \ldots, S_n, S \in \mathcal{S}$ where $n \geq 0$ is called the *arity* of f, also denoted with $\operatorname{arity}(f)$. An operator symbol $f \in \Omega$ with arity 0 is called a *constant*. Every predicate symbol $P \in \Pi$ has a unique sort declaration $P \subseteq S_1 \times \ldots \times S_n$. A predicate symbol $P \in \Pi$ with arity 0 is called a *propositional variable*. For every sort $S \in \mathcal{S}$ there must be at least one constant $a \in \Omega$ with range sort S.

In addition to the signature Σ, a variable set $\mathcal{X}$, disjoint from Ω is assumed, so that for every sort $S \in \mathcal{S}$ there exists a countably infinite subset of $\mathcal{X}$ consisting of variables of the sort S. A variable x of sort S is denoted by x_S.

Definition 3.1.2 (Term)**.** Given a signature $\Sigma = (\mathcal{S}, \Omega, \Pi)$, a sort $S \in \mathcal{S}$ and a variable set $\mathcal{X}$, the set $T_S(\Sigma, \mathcal{X})$ of all *terms* of sort S is recursively defined by (i) $x_S \in T_S(\Sigma, \mathcal{X})$ if $x_S \in \mathcal{X}$, (ii) $f(t_1, \ldots, t_n) \in T_S(\Sigma, \mathcal{X})$ if $f \in \Omega$ and $f : S_1 \times \ldots \times S_n \to S$ and $t_i \in T_{S_i}(\Sigma, \mathcal{X})$ for every $i \in \{1, \ldots, n\}$.

The sort of a term t is denoted by $\operatorname{sort}(t)$, i.e., if $t \in T_S(\Sigma, \mathcal{X})$ then $\operatorname{sort}(t) = S$. A term not containing a variable is called *ground*.

For the sake of simplicity it is often written: $T(\Sigma, \mathcal{X})$ for $\bigcup_{S \in \mathcal{S}} T_S(\Sigma, \mathcal{X})$, the set of all terms, $T_S(\Sigma)$ for the set of all ground terms of sort $S \in \mathcal{S}$, and $T(\Sigma)$ for $\bigcup_{S \in \mathcal{S}} T_S(\Sigma)$, the set of all ground terms over Σ.

A term t is called *shallow* if t is of the form $f(x_1, \ldots, x_n)$. A term t is called *linear* if every variable occurs at most once in t.

Note that the sets $T_S(\Sigma)$ are all non-empty, because there is at least one constant for each sort S in Σ. The sets $T_S(\Sigma, \mathcal{X})$ include infinitely many variables of sort S.

Definition 3.1.3 (Equation, Atom, Literal)**.** If $s, t \in T_S(\Sigma, \mathcal{X})$ then $s \approx t$ is an *equation* over the signature Σ. Any equation is an *atom* (also called *atomic formula*) as well as every $P(t_1, \ldots, t_n)$ where $t_i \in T_{S_i}(\Sigma, \mathcal{X})$ for every $i \in \{1, \ldots, n\}$ and $P \in \Pi$, $\operatorname{arity}(P) = n$, $P \subseteq S_1 \times \ldots \times S_n$. An atom or its negation of an atom is called a *literal*.

The literal $s \mathrel{\dot{\approx}} t$ denotes either $s \approx t$ or $t \approx s$. A literal is *positive* if it is an atom and *negative* otherwise. A negative equational literal $\neg(s \approx t)$ is written as $s \not\approx t$.

C Non equational atoms can be transformed into equations: For this a given signature is extended for every predicate symbol P as follows: (i) add a distinct sort Bool to $\mathcal{S}$, (ii) introduce a fresh constant true of the sort Bool to Ω, (iii) for every predicate P, $P \subseteq S_1 \times \ldots \times S_n$ add a fresh function $f_P : S_1, \ldots, S_n \to \text{Bool}$ to Ω, and (iv) encode every atom $P(t_1, \ldots, t_n)$ as an equation $f_P(t_1, \ldots, t_n) \approx \text{true}$, see Section 3.4. Definition 3.1.3 implicitly overloads the equality symbol for all sorts S. An alternative would be to have a separate equality symbol for each sort.

Definition 3.1.4 (Formulas)**.** The set $\mathrm{FOL}(\Sigma, \mathcal{X})$ of *many-sorted first-order formulas with equality* over the signature Σ is defined as follows for formulas $\phi, \psi \in F_\Sigma(\mathcal{X})$ and a variable $x \in \mathcal{X}$:

$\mathrm{FOL}(\Sigma, \mathcal{X})$	Comment
$\bot$	false
$\top$	true
$P(t_1, \ldots, t_n), s \approx t$	atom
$(\neg\phi)$	negation
$(\phi \wedge \psi)$	conjunction
$(\phi \vee \psi)$	disjunction
$(\phi \rightarrow \psi)$	implication
$(\phi \leftrightarrow \psi)$	equivalence
$\forall x.\phi$	universal quantification
$\exists x.\phi$	existential quantification

A consequence of the above definition is that $\mathrm{PROP}(\Sigma) \subseteq \mathrm{FOL}(\Sigma', \mathcal{X})$ if the propositional variables of Σ are contained in Σ' as predicates of arity 0. A formula not containing a quantifier is called *quantifier-free*.

Definition 3.1.5 (Positions)**.** It follows from the definitions of terms and formulas that they have a tree-like structure. For referring to a certain subtree, called subterm or subformula, respectively, sequences of natural numbers are used, called *positions* (as introduced in Chapter 2.1.3). The set of positions of a term, formula is inductively defined by:

$$\begin{array}{rl}
\mathrm{pos}(x) & := \{\epsilon\} \text{ if } x \in \mathcal{X} \\
\mathrm{pos}(\phi) & := \{\epsilon\} \text{ if } \phi \in \{\top, \bot\} \\
\mathrm{pos}(\neg\phi) & := \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\} \\
\mathrm{pos}(\phi \circ \psi) & := \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\} \cup \{2p \mid p \in \mathrm{pos}(\psi)\} \\
\mathrm{pos}(s \approx t) & := \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(s)\} \cup \{2p \mid p \in \mathrm{pos}(t)\} \\
\mathrm{pos}(f(t_1, \ldots, t_n)) & := \{\epsilon\} \cup \bigcup_{i=1}^{n} \{ip \mid p \in pos(t_i)\} \\
\mathrm{pos}(P(t_1, \ldots, t_n)) & := \{\epsilon\} \cup \bigcup_{i=1}^{n} \{ip \mid p \in pos(t_i)\} \\
\mathrm{pos}(\forall x.\phi) & := \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\} \\
\mathrm{pos}(\exists x.\phi) & := \{\epsilon\} \cup \{1p \mid p \in \mathrm{pos}(\phi)\}
\end{array}$$

where $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ and $t_i \in T(\Sigma, \mathcal{X})$ for all $i \in \{1, \ldots, n\}$.

The *prefix orders* (above, strictly above and parallel), the selection and replacement with respect to positions are defined exactly as in Chapter 2.1.3.

An term t (formula ϕ) is said to *contain* another term s (formula ψ) if $t|_p = s$ ($\phi|_p = \psi$). It is called a *strict subexpression* if $p \neq \epsilon$. The term t (formula ϕ) is called an *immediate subexpression* of s (formula ψ) if $|p| = 1$. For terms a subexpression is called a *subterm* and for formulas a *subformula*, respectively.

The *size* of a term t (formula ϕ), written $|t|$ ($|\phi|$), is the cardinality of $\mathrm{pos}(t)$, i.e., $|t| := |\mathrm{pos}(t)|$ ($|\phi| := |\mathrm{pos}(\phi)|$). The *depth* of a term, formula is the maximal length of a position in the term, formula: $\text{depth}(t) := max\{|p| \mid p \in \text{pos}(t)\}$ ($\text{depth}(\phi) := max\{|p| \mid p \in \text{pos}(\phi)\}$).

The set of *all* variables occurring in a term t (formula ϕ) is denoted by $\text{vars}(t)$ ($\text{vars}(\phi)$) and formally defined as $\text{vars}(t) := \{x \in \mathcal{X} \mid x = t|_p, p \in \text{pos}(t)\}$ ($\text{vars}(\phi) := \{x \in \mathcal{X} \mid x = \phi|_p, p \in \text{pos}(\phi)\}$). A term t (formula ϕ) is *ground* if $vars(t) = \emptyset$ ($vars(\phi) = \emptyset$). Note that $\text{vars}(\forall x.a \approx b) = \emptyset$ where a, b are constants. This is justified by the fact that the formula does not depend on the quantifier, see the semantics below. The set of *free* variables of a formula ϕ (term t) is given by $\text{fvars}(\phi, \emptyset)$ ($\text{fvars}(t, \emptyset)$) and recursively defined by $\text{fvars}(\psi_1 \circ \psi_2, B) := \text{fvars}(\psi_1, B) \cup \text{fvars}(\psi_2, B)$ where $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$, $\text{fvars}(\forall x.\psi, B) := \text{fvars}(\psi, B \cup \{x\})$, $\text{fvars}(\exists x.\psi, B) := \text{fvars}(\psi, B \cup \{x\})$, $\text{fvars}(\neg\psi, B) := \text{fvars}(\psi, B)$, $\text{fvars}(L, B) := \text{vars}(L) \backslash B$ ($\text{fvars}(t, B) := \text{vars}(t) \backslash B$. For $\text{fvars}(\phi, \emptyset)$ I also write $\text{fvars}(\phi)$.

The function top maps terms to their top symbols, i.e., $\text{top}(f(t_1, \ldots, t_n)) := f$ and $\text{top}(x) := x$ for some variable x.

In $\forall x.\phi$ ($\exists x.\phi$) the formula ϕ is called the *scope* of the quantifier. An occurrence q of a variable x in a formula ϕ ($\phi|_q = x$) is called *bound* if there is some $p < q$ with $\phi|_p = \forall x.\phi'$ or $\phi|_p = \exists x.\phi'$. Any other occurrence of a variable is called *free*. A formula not containing a free occurrence of a variable is called *closed*. If $\{x_1, \ldots, x_n\}$ are the variables freely occurring in a formula ϕ then $\forall x_1, \ldots, x_n.\phi$ and $\exists x_1, \ldots, x_n.\phi$ (abbreviations for $\forall x_1.\forall x_2 \ldots \forall x_n.\phi$, $\exists x_1.\exists x_2 \ldots \exists x_n.\phi$, respectively) are the *universal* and the *existential closure* of ϕ, respectively.

Example 3.1.6. For the literal $\neg P(f(x, g(a)))$ the atom $P(f(x, g(a)))$ is an immediate subformula occurring at position 1. The terms x and $g(a)$ are strict subterms occurring at positions 111 and 112, respectively. The formula $\neg P(f(x, g(a)))[b]_{111} = \neg P(f(b, g(a)))$ is obtained by replacing x with b. $\text{pos}(\neg P(f(x, g(a)))) = \{\epsilon, 1, 11, 111, 112, 1121\}$ meaning its size is 6, its depth 4 and $vars(\neg P(f(x, g(a)))) = \{x\}$.

Definition 3.1.7 (Polarity)**.** The *polarity* of a subformula $\psi = \phi|_p$ at position p is $pol(\phi, p)$ where pol is recursively defined by

$$
\begin{array}{rl}
\text{pol}(\phi, \epsilon) & := 1 \\
\text{pol}(\neg\phi, 1p) & := -\text{pol}(\phi, p) \\
\text{pol}(\phi_1 \circ \phi_2, ip) & := \text{pol}(\phi_i, p) \text{ if } \circ \in \{\wedge, \vee\} \\
\text{pol}(\phi_1 \rightarrow \phi_2, 1p) & := -\text{pol}(\phi_1, p) \\
\text{pol}(\phi_1 \rightarrow \phi_2, 2p) & := \text{pol}(\phi_2, p) \\
\text{pol}(\phi_1 \leftrightarrow \phi_2, ip) & := 0 \\
\text{pol}(P(t_1, \ldots, t_n), p) & := 1 \\
\text{pol}(t \approx s, p) & := 1 \\
\text{pol}(\forall x.\phi, 1p) & := \text{pol}(\phi, p) \\
\text{pol}(\exists x.\phi, 1p) & := \text{pol}(\phi, p)
\end{array}
$$

3.2 Semantics

Definition 3.2.1 (Σ-algebra)**.** Let $\Sigma = (\mathcal{S}, \Omega, \Pi)$ be a signature with set of sorts $\mathcal{S}$, operator set Ω and predicate set Π. A Σ-*algebra* $\mathcal{A}$, also called Σ-*interpretation*, is a mapping that assigns (i) a non-empty carrier set $S^{\mathcal{A}}$ to every sort $S \in \mathcal{S}$, so that $(S_1)^{\mathcal{A}} \cap (S_2)^{\mathcal{A}} = \emptyset$ for any distinct sorts $S_1, S_2 \in \mathcal{S}$, (ii) a total function $f^{\mathcal{A}} : (S_1)^{\mathcal{A}} \times \ldots \times (S_n)^{\mathcal{A}} \to (S)^{\mathcal{A}}$ to every operator $f \in \Omega$, $\operatorname{arity}(f) = n$ where $f : S_1 \times \ldots \times S_n \to S$, (iii) a relation $P^{\mathcal{A}} \subseteq ((S_1)^{\mathcal{A}} \times \ldots \times (S_m)^{\mathcal{A}})$ to every predicate symbol $P \in \Pi$, $\operatorname{arity}(P) = m$. (iv) the equality relation becomes $\approx^{\mathcal{A}} = \{(e, e) \mid e \in \mathcal{U}^{\mathcal{A}}\}$ where the set $\mathcal{U}^{\mathcal{A}} := \bigcup_{S \in \mathcal{S}} (S)^{\mathcal{A}}$ is called the *universe* of $\mathcal{A}$.

A (variable) *assignment*, also called a *valuation* for an algebra $\mathcal{A}$ is a function $\beta : \mathcal{X} \to \mathcal{U}_{\mathcal{A}}$ so that $\beta(x) \in S_{\mathcal{A}}$ for every variable $x \in \mathcal{X}$, where $S = \operatorname{sort}(x)$. A *modification* $\beta[x \mapsto e]$ of an assignment β at a variable $x \in \mathcal{X}$, where $e \in S_{\mathcal{A}}$ and $S = \operatorname{sort}(x)$, is the assignment defined as follows:

$$\beta[x \mapsto e](y) = \begin{cases} e & \text{if } x = y \\ \beta(y) & \text{otherwise.} \end{cases}$$

Informally speaking, the assignment $\beta[x \mapsto e]$ is identical to β for every variable except x, which is mapped by $\beta[x \mapsto e]$ to e.

The homomorphic extension $\mathcal{A}(\beta)$ of β onto terms is a mapping $T(\Sigma, \mathcal{X}) \to \mathcal{U}_{\mathcal{A}}$ defined as (i) $\mathcal{A}(\beta)(x) = \beta(x)$, where $x \in \mathcal{X}$ and (ii) $\mathcal{A}(\beta)(f(t_1, \ldots, t_n)) = f_{\mathcal{A}}(\mathcal{A}(\beta)(t_1), \ldots, \mathcal{A}(\beta)(t_n))$, where $f \in \Omega$, $\operatorname{arity}(f) = n$.

Given a term $t \in T(\Sigma, \mathcal{X})$, the value $\mathcal{A}(\beta)(t)$ is called the *interpretation* of t under $\mathcal{A}$ and β. If the term t is ground, the value $\mathcal{A}(\beta)(t)$ does not depend on a particular choice of β, for which reason the interpretation of t under $\mathcal{A}$ is denoted by $\mathcal{A}(t)$.

An algebra $\mathcal{A}$ is called *term-generated*, if every element e of the universe $\mathcal{U}_{\mathcal{A}}$ of $\mathcal{A}$ is the image of some ground term t, i.e., $\mathcal{A}(t) = e$.

Definition 3.2.2 (Semantics)**.** An algebra $\mathcal{A}$ and an assignment β are extended to formulas $\phi \in \operatorname{FOL}(\Sigma, \mathcal{X})$ by

$$\begin{array}{rcl}
\mathcal{A}(\beta)(\bot) & := & 0 \\
\mathcal{A}(\beta)(\top) & := & 1 \\
\mathcal{A}(\beta)(s \approx t) & := & 1 \text{ if } \mathcal{A}(\beta)(s) = \mathcal{A}(\beta)(t) \text{ and } 0 \text{ otherwise} \\
\mathcal{A}(\beta)(P(t_1, \ldots, t_n)) & := & 1 \text{ if } (\mathcal{A}(\beta)(t_1), \ldots, \mathcal{A}(\beta)(t_n)) \in P^{\mathcal{A}} \text{ and } 0 \text{ otherwise} \\
\mathcal{A}(\beta)(\neg\phi) & := & 1 - \mathcal{A}(\beta)(\phi) \\
\mathcal{A}(\beta)(\phi \wedge \psi) & := & \min(\{\mathcal{A}(\beta)(\phi), \mathcal{A}(\beta)(\psi)\}) \\
\mathcal{A}(\beta)(\phi \vee \psi) & := & \max(\{\mathcal{A}(\beta)(\phi), \mathcal{A}(\beta)(\psi)\}) \\
\mathcal{A}(\beta)(\phi \to \psi) & := & \max(\{(1 - \mathcal{A}(\beta)(\phi)), \mathcal{A}(\beta)(\psi)\}) \\
\mathcal{A}(\beta)(\phi \leftrightarrow \psi) & := & \text{if } \mathcal{A}(\beta)(\phi) = \mathcal{A}(\beta)(\psi) \text{ then } 1 \text{ else } 0 \\
\mathcal{A}(\beta)(\exists x_S.\phi) & := & 1 \text{ if } \mathcal{A}(\beta[x \mapsto e])(\phi) = 1 \text{ for some } e \in S_{\mathcal{A}} \text{ and } 0 \text{ otherwise} \\
\mathcal{A}(\beta)(\forall x_S.\phi) & := & 1 \text{ if } \mathcal{A}(\beta[x \mapsto e])(\phi) = 1 \text{ for all } e \in S_{\mathcal{A}} \text{ and } 0 \text{ otherwise}
\end{array}$$

A formula ϕ is called *satisfiable by* $\mathcal{A}$ *under* β (or *valid in* $\mathcal{A}$ *under* β) if $\mathcal{A}, \beta \models \phi$; in this case, ϕ is also called *consistent*; *satisfiable by* $\mathcal{A}$ if $\mathcal{A}, \beta \models \phi$ for some assignment β; *satisfiable* if $\mathcal{A}, \beta \models \phi$ for some algebra $\mathcal{A}$ and some assignment β; *valid in* $\mathcal{A}$, written $\mathcal{A} \models \phi$, if $\mathcal{A}, \beta \models \phi$ for any assignment β; in this case, $\mathcal{A}$ is called a *model* of ϕ; *valid*, written $\models \phi$, if $\mathcal{A}, \beta \models \phi$ for any algebra $\mathcal{A}$ and any assignment β; in this case, ϕ is also called a *tautology*; *unsatisfiable* if $\mathcal{A}, \beta \not\models \phi$ for any algebra $\mathcal{A}$ and any assignment β; in this case ϕ is also called *inconsistent*.

Note that $\bot$ is inconsistent whereas $\top$ is valid. If ϕ is a sentence that is a formula not containing a free variable, it is valid in $\mathcal{A}$ if and only if it is satisfiable by $\mathcal{A}$. This means the truth of a sentence does not depend on the choice of an assignment.

Given two formulas ϕ and ψ, ϕ *entails* ψ, or ψ is a *consequence* of ϕ, written $\phi \models \psi$, if for any algebra $\mathcal{A}$ and assignment β, if $\mathcal{A}, \beta \models \phi$ then $\mathcal{A}, \beta \models \psi$. The formulas ϕ and ψ are called *equivalent*, written $\phi \models\!\mid \psi$, if $\phi \models \psi$ and $\psi \models \phi$. Two formulas ϕ and ψ are called *equisatisfiable*, if ϕ is satisfiable iff ψ is satisfiable (not necessarily in the same models). Note that if ϕ and ψ are equivalent then they are equisatisfiable, but not the other way around. The notions of "entailment", "equivalence" and "equisatisfiability" are naturally extended to sets of formulas, that are treated as conjunctions of single formulas. Thus, given formula sets M_1 and M_2, the set M_1 entails M_2, written $M_1 \models M_2$, if for any algebra $\mathcal{A}$ and assignment β, if $\mathcal{A}, \beta \models \phi$ for every $\phi \in M_1$ then $\mathcal{A}, \beta \models \psi$ for every $\psi \in M_2$. The sets M_1 and M_2 are equivalent, written $M_1 \models\!\mid M_2$, if $M_1 \models M_2$ and $M_2 \models M_1$. Given an arbitrary formula ϕ and formula set M, $M \models \phi$ is written to denote $M \models \{\phi\}$; analogously, $\phi \models M$ stands for $\{\phi\} \models M$.

Clauses are implicitly universally quantified disjunctions of literals. A clause C is satisfiable by an algebra $\mathcal{A}$ if for every assignment β there is a literal $L \in C$ with $\mathcal{A}, \beta \models L$. Note that if $C = \{L_1, \ldots, L_k\}$ is a ground clause, i.e., every L_i is a ground literal, then $\mathcal{A} \models C$ if and only if there is a literal L_j in C so that $\mathcal{A} \models L_j$. A clause set N is satisfiable iff all clauses $C \in N$ are satisfiable by the same algebra $\mathcal{A}$. Accordingly, if N and M are two clause sets, $N \models M$ iff every model $\mathcal{A}$ of N is also a model of M.

Definition 3.2.3 (Congruence)**.** Let $\Sigma = (\mathcal{S}, \Omega, \Pi)$ be a signature and $\mathcal{A}$ a Σ-algebra. A *congruence* $\sim$ is an equivalence relation on $(S_1)^{\mathcal{A}} \cup \ldots \cup (S_n)^{\mathcal{A}}$ such that

1. if $a \sim b$ then there is an $S \in \mathcal{S}$ such that $a \in S^{\mathcal{A}}$ and $b \in S^{\mathcal{A}}$
2. for all $a_i \sim b_i$, $a_i, b_i \in (S_i)^{\mathcal{A}}$ and all functions $f : S_1 \times \ldots \times S_n \to S$ it holds $f^{\mathcal{A}}(a_1, \ldots, a_n) \sim f^{\mathcal{A}}(b_1, \ldots, b_n)$
3. for all $a_i \sim b_i$, $a_i, b_i \in (S_i)^{\mathcal{A}}$ and all predicates $P \subseteq S_1 \times \ldots \times S_n$ it holds $(a_1, \ldots, a_n) \in P^{\mathcal{A}}$ iff $(b_1, \ldots, b_n) \in P^{\mathcal{A}}$

The first condition guarantees that a congruence $\sim$ respects the disjoint sort structure. The second requires compatibility with function applications and the third compatibility with predicate definitions. Actually, for any Σ-algebra $\mathcal{A}$ the