***Isomorphism and homomorphism.*** The satisfaction relation $\models$ does not distinguish between isomorphic structures, i.e., structures that coincide up to renaming of the elements. Formally, two structures $\mathcal{A}$ and $\mathcal{B}$ for the same vocabulary *Voc* are called *isomorphic* if there exists a bijection $h : A \to B$ (where $A$ and $B$ are the domains of $\mathcal{A}$ and $\mathcal{B}$, respectively) such that

- $P^{\mathcal{B}} = \big\{ (h(a_1),\ldots,h(a_n)) : (a_1,\ldots,a_n) \in P^{\mathcal{A}} \big\}$ for all $P \in \mathit{Pred}_n$,
- $f^{\mathcal{B}}\big(h(a_1),\ldots,h(a_m)\big) = h\big(f^{\mathcal{A}}(a_1,\ldots,a_m)\big)$ for all $a_1,\ldots,a_m \in A$ and $f \in \mathit{Func}_m$.

In this case, $h$ is called an *isomorphism* from $\mathcal{A}$ to $\mathcal{B}$. Clearly, we then have:

$$(\mathcal{A},\mathcal{V}) \models \phi \quad \text{iff} \quad (\mathcal{B}, h \circ \mathcal{V}) \models \phi$$

for all formulas $\phi$. In particular, $\mathcal{A}$ and $\mathcal{B}$ are models for exactly the same FOL-sentences with or without equality. The concept of isomorphism between structures is rather strong since only those structures are identified that agree up to renaming the elements of the domain. To ensure the equivalence of two structures with respect to the class of FOL-formulas without equality, a weaker relation between structures is sufficient. Let $\mathcal{A}$ and $\mathcal{B}$ structures for the same vocabulary *Voc*. Let $A$ and $B$ be the domains of $\mathcal{A}$ and $\mathcal{B}$, respectively. A *homomorphism* from $\mathcal{A}$ to $\mathcal{B}$ is a function $h : A \to B$ such that:

- for all $n$-ary predicate symbols $P$ and $(a_1,\ldots,a_n) \in A^n$:

$$(a_1,\ldots,a_n) \in P^{\mathcal{A}} \quad \text{iff} \quad (h(a_1),\ldots,h(a_n)) \in P^{\mathcal{B}}$$

- for each $m$-ary function symbol $f$ and $(a_1,\ldots,a_m) \in A^m$:

$$h\big(f^{\mathcal{A}}(a_1,\ldots,a_m)\big) \;=\; f^{\mathcal{B}}\big(h(a_1),\ldots,h(a_m)\big)$$

Obviously, any isomorphism is also a homomorphism. For arbitrary homomorphisms, preservation of truth values for FOL-formulas is only guaranteed for quantifier-free FOL-formulas without equality. If, however, the given homomorphism $h$ from $\mathcal{A}$ to $\mathcal{B}$ is *surjective* (i.e., $h(A) = B$) then we have:

$$(\mathcal{A},\mathcal{V}) \models \phi \quad \text{iff} \quad (\mathcal{B}, h \circ \mathcal{V}) \models \phi$$

for each $\phi$ is a FOL-formula over *Voc* without equality and each valuation $\mathcal{V} : \mathit{Var} \to A$. The proof for this statement is left as an exercise. To see why surjectivity of $h$ is crucial, regard the equality-free FOL-formula $\phi = \forall x.\neg P(x)$ where $P$ is an unary predicate symbol. Let structures $\mathcal{A} = (A, P^{\mathcal{A}})$ and $\mathcal{B} = (B, P^{\mathcal{B}})$ be given by

$$A = \{0\},\ P^{\mathcal{A}} = \varnothing \quad \text{and} \quad B = \{0,1\},\ P^{\mathcal{B}} = \{1\}.$$

Then, $h : A \to B$, $h(0) = 0$ is a homomorphism from $\mathcal{A}$ to $\mathcal{B}$, but $\mathcal{A} \models \phi$, while $\mathcal{B} \not\models \phi$. Also the assumption that $\phi$ is a formula of FOL without equality is crucial. To illustrate this, consider the empty vocabulary and structures $\mathcal{A} = \{0,1\}$, $\mathcal{B} = \{0\}$ and the surjective homomorphism $h(0) = h(1) = 0$. Then:

$$\mathcal{A} \models \forall x \exists y.(x \neq y), \quad \mathcal{B} \not\models \forall x \exists y.(x \neq y)$$

**Positive normal forms (PNF).** In some applications, it is useful to work with formulas in *positive normal form* (PNF), sometimes also called *negation normal form*. These are almost negation-free formulas: negations must not be used in front of formulas of length 1 or more, but negations may appear on the level of literals, i.e., in front of atomic formulas $P(t_1,\ldots,t_n)$ and $t_1 = t_2$. To guarantee that all FOL-formulas have equivalent PNF-formulas, for each FOL-operator we need its dual in the syntax. That is, both conjunction and disjunction and both universal and existential quantification are required as basic operators. For cosmetic reasons, we also add *false* as the dual of *true* to the basic syntax of PNF-formulas. Thus, the abstract syntax of FOL-formulas in PNF (briefly PNF-formulas) is given by:

$$\phi \;::=\; \mathit{true} \mid \mathit{false} \mid P(\bar{t}) \mid \neg P(\bar{t}) \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \forall x.\phi \mid \exists x.\phi \mid \underbrace{t_1 = t_2 \mid t_1 \neq t_2}_{\text{optional}}$$

Here, $\bar{t}$ is a tuple of terms of length $n$ where $n$ is the arity of $P$. Using the equivalence rules

$$\begin{aligned} \neg \mathit{true} &\equiv \mathit{false} \\ \neg\neg\phi &\equiv \phi \\ \neg(\phi_1 \wedge \phi_2) &\equiv \neg\phi_1 \vee \neg\phi_2 \\ \neg\forall x.\phi &\equiv \exists x.\neg\phi \end{aligned}$$

as rewrite rules each FOL-formula $\phi$ can be transformed into an equivalent PNF-formula of the same asymptotic (word-)length. Recall that our syntax uses only the logical symbols $\wedge$, $\neg$ and $\forall$, while disjunction is defined by $\phi_1 \vee \phi_2 = \neg(\neg\phi_1 \wedge \neg\phi_2)$ and existential quantification by $\exists x.\phi = \neg\forall x.\neg\phi$. Hence, the rules

$$\neg(\phi_1 \vee \phi_2) \;\equiv\; \neg\phi_1 \wedge \neg\phi_2 \quad \text{and} \quad \neg\exists x.\phi \equiv \forall x.\neg\phi$$

arise by the rule for double negation. Here is an example for the transformation:

$$\begin{aligned} & \neg\forall x.\big(P(x,a) \wedge (Q(x) \vee \exists y.R(x,f(y)))\big) \\ \equiv\; & \exists x.\neg\big(P(x,a) \wedge (Q(x) \vee \exists y.R(x,f(y)))\big) \\ \equiv\; & \exists x.\big(\neg P(x,a) \vee \neg(Q(x) \vee \exists y.R(x,f(y)))\big) \\ \equiv\; & \exists x.\big(\neg P(x,a) \vee (\neg Q(x) \wedge \neg\exists y.R(x,f(y)))\big) \\ \equiv\; & \exists x.\big(\neg P(x,a) \vee (\neg Q(x) \wedge \forall y.\neg R(x,f(y)))\big) \end{aligned}$$

**Prenex and Skolem form.** A FOL-formula is in *prenex form* (briefly called a prenex FOL-formula) if it is a formula of the type $Q_1x_1\ldots Q_nx_n.\psi$, consisting of a quantifier prefix and a quantifier-free formula $\psi$. Here, the $Q_i$'s are quantifiers, i.e., $Q_i \in \{\forall,\exists\}$, and the variables $x_1,\ldots,x_n$ are pairwise distinct. Subformula $\psi$ is called the *matrix*. An algorithmic transformation from a given FOL-formula $\phi$ into an equivalent prenex FOL-formula is obtained by

(1) constructing an equivalent PNF-formula $\phi_{\text{PNF}}$,

(2) applying the concept of renaming bounded variables to transform $\phi_{\text{PNF}}$ into an equivalent PNF-formula $\phi'_{\text{PNF}}$ of the same length such that no variable is bounded by two or more quantifiers in $\phi'_{\text{PNF}}$ and no variable has both free and bounded occurrences in $\phi'_{\text{PNF}}$,

(3) transforming $\phi'_{\text{PNF}}$ into an equivalent prenex formula $\phi_{\text{prenex}}$ by moving all quantifiers to the left.

Step (3) is justified by the equivalences

$$\left.\begin{array}{rcl}(\forall x.\phi)\wedge\psi & \equiv & \forall x.(\phi\wedge\psi)\\ (\exists x.\phi)\wedge\psi & \equiv & \exists x.(\phi\wedge\psi)\end{array}\right\}\ \text{if } x \text{ does not appear in } \psi$$

and the same rules for disjunction (rather than conjunction) and the symmetric rules. Obviously, the constructed prenex formula $\phi_{\text{prenex}}$ is of word-length $\mathcal{O}(\|\phi\|)$. E.g.,

$$\phi \;=\; \phi_{\text{PNF}} \;=\; \forall x.P(f(x),a)\vee\exists y\forall z\exists w.\neg R(y,z,w)$$

is equivalent to

$$\phi_{\text{prenex}} \;=\; \forall x\exists y\forall z\exists w.\big(P(f(x),a)\vee\neg R(y,z,w)\big).$$

A FOL-formula is said to be in *Skolem form* (briefly called Skolem formula) if it is in prenex form and all quantifiers are universal. Skolem forms can serve for satisfiability checks, since there is an algorithm that takes as input a FOL-formula $\phi$ and returns a Skolem formula $\phi_{\text{Skolem}}$ such that $\phi$ is satisfiable if and only if $\phi_{\text{Skolem}}$ is satisfiable. The algorithm works as follows. We first transform $\phi$ into an equivalent prenex FOL-formula $\phi_{\text{prenex}}$ and then remove the existential quantifiers in $\phi_{\text{prenex}}$ as follows. We consider the left most existential quantifier $\exists y$ in $\phi_{\text{prenex}}$. Thus, $\phi_{\text{prenex}}$ has the form $\forall x_1\ldots\forall x_k\exists y.\psi$. We then introduce a fresh $k$-ary function symbol $f$ and replace $\phi_{\text{prenex}}$ with

$$\forall x_1\ldots\forall x_k.\psi[y/f(x_1,\ldots,x_k)].$$

Now we recursively apply the transformation to $\psi[y/f(x_1,\ldots,x_k)]$ and finally obtain a formula $\phi_{\text{Skolem}}$ in Skolem form (over an extended vocabulary) of word-length $\mathcal{O}(\|\phi\|^2)$. For instance,

$$\phi \;=\; \phi_{\text{prenex}} \;=\; \exists y\forall x_1\forall x_2\exists z.P(y,x_1,x_2,z)$$

is replaced with

$$\phi_{\text{Skolem}} \;=\; \forall x_1\forall x_2.P(c,x_1,x_2,f(x_1,x_2))$$

where $c$ is a new constant symbol and $f$ a new binary function symbol. Because of the new function symbols, we cannot expect $\phi$ and $\phi_{\text{Skolem}}$ to be equivalent. However, each model for $\phi_{\text{Skolem}}$ is also a model for $\phi$ (i.e., $\phi_{\text{Skolem}}\models\phi$) and each model $(\mathcal{A},\mathcal{V})$ for $\phi$ can be extended (by assigning appropriate meanings to the new function symbols) to a model for $\phi_{\text{Skolem}}$. Thus, $\phi$ is satisfiable iff so is $\phi_{\text{Skolem}}$.

**FOL without function symbols.** The use of function symbols in the syntax of FOL is useful since it often allows to provide short formulas for mathematical statements. However, the function symbols can be skipped without decreasing the expressiveness of FOL. Thus, FOL with *purely relational* vocabularies is as expressive as full FOL (with arbitrary vocabularies). Recall that a vocabulary is said to be purely relational if it does not contain any function symbol.

The rough idea for the transformation of a given FOL formula $\phi$ over some vocabulary $\mathit{Voc} = (\mathit{Pred},\mathit{Func})$ into a corresponding purely relational FOL formula (i.e., FOL formula over some purely relational vocabulary $\mathit{Voc}_{\text{rel}}$) is that any function $f$ can be identified with its graph

$$G_f \;=\; \big\{\,(\overline{a},b) : b = f(\overline{a})\,\big\}.$$

On the level of FOL, this means that for each function symbol $f$ of arity $k$, we may introduce a new predicate symbol $G_f$ of arity $k+1$ and replace any term $f(t_1,\ldots,t_k)$ with a fresh variable $y$ which is bounded by the formula $\exists y.\, G_f(t_1,\ldots,t_k,y)$. For example, let $P$ be a unary predicate symbol, $f$ a binary function symbol and $g$ a function symbol of arity 1. Then, the atomic formula

$$P\big(f(g(x_1),x_2)\big)$$

is replaced with

$$\exists y \exists z.\, \big(\underbrace{G_g(x_1,z)}_{g(x_1)=z} \wedge \underbrace{G_f(z,x_2,y)}_{f(z,x_2)=y} \wedge P(y)\big),$$

while the formula $f(c,x)=d$ (with constant symbols $c$ and $d$) is replaced with

$$\exists y \exists z \exists w.\, \big(\underbrace{G_c(z)}_{c=z} \wedge \underbrace{G_f(z,x,y)}_{f(z,x)=y} \wedge \underbrace{G_d(w)}_{d=w} \wedge (y=w)\big).$$

In this way, we obtain a transformation $\phi \mapsto \phi_{\text{rel}}$ which assigns to each FOL-formula $\phi$ over the vocabulary *Voc* a purely relational FOL-formula $\phi_{\text{rel}}$ over the vocabulary $\mathit{Voc}_{\text{rel}}$ which consists of the predicate symbols in *Voc* and the new predicate symbols $G_f$ for *Voc*'s function symbols. With this transformation of $\phi$ into a purely relational formula $\phi_{\text{rel}}$, we cannot guarantee that $\phi$ and $\phi_{\text{rel}}$ have "equivalent" models. Consider, for example, the FOL-formula

$$\begin{array}{lcll} \phi & = & \forall x \forall u.\big(P(x,u) \rightarrow u = f(x)\big) \wedge \theta & \text{where} \\ \theta & = & \exists x \exists u \exists z.\big(P(x,u) \wedge P(x,z) \wedge u \neq z\big) & \end{array}$$

where $P$ is a binary predicate symbol and $f$ an unary function symbol. For the switch from $\phi$ to $\phi_{\text{rel}}$ we replace the atomic subformula $u = f(x)$ with $\exists y.\big(G_f(x,y) \wedge u = y\big)$, which is equivalent to $G_f(x,u)$. Hence, we get:

$$\phi_{\text{rel}} \;\equiv\; \forall x \forall u.\big(P(x,u) \rightarrow G_f(x,u)\big) \wedge \theta$$

But then, the original FOL-formula is unsatisfiable, while, e.g., structure $\mathcal{A}$ with domain $A = \{0,1\}$ and $P^{\mathcal{A}} = G_f^{\mathcal{A}} = \big\{(0,0),(0,1)\big\}$ is a model for $\phi_{\text{rel}}$. To ensure the "equivalence" of $\phi$ and $\phi_{\text{rel}}$ we need further conditions that prescribe an interpretation of the auxiliary predicate symbols $G_f$ by graphs of functions. For this, we switch from $\phi_{\text{rel}}$ to

$$\phi'_{\text{rel}} \;\stackrel{\text{def}}{=}\; \phi_{\text{rel}} \wedge \bigwedge_{f \in \mathit{Func}(\phi)} \psi_f$$

where $\mathit{Func}(\phi)$ denotes the set of all function symbols that appear in $\phi$. (Note that there are only finitely many function symbols that appear in $\phi$.) If $f$ is a $k$-ary function symbol then formula $\psi_f$ is given by:

$$\psi_f \;\stackrel{\text{def}}{=}\; \forall \overline{x}\, \exists! y.\, G_f(\overline{x},y)$$

where $\overline{x} = (x_1,\ldots,x_k)$ and $\exists! y.\theta$ states the existence of a unique valuation for $y$ where $\theta$ holds. That is,

$$\exists! y.\theta \;\stackrel{\text{def}}{=}\; \exists y.\big(\theta \wedge \forall z.(\theta[y/z] \rightarrow (y=z))\big)$$

where $z$ is a fresh variable that does not appear in $\theta$. Thus, $y$ can be replaced with $z$ in $\theta$.

The "equivalence" of $\phi$ and $\phi'_{\text{rel}}$ can now be established in the following sense: Given a model $(\mathcal{A},\mathcal{V})$ for $\phi$ then $(\mathcal{A}',\mathcal{V})$ is a model for $\phi'_{\text{rel}}$ where $\mathcal{A}'$ has the same domain as $\mathcal{A}$ and assigns the same meaning to all predicate symbols $P$ of *Voc* and interpretes the predicate symbols $G_f$ by the graph of $f^{\mathcal{A}}$:

$$G_f^{\mathcal{A}'} \;=\; \big\{\,(a_1,\ldots,a_k,b) : f^{\mathcal{A}}(a_1,\ldots,a_k) = b\,\big\}. \qquad (*)$$

Vice versa, given a model for $\phi'_{\text{rel}}$ there is a corresponding model for $\phi$ with the same domain and interpretation of the predicate symbols in *Voc* and such that (*) holds. We conclude that given a vocabulary *Voc* then there is a purely relational vocabulary $\mathit{Voc}_{\text{rel}}$ and an algorithm that takes as input a FOL-formula $\phi$ over *Voc* and returns a FOL-formula $\phi'_{\text{rel}}$ over $\mathit{Voc}_{\text{rel}}$ of the same asymptotic length and the same asymptotic word-length such that $\phi$ is satisfiable over some structure with domain $A$ if and only if so is $\phi'_{\text{rel}}$.

**Many-sorted FOL.** In the standard FOL semantics a single domain serves as range for all variables. In several applications, however, the use of variables of different types (sorts) is useful. Many-sorted FOL is a variant of FOL where the variables and other symbols can have different domains. Vocabularies of many-sorted FOL extend standard vocabularies by a recursively enumerable set of *sorts*. i.e., they are triples $\mathit{Voc}_{ms} = (\mathit{Pred}, \mathit{Func}, \mathit{Sorts})$ where *Pred* and *Func* are as for standard FOL and *Sorts* is a nonempty set of sorts. Each variable $x$ is associated with a certain sort $\mathit{sort}(x) \in \mathit{Sorts}$. Any $m$-ary function symbol $f$ is associated with a $(m+1)$-tuple of sorts

$$\mathit{sort}(f) \;=\; (S_1,\ldots,S_m,S) \in \mathit{Sorts}^{m+1}$$

stating that $f$ stands for a function $S_1 \times \ldots \times S_m \to S$. Similarly, an $n$-tuple

$$\mathit{sort}(P) \;=\; (S_1,\ldots,S_n) \in \mathit{Sorts}^{n}$$

is assigned to each $n$-ary predicate symbol $P$, stating that $P$ represents a subset of $S_1 \times \ldots \times S_n$. The syntax of terms and formulas over many-sorted vocabularies is as for standard FOL, except that sort-consistency of terms and atomic formulas is required. Formally, the terms of some sort $S$ are defined as follows:

- Each variable $x$ with $\mathit{sort}(x) = S$ is a term of sort $S$.
- If $f$ is a $m$-ary function symbol with $\mathit{sort}(f) = (S_1,\ldots,S_m,S)$ and $t_i$ a term of sort $S_i$ for $i = 1,\ldots,m$, then $f(t_1,\ldots,t_m)$ is a term of sort $S$.

Nothing else is a term of sort $S$. The syntax of formulas is as for standard FOL, the only difference being that for each atom $P(t_1,\ldots,t_n)$ we require that if $\mathit{sort}(P) = (S_1,\ldots,S_n)$ then $t_i$ is a term of sort $S_i$ for $i = 1,\ldots,n$. A *structure* $\mathcal{A}$ of a many-sorted FOL-vocabulary assigns

- to each sort $S$ a nonempty set $S^{\mathcal{A}}$,
- to each $n$-ary predicate symbol $P$ where $\mathit{sort}(P) = (S_1,\ldots,S_n)$ a set $P^{\mathcal{A}} \subseteq S_1^{\mathcal{A}} \times \ldots \times S_n^{\mathcal{A}}$,
- to each $m$-ary function symbol $f$ where $\mathit{sort}(f) = (S_1,\ldots,S_m,S)$ a function

$$f^{\mathcal{A}} : S_1^{\mathcal{A}} \times \ldots \times S_m^{\mathcal{A}} \to S^{\mathcal{A}}.$$

A *variable valuation* for a many-sorted vocabulary $\mathit{Voc}_{ms}$ as above is a function $\mathcal{V}$ that associates with each variable $x$ an element $\mathcal{V}(x) \in S^{\mathcal{A}}$ where $S = \mathit{sort}(x)$. Given a structure $\mathcal{A}$ and variable valuation $\mathcal{V}$ for many-sorted FOL, the satisfaction relation is defined in the obvious way. For instance, if $x$ is a variable with $S = \mathit{sort}(x)$ then:

$$(\mathcal{A},\mathcal{V}) \models \exists x.\phi \quad \text{iff} \quad \text{there exists } a \in S^{\mathcal{A}} \text{ with } (\mathcal{A},\mathcal{V}[x := a]) \models \phi$$

$$(\mathcal{A},\mathcal{V}) \models \forall x.\phi \quad \text{iff} \quad (\mathcal{A},\mathcal{V}[x := a]) \models \phi \text{ for all } a \in S^{\mathcal{A}}$$

For example, the characteristic properties of a vector space can easily be formalized by many-sorted FOL-sentences. Here, we need two sorts: one sort $S_V$ for the vectors and one sort $S_F$ for the elements of the underlying field. Then, for instance, the associativity of the scalar multiplication in a vector space can be formalized by the FOL-sentence:

$$\forall\lambda\forall\mu\forall x.\,(\ \lambda \circ (\mu \circ x) = (\lambda * \mu) \circ x\ )$$

where $\lambda$ and $\mu$ are variables of sort $S_F$ (and hence they range over all elements of the field) and $x$ is a variable of sort $S_V$ (and hence it ranges over all vectors). Furthermore, "$\circ$" is used as function symbol for the scalar multiplication and "$*$" for multiplication in the underlying field.

For another example, some of the characteristic properties of *queues* can easily be specified in many-sorted FOL. For this, we use sorts *Queue* and some base domain *Elements* for the elements of the queues. Then, the emptiness check ("is the given queue empty?") can be modeled by an unary predicate symbol *empty* of type *Queue*. The operation *add* which stands for the insertion of a new element of *Elements* at the end of a queue can be modeled by a binary function symbol. Similarly, the operations *remove* and *front* can be specified by unary function symbols. The sorts of these function symbols are:

$$\begin{aligned} \mathit{sort}(\mathit{add}) &= (\mathit{Queue}, \mathit{Elements}, \mathit{Queue}) \\ \mathit{sort}(\mathit{remove}) &= (\mathit{Queue}, \mathit{Queue}) \\ \mathit{sort}(\mathit{front}) &= (\mathit{Queue}, \mathit{Elements}) \end{aligned}$$

Here are some examples for many-sorted FOL-formulas that describe the fifo-principle of the operations on queues:

$$\begin{aligned} \phi_1 &= \forall q\forall x.\,\big(\ \mathit{empty}(q) \rightarrow \mathit{empty}(\,\mathit{remove}(\mathit{add}(q,x))\,)\ \big) \\ \phi_2 &= \forall q\forall x.\,\big(\ \mathit{empty}(q) \rightarrow \mathit{front}(\mathit{add}(q,x)) = x\ \big) \\ \phi_3 &= \forall q\forall x.\,\big(\ \neg\mathit{empty}(q) \rightarrow \mathit{remove}(\mathit{add}(q,x)) = \mathit{add}(\mathit{remove}(q),x)\ \big) \end{aligned}$$

The first formula $\phi_1$ expresses that the insertion and immediate removal of an element $x$ to the empty queue yields the empty queue. The second formula specifies that the front element of the queue that arises by inserting $x$ into the empty queue is $x$. The last formula $\phi_3$ asserts the commutativity of insertion and deletion for nonempty queues.

Clearly, standard FOL can be viewed as a sublogic of many-sorted FOL which uses only a single sort. Vice versa, many-sorted FOL is as expressive as standard FOL. This follows from the fact that there is a syntactic translation $\phi \mapsto \phi'$ that transforms each many-sorted FOL-formula $\phi$ into a corresponding FOL-formula $\phi'$. For this, we simply treat the sorts as 1-ary predicate symbols and then replace any occurrence of $\forall x.\psi$ where $x$ is a variable of sort $S$ with $\forall x.(S(x) \rightarrow \psi)$ (and $\exists x.\phi$ with $\exists x.(S(x) \wedge \psi)$). Let $\widetilde{\phi}$ be the resulting formula. Since the meaning of a predicate symbol might be empty, while the interpretation of a sort in many-sorted FOL is a nonempty set, we have to add constraints stating that each sort has to be interpreted

by a nonempty set. Similarly, we need constraints stating the sort-consistency of the meanings of the predicate and function symbols. We therefore deal with

$$\phi' \stackrel{\text{def}}{=} \widetilde{\phi} \wedge \psi$$

where formula $\psi$ is the conjunction of the formulas $\exists x.S(x)$ for all sorts $S$ that appear in $\phi$ and formulas $\psi_P$ and $\psi_f$ for all predicate symbols $P$ and all function symbols $f$ that appear in $\phi$. The definition of the formulas $\psi_P$ and $\psi_f$ is as follows. If $P$ is a $n$-ary predicate symbol of sort $(S_1, \ldots, S_n)$ then:

$$\psi_P \stackrel{\text{def}}{=} \forall x_1 \ldots \forall x_n. \Big( P(x_1, \ldots, x_n) \rightarrow \bigwedge_{1 \leqslant i \leqslant n} S_i(x_i) \Big)$$

Similarly, if $f$ is a $m$-ary function symbol of sort $(S_1, \ldots, S_m, S)$ then:

$$\psi_f \stackrel{\text{def}}{=} \forall x_1 \ldots \forall x_m. \Big( \bigwedge_{1 \leqslant i \leqslant m} S_i(x_i) \rightarrow S(f(x_1, \ldots, x_m)) \Big)$$

The equivalence of the original many-sorted FOL-formula $\phi$ over some many-sorted vocabulary $\mathit{Voc}_{ms} = (\mathit{Pred}, \mathit{Func}, \mathit{Sorts})$ and the generated FOL-formula $\phi'$ over the induced (standard single-sorted) vocabulary $\mathit{Voc}$ holds in the following sense:

- If $(\mathcal{A}, \mathcal{V})$ is a model for the many-sorted formula $\phi$ then $(\mathcal{B}, \mathcal{V})$ is a model for $\phi'$ over $\mathit{Voc}$ where the domain $B$ of $\mathcal{B}$ is the union of the domains of all sorts. That is:

  $$B = \bigcup_{S \in \mathit{Sorts}} S^{\mathcal{A}}$$

  For each sort $S$, the set $S^{\mathcal{B}}$ associated with $S$ viewed as an unary predicate symbol of $\mathit{Voc}'$ is the domain of $S$ in the many-sorted structure $\mathcal{A}$, i.e., we define $S^{\mathcal{B}} \stackrel{\text{def}}{=} S^{\mathcal{A}}$. The interpretations of the predicate symbols are as in $\mathcal{A}$, i.e., $P^{\mathcal{B}} = P^{\mathcal{A}}$ for each predicate symbol $P$. For the $m$-ary function symbols $f$, let $f^{\mathcal{B}} : B^m \rightarrow B$ be an arbitrary extension of $f^{\mathcal{A}}$. Note that $f^{\mathcal{A}}$ is a function of the form $f^{\mathcal{A}} : A_1 \times \ldots \times A_m \rightarrow A'$ where the $A_i$'s and $A'$ are subsets of $B$. More precisely, if $f$ is of sort $(S_1, \ldots, S_m, S)$ then $A_i = S_i^{\mathcal{A}}$ for $1 \leqslant i \leqslant m$ and $A' = S^{\mathcal{A}}$. Hence, we may define $f^{\mathcal{B}}$ as a function that agrees with $f^{\mathcal{A}}$ for the tuples in $A_1 \times \ldots \times A_m$ and assigns an arbitrary value $b \in B$ to the remaining elements of $B^m$.

- If $(\mathcal{B}, \mathcal{V})$ is a model for $\phi'$ over $\mathit{Voc}$ such that for all variables $x$ we have $\mathcal{V}(x) \in S^{\mathcal{B}}$ where $S = \mathit{sort}(x)$ then a model $(\mathcal{A}, \mathcal{V})$ for the original many-sorted FOL-formula $\phi$ over $\mathit{Voc}_{ms}$ is obtained as follows. If $S$ is a sort in the many-sorted FOL-vocabulary $\mathit{Voc}_{ms}$ then $S^{\mathcal{A}}$ agrees with the meaning of $S$ under $\mathcal{B}$ when $S$ is viewed as an unary predicate symbol, i.e., $S^{\mathcal{A}} = S^{\mathcal{B}}$. Then:
  - if $P$ is a predicate symbol of sort $(S_1, \ldots, S_n)$ then $P^{\mathcal{A}} = P^{\mathcal{B}} \cap (S_1^{\mathcal{B}} \times \ldots \times S_n^{\mathcal{B}})$
  - if $f$ is a function symbol of sort $(S_1, \ldots, S_m, S)$ then $f^{\mathcal{A}} = f^{\mathcal{B}}|_{S_1^{\mathcal{B}} \times \ldots \times S_m^{\mathcal{B}}}$

Thus, for any many-sorted vocabulary $\mathit{Voc}_{ms}$ there exists a vocabulary $\mathit{Voc}$ and an algorithm that constructs for a given many-sorted FOL-sentence $\phi$ over $\mathit{Voc}_{ms}$ a FOL-sentence $\phi'$ over $\mathit{Voc}$ of the same asymptotic length and the same asymptotic word-length such that the models

for $\phi$ and $\phi'$ agree in the above sense. Note that this does not hold for many-sorted formulas with free variables. Regard, for instance, the unsatisfiable many-sorted FOL-formula

$$\phi \;=\; \neg P(x) \wedge \forall y.P(y)$$

where $P$ is an unary predicate symbol and $x$ and $y$ are variables with $\mathit{sort}(P) = \mathit{sort}(x) = \mathit{sort}(y) = S$. Then, the induced standard FOL-formula is

$$\phi' \;=\; \neg P(x) \wedge \forall y.(S(y) \to P(y)) \wedge \forall z.(P(z) \to S(z)) \wedge \exists z.S(z).$$

While the original many-sorted formula $\phi$ is unsatisfiable, each interpretation $\mathcal{I} = (\mathcal{A}, \mathcal{V})$ with domain $A$ where $P^{\mathcal{A}} = S^{\mathcal{A}}$ is a proper subset of $A$ and where $\mathcal{V}(x) \in A \setminus P^{\mathcal{A}}$ is a model for $\phi$. The problem in this example is that $\phi'$ does not contain any constraint for the sort-consistent interpretation of the free variable $x$ by the variable valuation $\mathcal{V}$. However, if $\phi$ is a many-sorted FOL-sentence then $\phi$ is satisfiable if and only if so is $\phi'$. Moreover, if $\phi'$ has a finite model (i.e., evaluates to true over some structure for $\mathit{Voc}_{ms}$ where the domains of all sorts are finite) then $\phi$ has a finite model, and vice versa. This justifies the statement that the use of typed symbols does not increase the expressiveness of FOL.

## 1.2 Deductive calculi for FOL

Validity of formulas and the consequence relation $\models$ are purely semantic features. The goal of *deductive systems* (also called deductive calculi or *proof systems*) is to provide a mechanic way to reason about validity or consequences based on the *syntax* of formulas. Speaking roughly, a deductive system consists of finitely many *axioms* and *proof rules*, often simply called rules. Most popular are *Hilbert proof systems* where the axioms are decidable sets of formulas, while the rules are decidable sets of $(n+1)$-ary formula-tuples $(\phi_1, \ldots, \phi_n, \phi)$. The intuitive meaning of $(\phi_1, \ldots, \phi_n, \phi)$ is that, "assuming $\phi_1, \ldots, \phi_n$ hold, then also $\phi$ holds", or formulated in proof-terminology "if $\phi_1, \ldots, \phi_n$ are provable then so is $\phi$". Thus, formulas $\phi_1, \ldots, \phi_n$ constitute the *premise*, while $\phi$ stands for the *consequence*. Axioms arise as special instances of rules for the case $n{=}0$. Thus, the formulas in an axiom are supposed to hold without any further assumptions. Typically, the axioms and rules are given by *schemata* of the form

$$\frac{\Xi_1, \ldots, \Xi_n}{\Xi} \qquad \begin{array}{l} \longleftarrow \text{ premise} \\ \longleftarrow \text{ consequence} \end{array}$$

where the $\Xi_i$'s and $\Xi$ are FOL-formulas that might use *formula symbols* as atoms. We use capital greek letters $\Phi, \Psi, \Theta, \ldots$ to denote the formula symbols. Sometimes, in addition to $\Xi_1, \ldots, \Xi_n$ the premise also contains a boolean condition on the syntax of $\Xi_1, \ldots, \Xi_n, \Xi$. The only requirement for such additional constraints for the premise is that they have to be decidable. Axiom schemata are special instances of rule schemata where $n{=}0$.

The formula symbols $\Phi$, $\Psi$, $\Theta$, ... in rule schemata range over all formulas over some fixed vocabulary and variable set. I.e., the above rule schema stands for the set of all tuples $(\phi_1, \ldots, \phi_n, \phi)$ that are obtained by replacing the formula symbols with concrete formulas. More precisely, an *instance* of a rule schema as above is a tuple $(\phi_1, \ldots, \phi_n, \phi)$ such that there exists a substitution $\sigma$ which replaces the formula symbols appearing in $\Xi_1, \ldots, \Xi_n$ and $\Xi$ with (concrete) formulas and which yields $\phi_i = \Xi_i\sigma$, $1 \leqslant i \leqslant n$, and $\phi = \Xi\sigma$. The set associated with a rule schema is

simply the set of its instances. For example, the *modus ponens* is given by the following rule schema:

$$\frac{\Phi,\ \Phi \to \Psi}{\Psi} \qquad \text{(modus ponens)}$$

Here, $\Phi$ and $\Psi$ are formula symbols and the above rule stands for the set of all triples $(\phi, \phi \to \psi, \psi)$ where $\phi$ and $\psi$ are formulas. (Again, we suppose here a fixed vocabulary *Voc* and variable-set *Var* such that $\phi$ and $\psi$ range over all FOL-formulas over (*Voc*, *Var*).) The meaning of modus ponens is that if $\phi$ and $\phi \to \psi$ hold then also $\psi$ holds.

**Definition 1.2.1 (Hilbert proof systems, derivations,proofs).** A Hilbert proof system $\mathfrak{D}$ consists of a finite set of proof rules, or briefly rules, where each proof rule is a decidable set of $(n+1)$-tuples of formulas for some $n \in \mathbb{N}$. Value $n$ is said to be the arity of the rule. Rules of arity 0 are called axioms. If $\mathfrak{F}$ is a formula-set then a *derivation* in $\mathfrak{D}$ (briefly $\mathfrak{D}$-derivation or $\mathfrak{D}$-proof) from $\mathfrak{F}$ denotes a finite sequence $\psi_1, \ldots, \psi_m$ of formulas such that for each $i \in \{1, \ldots, m\}$ at least one of the following conditions holds:

- $\psi_i$ is an element of $\mathfrak{F}$
- $\psi_i$ is an instance of an axiom
- there exists an instance $(\phi_1, \ldots, \phi_n, \phi)$ of some $n$-ary proof rule in $\mathfrak{D}$ such that $n \geqslant 1$ and $\psi_i = \phi$ and $\{\phi_1, \ldots, \phi_n\} \subseteq \{\psi_1, \ldots, \psi_{i-1}\}$.

The third option means that the $i$-th formula $\psi_i$ follows from one or more earlier items in the sequence by applying a proof rule. Value $m$ is called the *length* of the $\mathfrak{D}$-proof $\psi_1, \ldots, \psi_m$. ■

The *derivation relation* $\vdash_{\mathfrak{D}}$ is defined as follows:

$$\mathfrak{F} \vdash_{\mathfrak{D}} \phi \quad \text{iff} \quad \begin{cases} \text{there exists a } \mathfrak{D}\text{-proof } \psi_1, \ldots, \psi_m \\ \text{from } \mathfrak{F} \text{ such that } \psi_m = \phi \end{cases}$$

Formula $\phi$ is said to be $\mathfrak{D}$-provable from $\mathfrak{F}$ (or $\mathfrak{D}$-derivable from $\mathfrak{F}$) if $\mathfrak{F} \vdash_{\mathfrak{D}} \phi$. Derivations from $\mathfrak{F} = \varnothing$ are sequences of formulas obtained by the axioms and applying the proof rules. We simply write $\vdash_{\mathfrak{D}} \phi$ rather than $\varnothing \vdash_{\mathfrak{D}} \phi$, in which case $\phi$ is said to be $\mathfrak{D}$-provable. If $\phi$ and $\psi$ are formulas then the notation $\phi \vdash_{\mathfrak{D}} \psi$ is used to indicate that $\psi$ is $\mathfrak{D}$-provable from the singleton formula-set $\{\phi\}$.