

## byd.1 Many-Sorted Logic

fol:byd:msl:  
sec

In first-order logic, variables and quantifiers range over a single **domain**. But it is often useful to have multiple (disjoint) **domains**: for example, you might want to have a **domain** of numbers, a **domain** of geometric objects, a **domain** of functions from numbers to numbers, a **domain** of abelian groups, and so on.

Many-sorted logic provides this kind of framework. One starts with a list of “sorts”—the “sort” of an object indicates the “**domain**” it is supposed to inhabit. One then has **variables** and quantifiers for each sort, and (usually) an **identity predicate** for each sort. Functions and relations are also “typed” by the sorts of objects they can take as arguments. Otherwise, one keeps the usual rules of first-order logic, with versions of the quantifier-rules repeated for each sort.

For example, to study international relations we might choose a language with two sorts of objects, French citizens and German citizens. We might have a unary relation, “drinks wine,” for objects of the first sort; another unary relation, “eats wurst,” for objects of the second sort; and a binary relation, “forms a multinational married couple,” which takes two arguments, where the first argument is of the first sort and the second argument is of the second sort. If we use variables  $a, b, c$  to range over French citizens and  $x, y, z$  to range over German citizens, then

$$\forall a \forall x [(MarriedTo(a, x) \rightarrow (DrinksWine(a) \vee \neg EatsWurst(x)))]$$

asserts that if any French person is married to a German, either the French person drinks wine or the German doesn’t eat wurst.

Many-sorted logic can be embedded in first-order logic in a natural way, by lumping all the objects of the many-sorted **domains** together into one first-order **domain**, using unary **predicate symbols** to keep track of the sorts, and relativizing quantifiers. For example, the first-order language corresponding to the example above would have unary **predicate symbols** “*German*” and “*French*,” in addition to the other relations described, with the sort requirements erased. A sorted quantifier  $\forall x \varphi$ , where  $x$  is a **variable** of the German sort, translates to

$$\forall x (German(x) \rightarrow \varphi).$$

We need to add axioms that insure that the sorts are separate—e.g.,  $\forall x \neg(German(x) \wedge French(x))$ —as well as axioms that guarantee that “drinks wine” only holds of objects satisfying the predicate  $French(x)$ , etc. With these conventions and axioms, it is not difficult to show that many-sorted **sentences** translate to first-order **sentences**, and many-sorted **derivations** translate to first-order **derivations**. Also, many-sorted **structures** “translate” to corresponding first-order **structures** and vice-versa, so we also have a completeness theorem for many-sorted logic.

**Photo Credits**

**Bibliography**