sally quantified sentences can be asserted using inverse links and the singly boxed and doubly boxed arrows applied to categories, but that still leaves us a long way short of full first-order logic. Negation, disjunction, nested function symbols, and existential quantification are all missing. Now it is possible to extend the notation to make it equivalent to first-order logic—as in Peirce’s existential graphs—but doing so negates one of the main advantages of semantic networks, which is the simplicity and transparency of the inference processes. Designers can build a large network and still have a good idea about what queries will be efficient, because (a) it is easy to visualize the steps that the inference procedure will go through and (b) in some cases the query language is so simple that difficult queries cannot be posed. In cases where the expressive power proves to be too limiting, many semantic network systems provide for procedural attachment to fill in the gaps. Procedural attachment is a technique whereby a query about (or sometimes an assertion of) a certain relation results in a call to a special procedure designed for that relation rather than a general inference algorithm.

One of the most important aspects of semantic networks is their ability to represent default values for categories. Examining Figure 12.5 carefully, one notices that John has one leg, despite the fact that he is a person and all persons have two legs. In a strictly logical KB, this would be a contradiction, but in a semantic network, the assertion that all persons have two legs has only default status; that is, a person is assumed to have two legs unless this is contradicted by more specific information. The default semantics is enforced naturally by the inheritance algorithm, because it follows links upwards from the object itself (John in this case) and stops as soon as it finds a value. We say that the default is overridden by the more specific value. Notice that we could also override the default number of legs by creating a category of OneLeggedPersons, a subset of Persons of which John is a member.

We can retain a strictly logical semantics for the network if we say that the Legs assertion for Persons includes an exception for John:

\[ x : x \in \text{Persons} \land x \neq \text{John} \rightarrow \text{Legs}(x, 2) \]

For a fixed network, this is semantically adequate but will be much less concise than the network notation itself if there are lots of exceptions. For a network that will be updated with more assertions, however, such an approach fails—we really want to say that any persons as yet unknown with one leg are exceptions too. Section 12.6 goes into more depth on this issue and on default reasoning in general.

### 12.5.2 Description logics

The syntax of first-order logic is designed to make it easy to say things about objects. Description logics are notations that are designed to make it easier to describe definitions and properties of categories. Description logic systems evolved from semantic networks in response to pressure to formalize what the networks mean while retaining the emphasis on taxonomic structure as an organizing principle.

The principal inference tasks for description logics are subsumption (checking if one category is a subset of another by comparing their definitions) and classification (checking whether an object belongs to a category). Some systems also include consistency of a category definition—whether the membership criteria are logically satisfiable.
The CLASSIC language (Borgida et al., 1989) is a typical description logic. The syntax of CLASSIC descriptions is shown in Figure 12.7. For example, to say that bachelors are unmarried adult males we would write

\[
\text{Bachelor} = \text{And}(\text{Unmarried}, \text{Adult}, \text{Male})
\]

The equivalent in first-order logic would be

\[
\text{Bachelor}(x) \land \text{Unmarried}(x) \land \text{Adult}(x) \land \text{Male}(x)
\]

Notice that the description logic has an algebra of operations on predicates, which of course we can’t do in first-order logic. Any description in CLASSIC can be translated into an equivalent first-order sentence, but some descriptions are more straightforward in CLASSIC. For example, to describe the set of men with at least three sons who are all unemployed and married to doctors, and at most two daughters who are all professors in physics or math departments, we would use

\[
\text{And}(\text{Man}, \text{AtLeast}(3, \text{Son}), \text{AtMost}(2, \text{Daughter})
\text{All}(\text{Son}, \text{And}(\text{Unemployed}, \text{Married}, \text{All}(\text{Spouse}, \text{Doctor}))),
\text{All}(\text{Daughter}, \text{And}(\text{Professor}, \text{Fills}(\text{Department}, \text{Physics}, \text{Math}))))
\]

We leave it as an exercise to translate this into first-order logic.

Perhaps the most important aspect of description logics is their emphasis on tractability of inference. A problem instance is solved by describing it and then asking if it is subsumed by one of several possible solution categories. In standard first-order logic systems, predicting the solution time is often impossible. It is frequently left to the user to engineer the representation to detour around sets of sentences that seem to be causing the system to take several weeks to solve a problem. The thrust in description logics, on the other hand, is to ensure that subsumption-testing can be solved in time polynomial in the size of the descriptions.

Notice that the language does not allow one to simply state that one concept, or category, is a subset of another. This is a deliberate policy: subsumption between categories must be derivable from some aspects of the descriptions of the categories. If not, then something is missing from the descriptions.

CLASSIC provides efficient subsumption testing in practice, but the worst-case run time is exponential.