Semantic networks

EXISTENTIAL GRAPHS

In 1909, Charles Peirce proposed a graphical notation of nodes and arcs called **existential graphs** that he called "the logic of the future." Thus began a long-running debate between advocates of "logic" and advocates of "semantic networks." Unfortunately, the debate obscured the fact that semantics networks—at least those with well-defined semantics—are a form of logic. The notation that semantic networks provide for certain kinds of sentences is often more convenient, but if we strip away the "human interface" issues, the underlying concepts—objects, relations, quantification, and so on—are the same.

There are many variants of semantic networks, but all are capable of representing individual objects, categories of objects, and relations among objects. A typical graphical notation displays object or category names in ovals or boxes, and connects them with labeled arcs. For example, Figure 10.9 has a $MemberOf$ link between *Mary* and *FemalePersons,* corresponding to the logical assertion $Mary \in FemalePersons$; similarly, the $SisterOf$ link between *Mary* and *John* corresponds to the assertion $SisterOf(Mary, John)$. We can connect categories using $SubsetOf$ links, and so on. It is such fun drawing bubbles and arrows that one can get carried away. For example, we know that persons have female persons as mothers, so can we draw a $HasMother$ link from *Persons* to *FemalePersons?* The answer is no, because $HasMother$ is a relation between a person and his or her mother, and categories do not have mothers.[8] For this reason, we have used a special notation—the double-boxed link—in Figure 10.9. This link asserts that

$$\forall x \;\; x \in Persons \;\Rightarrow\; [\forall y \;\; HasMother(x, y) \;\Rightarrow\; y \in FemalePersons].$$

We might also want to assert that persons have two legs—that is,

$$\forall x \;\; x \in Persons \;\Rightarrow\; Legs(x, 2).$$

As before, we need to be careful not to assert that a category has legs; the single-boxed link in Figure 10.9 is used to assert properties of every member of a category.

The semantic network notation makes it very convenient to perform **inheritance** reasoning of the kind introduced in Section 10.2. For example, by virtue of being a person, Mary inherits the property of having two legs. Thus, to find out how many legs Mary has, the inheritance algorithm follows the $MemberOf$ link from *Mary* to the category she belongs to, and then follows $SubsetOf$ links up the hierarchy until it finds a category for which there is a boxed *Legs* link—in this case, the *Persons* category. The simplicity and efficiency of this inference mechanism, compared with logical theorem proving, has been one of the main attractions of semantic networks.

MULTIPLE INHERITANCE

Inheritance becomes complicated when an object can belong to more than one category or when a category can be a subset of more than one other category; this is called **multiple inheritance.** In such cases, the inheritance algorithm might find two or more conflicting values

[8] Several early systems failed to distinguish between properties of members of a category and properties of the category as a whole. This can lead directly to inconsistencies, as pointed out by Drew McDermott (1976) in his article "Artificial Intelligence Meets Natural Stupidity." Another common problem was the use of IsA links for both subset and membership relations, in correspondence with English usage: "a cat is a mammal" and "Fifi is a cat." See Exercise 10.25 for more on these issues.

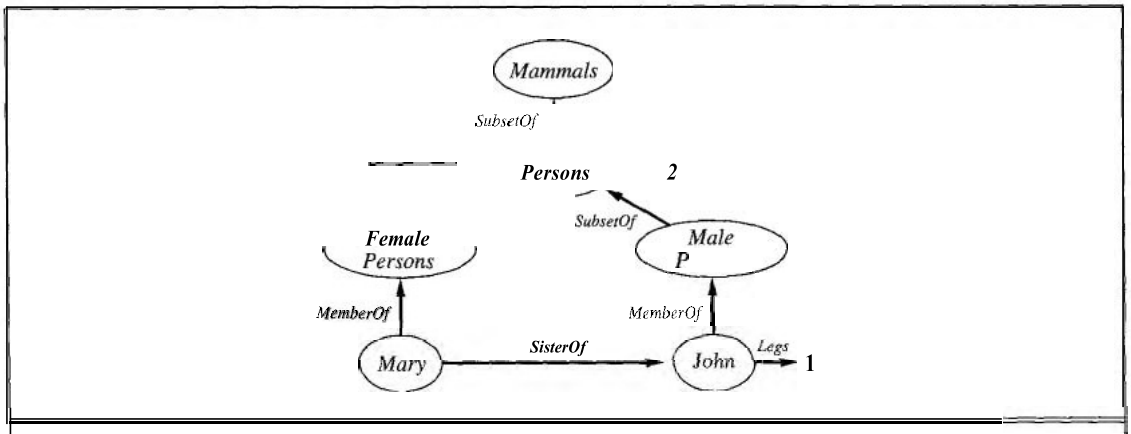

Figure 10.9 A semantic network with four objects (John, Mary, 1, and 2) and four categories. Relations are denoted by labeled links.

answering the query. For this reason, multiple inheritance is banned in some object-oriented programming (OOP) languages, such as Java, that use inheritance in a class hierarchy. It is usually allowed in semantic networks, but we defer discussion of that until Section 10.7.

INVERSE LINK

Another common form of inference is the use of inverse links. For example, $HasSister$ is the inverse of $SisterOf$, which means that

$$\forall p, s \;\; HasSister(p, s) \Leftrightarrow SisterOf(s, p) .$$

This sentence can be asserted in a semantic network if links are reified—that is, made into objects in their own right. For example, we could have a $SisterOf$ object, connected by an $Inverse$ link to $HasSister$. Given a query asking who is a $SisterOf$ John, the inference algorithm can discover that $HasSister$ is the inverse of $SisterOf$ and can therefore answer the query by following the $HasSister$ link from $John$ to $Mary$. Without the inverse information, it might be necessary to check every female person to see whether that person has a $SisterOf$ link to John. This is because semantic networks provide direct indexing only for objects, categories, and the links emanating from them; in the vocabulary of first-order logic, it is as if the knowledge base were indexed only on the first argument of each predicate.

The reader might have noticed an obvious drawback of semantic network notation, compared to first-order logic: the fact that links between bubbles represent only *binary* relations. For example, the sentence $Fly(Shankar, NewYork, NewDelhi, Yesterday)$ cannot be asserted directly in a semantic network. Nonetheless, we can obtain the effect of n-ary assertions by reifying the proposition itself as an event (see Section 10.3) belonging to an appropriate event category. Figure 10.10 shows the semantic network structure for this particular event. Notice that the restriction to binary relations forces the creation of a rich ontology of reified concepts; indeed, much of the ontology developed in this chapter originated in semantic network systems.

Reification of propositions makes it possible to represent every ground, function-free atomic sentence of first-order logic in the semantic network notation. Certain kinds of universally 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

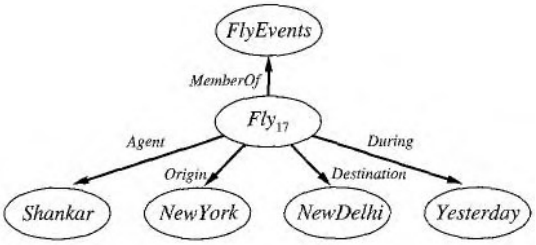

Figure 10.10 A fragment of a semantic network showing the representation of the logical assertion $Fly(Shankar, NewYork, NewDelhi, \texttt{Yesterday})$.

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 or Hendrix's (1975) partitioned semantic networks—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.

DEFAULT VALUES

One of the most important aspects of semantic networks is their ability to represent **default values** for categories. Examining Figure 10.9 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.

OVERRIDING

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

$$\forall x \;\; x \in Persons \wedge x \neq John \Rightarrow Legs(x, 2) .$$

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 10.7 goes into more depth on this issue and on default reasoning in general.