AN OVERVIEW OF A PROCEDURAL APPROACH TO SEMANTIC NETWORKS

Hector J. Levesque
John Mylopoulos
Department of Computer Science
University of Toronto

The purpose of this short paper is to summarize the salient features of a formalism for the construction and use of a model representing knowledge of some domain. For a much more detailed exposition, including examples and motivation, see [Levesque 77 J where, among other things, portions of the formalism are represented using the formalism itself.

Although the underlying ideas used in the development of the formalism are adaptations from diverse sources, our starting point is the work on semantic networks. A major problem in many semantic network proposals is the lack of a semantic theory [Hayes 74] in terms of which one can explain what a semantic network means. We are attempting to deal with this problem by using a "procedural semantics" approach based on LAbrial 74] in contrast to the more declarative approach of classical Predicate Logic.

The most primitive construct provided by the formalism is the object representing any single relevant conceptual unit. A fundamental kind of object is the class which is a collection of objects sharing common properties. These objects arc said to be instances of the class and may themselves be classes . A very important kind of class is the binary relation representing a mapping from one class to another. Instances of a relation represent assertions made regarding two objects, one from each class.

There are four basic operations defined on classes: create or destroy an instance of a class, test whether an object is an instance of a class and fetch all instances of a class. Similarly, four"operations are defined for each relation: assert or deny that a relation holds between two objects, test whether a relation holds and fetch all objects that are mapped by a relation from a given object. These basic operations can be combined and parametrized to form programs which are classes whose instances are called processes. Programs can be decomposed into a prerequisite, a main body, an effect and a complaint. A~program can Tail if its prerequisite is not satisfied. Otherwise, the main body is executed and if this execution is successful, the effect part is executed next, else the complaint part of the program is executed.

To specify the semantics of a given class (or relation), programs can be attached to any or all of the operations for the class and will be used instead of default programs supplied by the formalism. In addition, to refine or extend the behaviour of a class under its defined operations, only a part of a program (e.g. a prerequisite) need be given and the rest can default to the standard operation.

The objects constituting a model can be organized in terms of two orthogonal abstraction facilities called respectively the ISA and PARTOF hierarchies.

Apart from the conceptual" gains offered by having a taxonomy of classes, the main purpose of the ISA hierarchy is the inheritance of definitional properties which is a generalization of the default mechanism that capitalizes on the expected similarity between a subclass and a superclass. The PARTOF hierarchy, on the other hand, involves the composition of groups of objects into functional units. Such units are called structures and the objects that constitute them, their parts. For classes, the parts are variables (slots) which are bound to objects (slot fi llers) whenever the class is instantiated The slots may have default values and relations called dependencies attaching them to other objects and having many possible procedural interpretations such as determining the acceptability of a slot filler or inferring its value. Whereas relations represent the assertional properties of a class, slots represent the definitional properties. Consequently, the PARTOF hierarchy determines the inheritable properties of a class while the ISA hierarchy specifies the inheritance paths.

The metaclass "program" (the class of all programs) can be thought of as a structure having prerequisite,effect etc. as slots and each program fills these slots by providing particular prerequisites etc. This allows programs to be integrated into the two hierarchies and benefit from the organization (e.g. inheritance) like any other class. Similarly, the metaclass "class" is a structure whose slots represent the four basic programs for classes and the semantics of each class are defined by filling these slots by specific programs. The default values for these slots (and the:four similar ones in "relation") constitute the standard behaviour of classes and relations providing the primitives in terms of which everything else can be defined and linking the formalism to a physical implementation.

References

[Levesque 77] Levesque, H. "A procedural approach to semantic networks", TR-105, Dept. of Computer Science, U. of Toronto, 1977. [Hayes 74] Hayes, P.J. "Some problems and non-problems in representation theory", Proc. AISB summer conference,1974. [Abrial 74] Abrial, J.R. "Data semantics" in Data Management Systems, Klimbie and Koffeman (eds.), North-Holland, 1974.