A Conceptual Schema for Knowledge-Based Systems

John F. Sowa
IBM Systems Research Institute
205 East 42nd Street
New York, NY 10017

ABSTRACT: Knowledge-based systems are data bases with more powerful front ends for dealing with the meaning of data. This paper discusses requirements for a conceptual schema that is general enough to support knowledge bases as well as ordinary data bases. It presents seven features that such a schema must support and evaluates various approaches to data base semantics in terms of them. The AI notations for semantic networks or conceptual graphs are highly general ones that can support all seven features.

Representing Knowledge

Both data base (DB) and artificial intelligence (AI) systems must represent and process knowledge about the real world. The primary difference between the two fields lies in the volume of data that they process and the complexity of the representations. Data base systems started as file management systems with large volumes of data organized in a small number of record types. Although they have evolved towards more complex structures, data bases still tend to have a large amount of repetition of very few types. AI, by contrast, started with small problems of highly complex structure: understanding natural language, analyzing a visual scene, or guiding a robot. In AI, the total amount of data is usually small enough to reside in main storage, but almost every item is unique, with its own characteristic relationships.

The difference between DB and AI systems can be measured by the ratio of data descriptors to individual data items. In a DB system, thousands of employee records may all have an identical format: one set of descriptors is sufficient to describe every record. In typical AI systems, however, the ratio is almost one-to-one: since there is so little repetition, each item must have its own descriptor; in fact, most AI systems don't even distinguish data items from data descriptors. Because of the difference in ratios, the two fields have major differences in emphasis and priorities. DB systems emphasize efficient storage and retrieval, but have tended to leave descriptors as an afterthought—only recently have they been moved out of the programmer's notebook and into data dictionaries. AI systems, on the other hand, have developed sophisticated techniques for representing a great deal of knowledge about a small number of items; yet they have been impractical for commercial applications because they have high overhead, reside totally in main storage, and ignore backup and error recovery.

Since the difference between DB and AI systems is not a qualitative difference in their goals for representing knowledge but a quantitative difference in the ratio of descriptors to data items, one can expect intermediate cases to appear. As AI systems begin to deal with larger problems, they will find more repetition of items with the same descriptors. As DB systems move towards less highly structured applications, they will have to put greater emphasis on describing fewer items with more complex relationships. Both fields have a great deal to contribute to each other: DB has more practical experience in security, efficiency, and reliability; AI has developed more sophisticated techniques for representing the meaning of data.

What to Represent

Knowledge-based systems have been implemented, at least partially, in various prototype systems in AI. Brachman and Smith (1980) compiled a roster of 83 current projects on knowledge representation with a total of 277 active research workers. Typical knowledge-based systems include the projects at Stanford for medical diagnoses and organic chemistry (Feigenbaum 1977), Heidorn's NLPQ system (1972) for carrying on a dialog about queuing systems and automatically generating a program to simulate them, and various dissertations at Yale based on *scripts* (Schank & Abelson 1977). Three knowledge-based systems that apply AI techniques to data base queries

include the LADDER system (Hendrix et al. 1978), which answers English questions about naval logistics data, PLANES (Waltz 1978), which can answer questions like "Which A7's that had engine damage in Jan. 1973 flew in Feb. 1973?" and RENDEZVOUS (Codd 1978), which carries on a dialog with the user that combines natural language questions with prompting by menus. The distinguishing characteristic of all these systems is that they incorporate a great deal of knowledge about some aspect of the real world and make use of it in a dialog in much the same way that people do.

With such a diversity of projects and approaches, there is a corresponding variety of notations and formalisms. Each approach has a different emphasis and collection of features, but all of them must represent aspects of seven basic kinds of knowledge:

1. Generalization hierarchy: Types of entities or concepts of entities must be ordered according to levels of generality, such as COLLIE, DOG, ANIMAL, LIVING-THING, PHYSICAL-OBJECT, ENTITY.
2. Functional dependencies: The notation must show which entity types are keys or independent variables and which ones are functionally dependent on the keys. It should also specify quantifiers that show whether a function is many-to-one, one-to-one, or *n*-to-*m*.
3. Domain roles: Besides showing that two entity types are functionally dependent, the notation must describe the role that the dependency represents: *e.g.* instead of merely saying that there are two functions f_1 and f_2 such that for each person x, $f_1(x)$ is male and $f_2(x)$ is female, the representation should indicate that $f_1(x)$ is the father of x and $f_2(x)$ is the mother of x. Ideally, the roles should be specified in enough detail to be automatically translatable into an English description.
4. Definitional mechanisms: Concepts of composite entities, types, and relations must be definable in terms of structures of other concepts. As in Aristotle's method of definition by *genus* and *differentia,* the definition of EMPLOYEE would specify the genus or more general type PERSON and the characteristic differentia "one who WORKS for a COMPANY."
5. Conventional schemata: For each type of concept or entity, the notation must describe the conventional, normally occurring, or default roles that it plays with respect to other concepts. Whereas a type definition for EMPLOYEE presents the primary defining characteristic, the schema would include the background information that an employee has an employee number, earns a salary, reports to a manager, works in a department, etc.
6. Procedural attachment: The notation should indicate how an external procedure may be related to a functional dependency, and under what conditions it would be invoked to compute the function. Computed functions should be described in a way that is parallel to the stored functions represented in the data base. In AI, schemata have also been called *frames, scripts,* and *scenarios.*
7. Inference mechanisms: The notation should be supplemented with rules of inference that can determine implications that follow from the explicitly stored data and can detect violations of constraints upon the data.

Comparison of Representations

The bewildering variety of knowledge representations can be related to one another by determining which features each one is trying to represent. Bachman diagrams (1969), for example, appear to be completely unrelated to the semantic networks in AI (Findler 1979). In fact, they represent two equally fundamental, but different kinds of information: Bachman diagrams represent functional dependencies, and semantic networks emphasize the domain roles (or case relations) that are especially important for natural language. Working independently, Roussopoulos (1976) and Sowa (1976) both overlaid functional dependency arcs onto semantic networks and used the functional dependencies to show scope of quantifiers.

Various proposals for data base semantics can be characterized by the selection of features that they choose to represent. The characterizations may be slightly blurred, however, by the fact that notations that emphasize certain features may also represent other features in a rudimentary form.

- Chen's entity-relationship model (1976) is a refinement of Bachman diagrams that emphasizes functional dependencies.
- The functional dependency model by Housel, Waddle, and Yao (1979) treats functional dependencies and the generalization hierarchy.
- Smith and Smith (1977) represented the generalization hierarchy and the conventional schemata, which they called *aggregation.* They also represented functional dependencies and domain roles to some extent, but did not develop them as fully. At the Workshop on Data Abstractions, however, they did present further work on representing domain roles.

- Hemphill and Rhyne (1978) applied Schank and Abelson's scripts (1977) to data base semantics with primary emphasis on domain roles and conventional schemata, but they did not treat functional dependencies and definitional mechanisms.
- McSkimin and Minker (1979) developed a sorted first-order predicate calculus that treats the inference mechanisms, generalization hierarchy, and functional dependencies, but ignores domain roles and conventional schemata.
- One of the most complete representations is the TAXIS system by Mylopoulos, Bernstein, and Wong (1980), which has aspects of all seven features, although some are developed further than others.

These systems are representative of many notations proposed for DB semantics and the ANSI/SPARC conceptual schema (Tsichritzis & Klug 1978). Most of them are strong in describing functional dependencies, but weak on domain roles and inference mechanisms. Only the ones that have a strong influence from AI and computational linguistics are rich enough to support a natural language interface to knowledge-based systems.

Conceptual Graphs

The Project on Data Base Semantics at the IBM Systems Research Institute is developing the theory of conceptual graphs (Sowa 1976, 1979a,b, & forthcoming) as a complete formalism for knowledge-based systems. Conceptual graphs were originally designed as a semantic representation for natural language, but were applied to data bases as a practical area for testing and refining the theory. The original paper (Sowa 1976) developed all of the seven features except definitional mechanisms, which were presented in Sowa (1979a). Heidorn's NLP system (1972) is being used to implement conceptual graphs, and a more complete presentation of them will be given in the forthcoming book.

References

Bachman, Charles W. (1969) "Data structure diagrams," *Data Base,* vol. 1, no. 2, pp. 4-10.

Brachman, Ronald J., & Brian C. Smith, eds. (1980) *SIGART Newsletter, Special issue on knowledge representation,* No. 70, Feb. 1980.

Chen, Peter Pin-Shan (1976) "The entity-relationship model—toward a unified view of data," *ACM Transactions on Data Base Systems,* vol. 1, no. 1, pp. 9-36.

Codd, E. F. (1978) "How about recently?" in B. Shneiderman, ed., *Databases: improving usability and responsiveness,* Academic Press, New York, pp. 3-28.

Feigenbaum, Edward A. (1977) "The art of artificial intelligence," *Proceedings of IJCAI-77,* MIT, Cambridge, Mass., pp. 1014-1029.

Findler, Nicholas V., ed. (1979) *Associative networks: representation and use of knowledge by computers,* Academic Press, New York.

Heidorn, George E. (1972) *Natural language inputs to a simulation programming system,* Technical report NPS-55HD72101A, Naval Postgraduate School, Monterey.

Hemphill, Linda G., & James R. Rhyne (1978) *A model for information representation in natural language query systems,* Technical report RJ2304, IBM San Jose.

Hendrix, Gary G., Earl D. Sacerdoti, Daniel Sagalowicz, & Jonathan Slocum (1978) "Developing a natural language interface to complex data," *ACM Transactions on Database Systems,* vol. 3, pp. 105-147.

Housel, B. C., V. Waddle, & S. B. Yao (1979) "The functional dependency model for logical database design," *Proceedings of the Fifth VLDB,* pp. 194-208.

McSkimin, James, & Jack Minker (1979) "A predicate calculus based semantic network for deductive searching," in Findler (1979).

Mylopoulos, John, Philip A. Bernstein, & Harry K. T. Wong (1980) "A language facility for designing interactive database-intensive applications," *Transactions on Database Systems,* vol. 5, pp. 185-207.

Roussopoulos, Nicholas D. (1976) *A semantic network model of data bases,* PhD Thesis, University of Toronto, Dept. of Computer Science.

Schank, Roger C., & Robert P. Abelson (1977) *Scripts, plans, goals and understanding,* Lawrence Erlbaum Associates, New York.

Smith, John M., & Diane C. P. Smith (1977b) "Database abstractions: aggregation and generalization," *ACM Transactions on Database Systems,* vol. 2, no. 2, pp. 105-133.

Sowa, John F. (1976) "Conceptual graphs for a data base interface," *IBM Journal of Research and Development,* vol. 20, No. 4, pp. 336-357.

Sowa, John F. (1979a) "Definitional mechanisms for conceptual graphs," in V. Claus, H. Ehrig, & G. Rozenberg, eds., *Graph grammars and their application to computer science and biology,* Springer Verlag, Berlin, pp. 426-439.

Sowa, John F. (1979b) "Semantics of conceptual graphs," in *Proceedings of the 17th Annual Meeting of the Association for Computational Linguistics,* pp. 39-44.

Sowa, John F. (forthcoming) *Conceptual structures: information processing in mind and machine,* Addison-Wesley, Reading, MA. To appear in fall of 1981.

Tsichritzis, Dennis, & Anthony Klug, eds. (1978) "The ANSI/X3/SPARC DBMS framework. Report of the study group on database management systems," *Information Systems,* vol. 3, pp. 173-191.

Waltz, David L. (1978) "An English language question answering system for a large relational database," *Communications of the ACM,* vol. 21, no. 7, pp. 526-539.