the 1980s by Koza (1992, 1994, 1999)). Such methods were part of a general movement for using biological ideas in computer science that started with pioneers such as von Neumann (*q.v.*), Turing (*q.v.*), and Wiener (*q.v.*), and continues today with evolutionary methods, neural networks, and methods based on the immune system, insect colonies, and other biological systems.

GA, EP, and ES methods differ in many details—in general, ESs and EP each define fairly specific versions of the evolutionary process described above, whereas the term "genetic algorithm," while originally referring to a specific algorithm, has come to refer to many considerably different variations of the basic scheme. ESs were originally formulated to work on real-valued parameter optimization problems, such as aircraft wing-shape optimization. In the original formulation of EP, candidate solutions to given tasks were represented as finite-state machines (*see* SWITCHING THEORY), which were evolved by randomly mutating their state transition diagrams and selecting the fittest. Since the early 1990s there has been much cross-fertilization among the three areas, and the original distinctions among GAs, ESs, and EP have blurred considerably in the current use of these labels, and a somewhat broader formulation has emerged, known as "evolutionary computation."

Setting the parameters for the evolutionary process (population size, selection strength, mutation rate, crossover rate, and so on) is often a matter of guesswork and trial and error, though some theoretical and heuristic (*q.v.*) guidelines have been discovered. An alternative is to have the parameters "self-adapt"—by changing their values automatically over the course of evolution in response to selective pressures. Self-adapting parameters are an intrinsic part of ESs and EP, and are the subject of much research in GAs.

GAs, as well as EP and ESs, have been applied widely. Examples of applications include numerical parameter optimization and combinatorial optimization, the automatic design of computer programs, bioengineering, financial prediction, robot learning, evolving production systems for artificial intelligence (*q.v.*) applications, and designing and training neural networks. There has been considerable success in combining EC methods with other types of search methods, such as simple gradient ascent and simulated annealing. Such hybrid models are thought by many to be the best approach to optimization in complex and ill-understood problem spaces (Davis, 1991).

In addition to these "problem-solving" applications, evolutionary methods have been used in models of natural systems in which evolutionary processes take place, including economic systems, immune systems, ecologies, biological evolution, evolving systems with adaptive individuals, insect societies, and more complex social systems. (*See* Goldberg (1989), or Mitchell (1996) for an overview of applications in some of these areas.)

Much current research in the evolutionary computation field is on making the basic framework more biologically realistic, both for modeling purposes and in the hope that more realism will improve the search performance of these methods.

Bibliography

1966. Fogel, L. J., Owens, A. J., and Walsh, M. J. *Artificial Intelligence through Simulated Evolution.* New York: John Wiley.

1975. Holland, J. H. *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press. (Second edition: MIT Press, 1992.)

1989. Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Reading, MA: Addison-Wesley.

1991. Davis, L. D. (ed.) *Handbook of Genetic Algorithms.* New York: Van Nostrand Reinhold.

1992. Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: MIT Press.

1992. Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs.* New York: Springer-Verlag.

1994. Koza, J. R. *Genetic Programming II: Automatic Discovery of Reusable Programs.* Cambridge, MA: MIT Press.

1995 Fogel, D. B. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* Los Alamitos, CA: IEEE Press.

1995. Schwefel, H-P. *Evolution and Optimum Seeking.* New York: John Wiley.

1996. Back, T. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms.* Oxford: Oxford University Press.

1996. Mitchell, M. *An Introduction to Genetic Algorithms.* Cambridge, MA: MIT Press.

1999. Koza, J. R., Forrest, H. B., Martin, A. K., and David, A. *Genetic Programming III: Darwinian Invention and Problem Solving.* San Francisco: Morgan Kaufmann.

Melanie Mitchell

GEOGRAPHIC INFORMATION SYSTEM (GIS)

For articles on related subjects *see* COMPUTER GRAPHICS; DATABASE MANAGEMENT SYSTEM; and RELATIONAL DATABASE.

Geographic information describes the locations, characteristics, and shapes of features and phenomena on the surface of the earth. Traditionally, such information has been produced, disseminated, and used in the form of paper maps and atlases, using various projections to allow the curved surface of the earth to be represented on flat paper. Increasingly, however, it has been possible to produce such information in digital form, and the advent of instruments capable of sensing the earth's surface from space accelerated this process. Today, vast amounts of geographic information are available for use in digital form, and a system that handles, processes, edits, manipulates, analyzes, and displays such data is a called a *geographic information system* (GIS). It is estimated that the total annual sales of GIS software exceed $1 billion, and that the total activity related to digital geographic information is at least an order of magnitude greater.

History and Applications

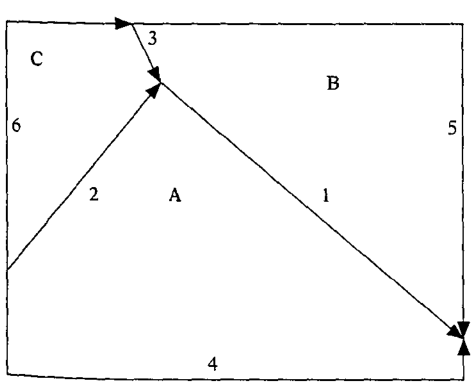
GIS began in the mid-1960s; the Canada Geographic Information System is often credited with being the first major project. It applied computer technology to the analysis of vast amounts of map data collected by the Canada Land Inventory (Foresman, 1998). GIS requires specialized input and output devices, including map-sized *digitizers*, high-resolution *scanners*, and pen *plotters*, and the development of these peripherals in the 1960s was a major impetus. However the major growth period of GIS began only in the early 1980s, following the development of *database management systems* (DBMS) that support a *relational database* and super-minicomputers like the VAX and Prime. Today a GIS is most often encountered as a networked application running on Unix (*q.v.*) or Windows NT systems, although simpler versions are available for use in the field on smaller platforms, including palmtops (*see* PORTABLE COMPUTERS).

GIS databases make use of both *raster* and *vector* representations, and although most software products have capabilities in both areas, application domains tend to be dominated by one or the other. Raster-based representations are most common where remote sensing is a major source of raw data, and where spatial resolution is limited, and are thus most likely to be found applied to natural resource management, agriculture, forestry, or environmental science. In vector-based representations, geographic phenomena are represented as discrete points, lines, areas, or volumes, with associated attributes. In addition, there will be support for the representation of *topological* relationships between objects, including connectivity (e.g. street or sewer networks) and adjacency (e.g. administrative subdivisions). A vector-based GIS is most likely to be found in transportation, assessment and management of land ownership records, infrastructure and utility maintenance, vehicle routing and scheduling, and related applications.

An important concept in GIS is the *layer*, defined as a representation of some specific variable, class of objects, or phenomenon over the geographic area of the database. The ability to combine information from different layers, and thus to analyze the relationships between facts about the same geographic location that may be separated on different paper maps, is an important characteristic of a GIS, and one that helps to distinguish it from computer-aided design (*see* CAD/CAM). A layered map produced by the ARC-INFO GIS is shown in Fig. 1 under SCIENTIFIC VISUALIZATION on Color Page CP-14.

Architecture

Most GIS products, particularly those focused on vector-based applications, are built on a standard relational DBMS. Because of the difficulty of handling variable-length strings of coordinates efficiently in a relational DBMS, the early products of the 1980s adopted a *hybrid* architecture in which the attributes of objects and the topological relationships between them are represented in relational tables, and the geometric form of the objects is represented in a custom database. For example, a map of population by county could be represented in three structures: a relational



A	1000
B	2000
C	2780

	R	*L*
1	A	B
2	A	C
3	C	B
4	0	A
5	B	0
6	C	0

1	$x1,y1,x2,y2,\ldots$
2	$x1,y1,x2,y2,\ldots$
3	$x1,y1,x2,y2,\ldots$
4	$x1,y1,x2,y2,\ldots$
5	$x1,y1,x2,y2\ldots$
6	$x1,y1,x2,y2\ldots$

Figure 1. A simple map of counties (left) and the three structures used to represent it in a vector-based GIS: a relational table of county names and attributes (upper middle); a relational table of county adjacencies that indicates the area to the right and left of each directed edge with 0 representing the outside (upper right); and a file of ordered coordinates for each common boundary.

table identifying the name of each county and its associated population; a relational table identifying each common boundary between two counties with the names of the adjacent counties; and a file of variable-length records, each containing the ordered pairs of coordinates needed to describe one common boundary. Fig. 1 shows a simple example of this basic GIS concept (note the use of 0 to denote the unmapped area, and the use of the order of coordinates to determine the right and left sides of each common boundary).

More recently, the improved performance of computer platforms has allowed integration of all three data structures within a relational DBMS, leading to a new architecture in which the DBMS is the sole repository of the GIS database. Object-oriented models are also used, although to date most GIS products that use this approach are actually implemented over a relational DBMS. A raster-based GIS generally does not use a database product.

Functionality

GISs are characterized by (a) the ability to represent a wide range of geographic information types, and (b) functions to perform a wide range of manipulations. These include support for changes of projections and coordinate systems; determination of relationships between objects, such as proximity, overlap, or connectivity; calculation of statistics, ranging from measures of length and area to complex statistical analysis; display, including support for many map-making functions; and support for decision-making. An important set of GIS applications lies in vehicle routing and scheduling in support, for example, of school bus or parcel delivery systems. Such applications implement a range of methods of operations research in a spatial context.

Research Issues

A GIS is a challenging application, and much research effort is being devoted to advancing the technology. Among many topics, research focuses particularly on issues of representation (extending the set of GIS data models to include time, the third spatial dimension, and scale); uncertainty (description and modeling of the errors and inaccuracies of geographic information); and cognition (design of better user interfaces, and simplification of user interaction). The term *geographic information science* is increasingly used to denote basic research issues surrounding the use of digital geographic information.

Sources of Information

Several excellent texts and state-of-the-art reviews are available; see, for example, Burrough and McDonnell (1998), Demers (1997), Longley *et al.* (1998), or Laurini and Thompson (1992). There are magazines devoted to GIS, including *GIS World* (published in Fort Collins, CO), and affiliated magazines are published in Europe and Asia. GIS research results are published in many journals, including the *International Journal of Geographical Information Systems*, *Transactions in GIS*, *Cartography and GIS*, and *Geoinformatica*, the last of which particularly encourages papers in the computer science aspects of GIS. Many international conference series address GIS research, including the International Symposium on Spatial Data Handling, the International Symposium on Spatial Databases, and the Conference on Spatial Information Theory.

Many Websites offer comprehensive resources on GIS, including `http://www.esri.com` (the Website of Environmental Systems Research Institute, Redlands, CA, a major GIS software developer), and `http://www.gisworld.com` (the Website of the magazine). The international UNIGIS consortium offers GIS courses by distance learning, and there are extensive curriculum materials available at `http://www.ncgia.ucab.org` (the Website of the US National Center for Geographic Information and Analysis).

Bibliography

1992. Laurini, R., and Thompson, D. *Fundamentals of Spatial Information Systems.* San Diego, CA: Academic Press.
1997. Demers, M. N. *Fundamentals of Geographic Information Systems.* New York: John Wiley.
1998. Burrough, P. A., and McDonnell, R. *Principles of Geographical Information Systems*, 2nd Ed. New York: Oxford University Press.
1998. Foresman, T. W. (ed.) *The History of Geographic Information Systems: Perspectives from the Pioneers.* Upper Saddle River, NJ: Prentice Hall.
1998. Longley, P. A., Goodchild, M. F., Maguire, D. J., and Rhind, D. W. (eds.) *Geographical Information Systems: Principles, Techniques, Management and Applications.* New York: John Wiley.

Michael Goodchild

GEOMETRY, COMPUTATIONAL

See COMPUTATIONAL GEOMETRY.

GLOBAL AND LOCAL VARIABLES

For articles on related subjects *see* BLOCK STRUCTURE; PROCEDURE-ORIENTED LANGUAGES; and SIDE EFFECT.

The entity denoted by a variable name in a computer program can generally be accessed (i.e. examined or changed) only in certain parts of the program. The domain of the program within which a variable name can be used is called the *scope* of the variable.