TRANSPORTATION RESEARCH
CIRCULAR
Number E-C113 January 2007

Artificial Intelligence in Transportation

Information for Application

TRANSPORTATION RESEARCH BOARD
OF THE NATIONAL ACADEMIES

TRANSPORTATION RESEARCH BOARD
2006 EXECUTIVE COMMITTEE OFFICERS

Chair: Michael D. Meyer, Professor, School of Civil and Environmental Engineering, Georgia Institute of Technology, Atlanta
Vice Chair: Linda S. Watson, Executive Director, LYNX–Central Florida Regional Transportation Authority, Orlando
Division Chair for NRC Oversight: C. Michael Walton, Ernest H. Cockrell Centennial Chair in Engineering, University of Texas, Austin
Executive Director: Robert E. Skinner, Jr., Transportation Research Board

TRANSPORTATION RESEARCH BOARD
2006 TECHNICAL ACTIVITIES COUNCIL

Chair: Neil J. Pedersen, State Highway Administrator, Maryland State Highway Administration, Baltimore
Technical Activities Director: Mark R. Norman, Transportation Research Board

Christopher P. L. Barkan, Associate Professor and Director, Railroad Engineering, University of Illinois at Urbana–Champaign, *Rail Group Chair*
Shelly R. Brown, Principal, Shelly Brown Associates, Seattle, Washington, *Legal Resources Group Chair*
Christina S. Casgar, Office of the Secretary of Transportation, Office of Intermodalism, Washington, D.C., *Freight Systems Group Chair*
James M. Crites, Executive Vice President, Operations, Dallas–Fort Worth International Airport, Texas, *Aviation Group Chair*
Arlene L. Dietz, C&A Dietz, LLC, Salem, Oregon, *Marine Group Chair*
Robert C. Johns, Director, Center for Transportation Studies, University of Minnesota, Minneapolis, *Policy and Organization Group Chair*
Patricia V. McLaughlin, Principal, Moore Iacofano Golstman, Inc., Pasadena, California, *Public Transportation Group Chair*
Marcy S. Schwartz, Senior Vice President, CH2M HILL, Portland, Oregon, *Planning and Environment Group Chair*
Leland D. Smithson, AASHTO SICOP Coordinator, Iowa Department of Transportation, Ames, *Operations and Maintenance Group Chair*
L. David Suits, Executive Director, North American Geosynthetics Society, Albany, New York, *Design and Construction Group Chair*
Barry M. Sweedler, Partner, Safety & Policy Analysis International, Lafayette, California, *System Users Group Chair*

TRANSPORTATION RESEARCH CIRCULAR E-C113

Artificial Intelligence in Transportation

Information for Application

Transportation Research Board
Artifical Intelligence and Advanced Computing Applications Committee

January 2007

Transportation Research Board
500 Fifth Street, NW
Washington, DC 20001
www.TRB.org

TRANSPORTATION RESEARCH CIRCULAR E-C113
ISSN 0097-8515

The **Transportation Research Board** is a division of the National Research Council, which serves as an independent adviser to the federal government on scientific and technical questions of national importance. The National Research Council, jointly administered by the National Academy of Sciences, the National Academy of Engineering, and the Institute of Medicine, brings the resources of the entire scientific and technical communities to bear on national problems through its volunteer advisory committees.

The **Transportation Research Board** is distributing this Circular to make the information contained herein available for use by individual practitioners in state and local transportation agencies, researchers in academic institutions, and other members of the transportation research community. The information in this Circular was taken directly from the submission of the authors. This document is not a report of the National Research Council or of the National Academy of Sciences.

Policy and Organization Group
Robert C. Johns, *Chair*

Data and Information Systems Section
Alan E. Pisarski, *Chair*

Artificial Intelligence and Advanced Computing Applications Committee
Gary S. Spring, *Chair*

Montasir Abbas
Baher Abdulhai
Ghassan Abu-Lebdeh
M. Asghar Bhatti
Ruey Long Cheu
Shih-Miao Chin
Mashrur A. Chowdhury
James Michael Cooper
Michael J. Demetsky
Richard C. Hanley
Sherif Ishak
Manoj K. Jha
Shinya Kikuchi
David B. Reinke
C. E. Tapie Rohm, Jr.
Adel W. Sadek
Kristen L. Sanford Bernhardt
Yung-Ching Shen
Carlos Sun
Hualiang (Harry) Teng
Dusan Teodorovic
Ramkumar Venkatanarayan
Ashley G. Williams
Billy M. Williams

Thomas M. Palmerlee, *Senior Program Officer*
David Floyd, *Senior Program Associate*

Transportation Research Board
500 Fifth Street, NW
Washington, DC 20001
www.TRB.org

Jennifer Correro, Proofreader and Layout

Foreword

The Transportation Research Board's Artificial Intelligence and Advanced Computing Committee (ABJ70) has as part of its mission to serve as a technical forum on the application of artificial intelligence (AI) to transportation problems, and to disseminate information about AI applications that is deemed credible and potentially useful to the transportation community. To this end, this Transportation Research Circular, created by members of ABJ70, provides six articles describing five general AI areas, namely, knowledge-based systems, neural networks, fuzzy sets, genetic algorithms, and agent-based models. It is designed to serve as an informational resource for transportation practitioners and managers with respect to AI tools within these general areas.

Each article, for its related AI paradigm, details the types of problems to which the paradigm is best suited, its strengths and weaknesses, example applications, and guidelines for its application. The articles are meant, as one of the authors states, as a sort of Cliff Notes for AI Applications in Transportation. In describing the state of the art vis a vis these areas of AI, it is hoped that better decisions will be made about what tools to choose, under what conditions and for what specific applications for a wide range of transportation problems.

—Gary S. Spring, Chair
TRB Artificial Intelligence and Advanced Computing Committee

Contents

Artificial Intelligence Applications in Transportation

ADEL W. SADEK
University of Vermont

At the turn of the 21st century, transportation professionals face challenges of increasing complexity. Transportation professionals are asked to meet the goals of providing safe, efficient, and reliable transportation while minimizing the impact on the environment and communities. This has turned out to be quite difficult given the constant increase in travel demand, fueled by economic development, and the ever-growing demands to do more with less. A partial listing of some of those challenges that transportation professionals face includes capacity problems, poor safety record, unreliability, environmental pollution, and wasted energy. Adding to the challenge is the fact that transportation systems are inherently complex systems involving a very large number of components and different parties, each having different and often conflicting objectives.

In recent years, there has been increased interest among both transportation researchers and practitioners in exploring the feasibility of applying artificial intelligence (AI) paradigms to address some of the aforementioned problems in order to improve the efficiency, safety, and environmental-compatibility of transportation systems. AI researchers, especially in the 1950s and 1960s, often adopted lofty goals for the field such as the development of general-purpose problem solvers. As transportation researches and professionals, however, our objective in researching AI applications to transportation is much more modest. Our interest is primarily to utilize the tools and methods that the AI community has developed to address real transportation problems that have been quite challenging to solve using traditional and classical solution methods. Given this, we adopt the following definition for AI in this circular: AI refers to methods and approaches that mimic biologically intelligent behavior in order to solve problems that so far have been difficult to solve by classical mathematics.

ARTIFICIAL INTELLIGENCE METHODS

At the present time, AI methods can be divided into two broad categories: (*a*) symbolic AI, which focuses on the development of knowledge-based systems (KBS); and (*b*) computational intelligence, which includes such methods as neural networks (NN), fuzzy systems (FS), and evolutionary computing. A very brief introduction to these AI methods is given below, and each method is discussed in more detail in the different sections of this circular.

Knowledge-Based Systems

A KBS can be defined as a computer system capable of giving advice in a particular domain, utilizing knowledge provided by a human expert. A distinguishing feature of KBS lies in the separation behind the knowledge, which can be represented in a number of ways such as rules, frames, or cases, and the inference engine or algorithm which uses the knowledge base to arrive at a conclusion.

Neural Networks

NNs are biologically inspired systems consisting of a massively connected network of computational "neurons," organized in layers (Figure 1). By adjusting the weights of the network, NNs can be "trained" to approximate virtually any nonlinear function to a required degree of accuracy. NNs typically are provided with a set of input and output exemplars. A learning algorithm (such as back propagation) would then be used to adjust the weights in the network so that the network would give the desired output, in a type of learning commonly called *supervised* learning.

Fuzzy Systems

Fuzzy set theory was proposed by Zadeh (1965) as a way to deal with the ambiguity associated with almost all real-world problems. Fuzzy set membership functions provide a way to show that an object can partially belong to a group. Classic set theory defines sharp boundaries between sets, which mean that an object can only be a member or a nonmember of a given set. Fuzzy membership functions allow for gradual transitions between sets and varying degrees of membership for objects within sets. Complete membership in a fuzzy function is indicated by a value of +1, while complete non-membership is shown by a value of 0. Partial membership is represented by a value between 0 and +1.

Figure 2 shows an example of a fuzzy membership function defined for the set of "medium traffic volume" on a certain highway. In this example, traffic volumes between 800 and 1,000 vehicles per hour (vph) fully belong to the medium traffic level set. Traffic volumes less than 400 vph or more than 1,400 vph would not be regarded as medium at all (membership function value = 0). However, values between 400 and 800 vph, or between 10,00 and 1,400 vph

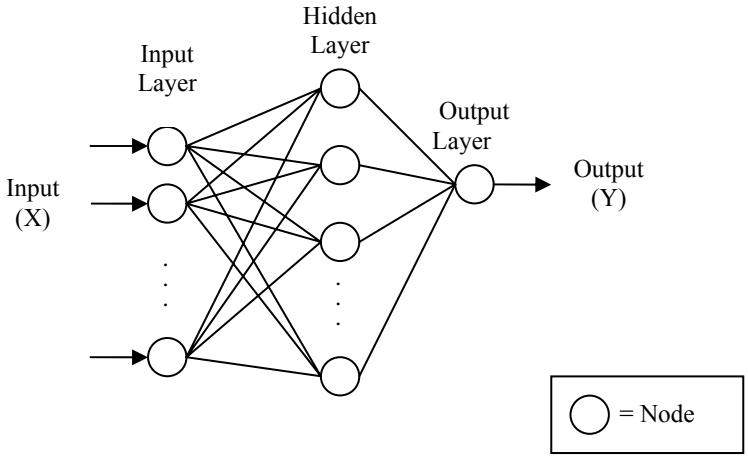

FIGURE 1 A multilayer neural network.

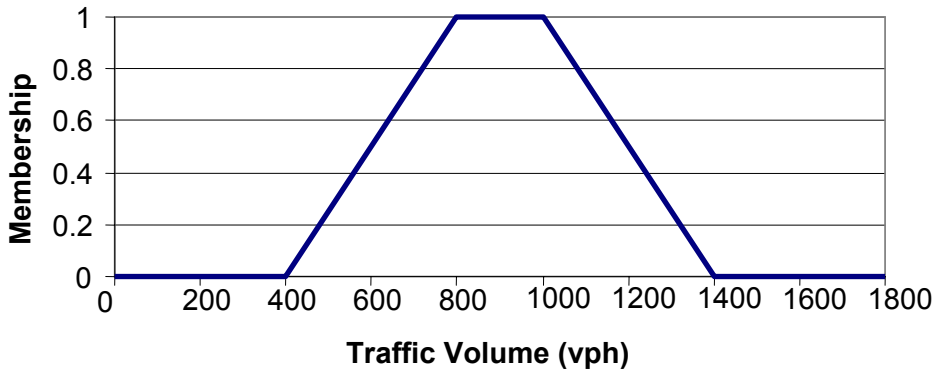

FIGURE 1 Example of a fuzzy membership function for medium traffic volume.

would have partial membership in the medium traffic level set. In a crisp set definition, on the other hand, only values between 800 and 1,000 vph would be regarded as medium, while all other values would not (for example, a traffic volume of 799 vph would not be regarded as a medium traffic level). The use of fuzzy set theory does not necessarily minimize uncertainty related to problem objectives or input values, but rather provides a standardized way to systematically capture and define ambiguity.

Genetic Algorithms

Genetic algorithms (GAs) are stochastic algorithms whose search methods are based on the principle of survival of the fittest. In recent years, GAs have been applied to a wide range of difficult optimization problems for which classical mathematical programming solution approaches were not appropriate. The basic idea behind GAs is quite simple. The procedure starts with a randomly generated initial population of individuals, where each individual or chromosome represents a potential solution to the problem under consideration. Each solution is evaluated to give some measure of its "fitness." A new population is then formed by selecting the more fit individuals. Some members of this new population undergo alterations by means of genetic operations (typically referred to as crossover and mutation operations) to form new solutions. This process of evaluation, selection, and alteration is repeated for a number of iterations (generations in GA terminology). After some number of generations, it is expected that the algorithm "converges" to a near-optimum solution.

In addition to the aforementioned AI methods, there has recently been an interest in a new modeling paradigm known as agent-based modeling (ABM). This modeling approach came out of research work in AI as well as in complex systems analysis. The idea behind ABM is to describe a system from the perspective of its constituent units. The approach is therefore quite appropriate for modeling complex systems whose behavior emerges as a result of interactions among the components making up the system. Since transportation systems exhibit almost all the characteristics of complex systems, ABM has been attracting a lot of attention within the transportation research community. Given this, ABM will be discussed in the last section of this circular.

A BRIEF HISTORY OF ARTIFICIAL INTELLIGENCE

The modern history of AI can be traced back to the year 1956 when John McCarthy proposed the term as the topic for a conference held at Dartmouth College in New Hampshire devoted to the subject. The initial goals for the field were too ambitious and the first few AI systems failed to deliver what was promised. After a few of these early failures, AI researchers started setting some more realistic goals for themselves. In the 1960s and the 1970s, the focus of AI research was primarily on the development of KBS or expert systems. During these years, expert systems technology were applied to a wide range of problems and fields ranging from medical diagnosis to inferring molecular structure to natural language understanding. The same period also witnessed early work on NNs, which showed how a distributed structure of elements could collectively represent an individual concept, with the added advantage of robustness and parallelism. However, the publication of Minsky and Papert's book *Perceptrons* in 1969, which argued for the limited representation capabilities of NN, led to the demise of NN research in the 1970s.

The late 1980s and the 1990s saw a renewed interest in NN research when several different researchers reinvented the back propagation learning algorithm (although the algorithm was really first discovered in 1969). The back propagation algorithm was soon applied to many learning problems causing great excitement within the AI community. The 1990s also witnessed some dramatic changes in the content and methodology of AI research. The focus of the field has been shifting toward grounding AI methods on a rigorous mathematical foundation, as well as to tackle real-world problems and not just toy examples. There is also a move toward the development of hybrid intelligent systems (i.e., systems that use more than one AI method) stemming from the recognition that many AI methods are complementary. Hybrid intelligent systems also started to use newer paradigms that mimic biological behavior such as GAs and fuzzy logic.

WHAT MAKES ARTIFICIAL INTELLIGENCE APPROPRIATE FOR TRANSPORTATION PROBLEMS?

Transportation problems exhibit a number of characteristics that make them amenable to solution using AI techniques. First, transportation problems often involve both quantitative as well as qualitative data. The fact that we often have to deal with qualitative data in transportation makes the use of expert and FS an obvious choice. Second, in transportation we often deal with systems whose behavior is very hard to model with traditional approach, either because the interactions among the different system components are not fully understood or because one is dealing with a lot of uncertainty stemming from the human component of the system. For such complex systems, building empirical models, based on observed data are, may be the only option remaining. NNs, given their universal function approximation capabilities, are perfect tools for building such models. Third, transportation problems often lead to challenging optimization problems that are quite challenging to solve using traditional mathematical programming techniques, either because the relationships are hard to specify analytically or because of the size of the problem and its computational intractability. For these problems, GAs may provide an alternative solution approach. Finally, the complex nature of transportation systems and the fact

that their behavior emerges as a result of interactions among the system components makes ABM techniques quite appropriate for study the behavior of the system.

ARTIFICIAL INTELLIGENCE APPLICATION AREAS

AI application areas are quite diverse. This section lists some of those application areas to which AI methods has been applied over the years, and explains how these may be relevant to transportation applications. Among the most important of AI application areas are the following:

- System identification and function approximation, which is concerned with building empirical dynamic models of systems from measured data, or mapping system inputs to outputs. As previously mentioned, in transportation systems, many of the interrelationships between the variables or components of a transportation system are not fully understood. Given this, empirical models are quite common.
- Nonlinear prediction focuses on the prediction of the behavior of systems where the relationship between input and output is not linear. This is often the case with transportation problems including predicting traffic demand, or predicting the deterioration of transportation infrastructure as a function of traffic, construction, and environmental factors.
- Control focuses on controlling a system so as to achieve a desired output. Control applications abound in transportation. Examples include signal control of traffic at road intersections, ramp metering on freeways, dynamic route guidance, positive train control on railroads, and air traffic control.
- Pattern recognition or classification describes a broad range of problems where the goal is to classify an object or put it in its right class or category. Pattern recognition is often associated with image processing, although many prediction problems can also be regarded as a pattern classification problem. Examples of pattern recognition or classification problems in transportation include automatic incident detection (i.e., classifying the traffic state as incident or incident free), image processing for traffic data collection and for identifying cracks in pavements or bridge structures. Another example of a transportation pattern recognition problem involves the very important area of transportation equipment diagnosis.
- Clustering refers to the problem of grouping cases with similar characteristics together, and identifying the number of groups or classes. For transportation, clustering could be used to identify specific classes of drivers based on driver behavior, for example.
- Planning refers to the act of formulating a program for a definite course of action intended to achieve a desired goal. In transportation, the goal of the transportation planning process is to identify the transportation needs of a community and to recommend the best course of action required to meet those demands, while taking into account the economic, social, and environmental impacts of transportation. AI-based decision support systems for transportation planning could be quite useful, especially when accurate analytical models are lacking, and when problems involve multiple stakeholders with often conflicting objectives.
- Design is a key activity of the transportation engineering profession, including geometric design of highways, interchange design, structural design of pavements and bridges, culvert design, retaining walls design, and guardrail design, to list a few examples. AI methods could add a lot to the value and capabilities of computer-aided design which is now commonly used for engineering design applications, by providing additional decision-support capabilities.

- Decision making refers to the cognitive process of selecting a course of action from among multiple alternatives. Transportation officials are continuously faced with challenging situations where a decision needs to be made. Examples of these situations include deciding whether to build a new road, how much money should be allocated to maintenance and rehabilitation activities and which road segments or bridges to maintain, and whether to divert traffic to an alternative route in an incident situation.
- Optimization refers to the study of problems in which one seeks to minimize or maximize a function by choosing values for a set of decision variables while satisfying a set of constraints. Optimization problems abound in transportation. Examples include designing an optimal transit network for a given community, developing an optimal shipping policy for a company, developing an optimal work plan for maintaining and rehabilitating a pavement network, and developing an optimal timing plan for a group of traffic signals.

PURPOSE AND SCOPE OF THIS CIRCULAR

The main objective of the current circular is to introduce the reader to some of those AI paradigms that have recently been applied to transportation problems. Specifically, the circular focuses on the following five paradigms:

1. KBS,
2. Artificial NNs,
3. FS,
4. GAs, and
5. ABM.

Besides this introduction, the circular is divided into five parts, each discussing one of the above mentioned paradigms. Following a brief description of the paradigm, each part describes the types of problems for which the paradigm or method is most appropriate, as well as the strengths and weakness of the method. Examples of the paradigm's application to specific transportation problems are provided, along with a description of variants or advanced implementations of the basic AI paradigm.

REFERENCES

Minsky, M. L., and S. Papert. *Perceptrons: An Introduction to Computational Geometry* (First edition). MIT Press, Cambridge, Massachusetts, 1969.

Russel, S. J., and P. Norvig. Artificial Intelligence: A Modern Approach. Prentice-Hall, Upper Saddle River, New Jersey, 2002.

Zadeh, L. A. Fuzzy Sets. *Information and Control*. Vol. 8, 1965, pp. 338–353.

Knowledge-Based Systems in Transportation

GARY SPRING
Merrimack College

There exist many excellent references on KBS. The purpose of this monograph is not to serve as one more such resource but rather to serve as a sort of Cliff Notes on the technology. The following few pages provide a description of the basic KBS paradigm, the types of problems to which it is best suited and why, guidelines for application and some discussion of advanced implementations of KBS.

THE BASIC PARADIGM

Research in AI focuses on replicating the analytical, problem solving, and learning capabilities of the brain using software. KBS, a subcategory of AI, bring the benefits of knowledge and intelligence to the solution of complex problems. Indeed, the power of these systems derives from their use of knowledge to reduce the number of problem solutions that need be considered.

KBSs were first introduced to the engineering profession in the 1970s and their use has grown and evolved throughout the intervening years. In the 30 years that have passed since the first documented KBS (the trinity of classic systems: DENDRAL, MYCIN, and PROSPECTOR) were reported, the basic architecture of KBS has changed little. In general, the defining feature of KBS as compared to other software systems is their separation of knowledge about a problem from the process by which the problem is solved. That is, so-called domain knowledge (knowledge base) and algorithmic control of the program (traditionally called the inference engine) are separate. The explicit separation of knowledge from control makes it easier to add new knowledge or remove existing knowledge when necessary. Hopgood (1992) makes an analogy to the functioning of our brains, whose control processes are approximately unchanging in nature while individual behavior is constantly modified by new knowledge and experience (updating the knowledge base). Other defining features include an interface from which the inference process is transparent, a readable knowledge base, a capability to grow and change and an ability to operate under uncertainty (Fenves, 1986).

In practice, these systems have three components: a knowledge base in the form of rules, frames or objects, for example; an inference engine in the form of algorithms on how to control the processing of knowledge; and a database which may be thought of to be the system's window on the world.

Many AI systems have been placed under the rubric of KBS, including expert systems, case-based reasoning, agent-based, FS, and many others. KBSs have been applied, in some cases very successfully, to transportation problems for more than 20 years. Indeed, more than 200 AI-related systems have been described in the literature during this time (Figure 3). Of these, almost 90% were some form of KBS. They held much promise as powerful problem-solving tools—solving problems that heretofore it had not been possible to solve using software.

In recent years, emphasis has been less on developing independent KBS and more on integrating them into other paradigms, such as geographic information systems (GIS), object-oriented databases and even artificial NN. Indeed, one can see the use of the basic KBS concepts, described

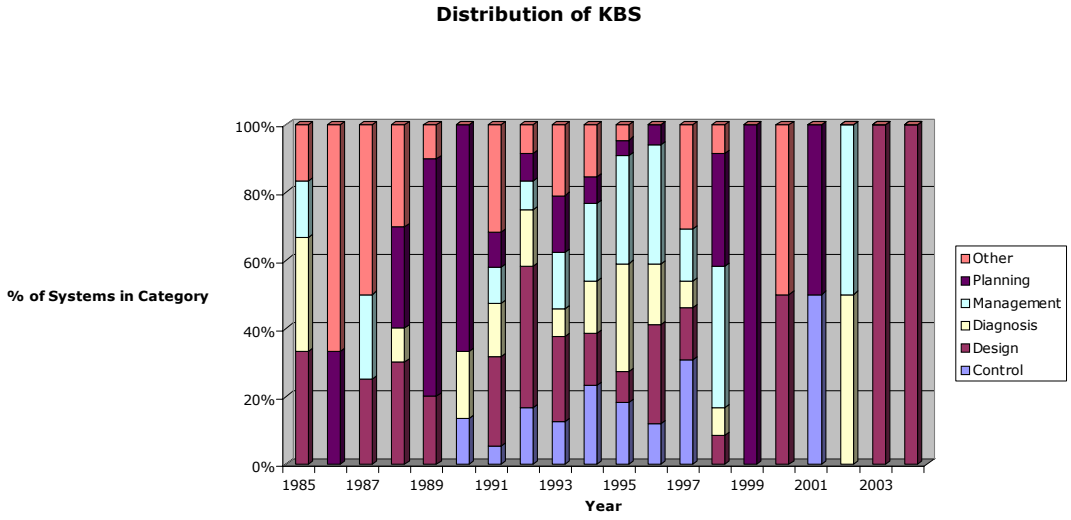

FIGURE 3 Summary of transportation-related KBS appearing in the literature (1985–present).

below, in spreadsheets, word processing software, and other every day applications as well.

Knowledge

The knowledge component of KBS consists of a set of independent knowledge elements in the form of rules, frames, or objects. The choice of which form to use depends largely upon the problem to be solved and the tools that are available for use in coding the system. Rules of the form "if X, then Y" are the most common way of representing knowledge because they are most often the way we express our heuristic knowledge. They are therefore eminently understandable, fairly easy to extract from humans, and are very portable—thus allowing the system flexibility in the addition or change to its knowledge.

Frames are slightly more complex in that they represent knowledge by association and taxonomies. They are very closely aligned with object-oriented systems in that both provide ways to represent and organize information. The frame provides slots that contain information about the object being represented. Similar to object-oriented systems, and unlike static database systems, information can take the form of facts, rules, procedures, or pointers to other frames. Unlike database systems, frames are meant to capture the essence of concepts or stereotypical situations, for example being in a living room or going out for dinner, by clustering all relevant information for these situations together. This includes information about how to use the frame, information about expectations (which may turn out to be wrong), information about what to do if expectations are not confirmed, and so on. A great deal of procedurally expressed knowledge may be part of the frames. Collections of such frames are organized in frame systems in which the frames are interconnected—called "classes" in object-oriented terminology. Frames are very useful for causal and commonsense knowledge.

Reasoning

The inference engine establishes the focus for a particular problem and decides upon actions to take. Common strategies for control in rule-based systems are backward chaining, forward chaining, or some mixture of the two. Forward chaining uses known facts and rules about data to generate hypotheses. This strategy is especially appropriate in situations where data are expensive to collect, but few in quantity (Figure 4).

Backward chaining requires beginning with a goal and then searching through a set of facts and rules in order to satisfy the goal. Backward chaining is useful in situations where the quantity of data is potentially very large and where some specific characteristic of the system under consideration is of interest. Typical situations are various problems of diagnosis or forensics. For example, in solving a diagnosis problem, one needs to begin by collecting as much information as possible in order to form alternative hypotheses that may then be assessed (forward chaining). Then using these hypotheses, one can examine the data and what we know about the data to make some diagnoses (backward chaining).

Reasoning using a frames representation approach relies more on matching and the hierarchy of the system than deduction as is the case with rules. The ability to attach procedures and characteristics to frames and the arrangement into hierarchies and classes, have been adopted for development of object-oriented systems. An example of reasoning with a system representing a highway intersection with a frame in which its "slots" contain number of legs, crash experience, type of control, volumes, algorithms to access data base software and manipulate the resulting data, and thus determine level of service (LOS), delays, etc., and perhaps rules governing likelihood that it is a "hazardous" location.

Using this unique separation of knowledge from control, there exist several software tools that provide the inferring mechanism with an empty knowledge base to be filled in by the buyer. These are commonly known as "shells." Many of these now are available for implementation on the Internet and some are platform independent, using a Java-based code.

Types of Problems

Hayes (1994) pointed out that KBS are principally used for three reasons:

1. To improve the reasoning of the applications system—time-critical systems (such as real-time control), for example, benefit from speed of light access to all knowledge, all of the time, consistently applied.
2. To increase the flexibility of the applications system—the ability to solve problems with incomplete information and that are not completely formulated increases system flexibility substantially, and
3. To increase the human-like qualities of the system—the ability to provide cogent explanations about why decisions are made makes for much better interfaces and increases trust in the system.

Examples of problems that are appropriate for KBS solution in transportation include: diagnosing hazardous highway locations, planning construction activities, designing structural members for and/or assessing the structural integrity of bridges, scheduling airline maintenance activities, dispatch and control of rail and transit, developing traffic management strategies given

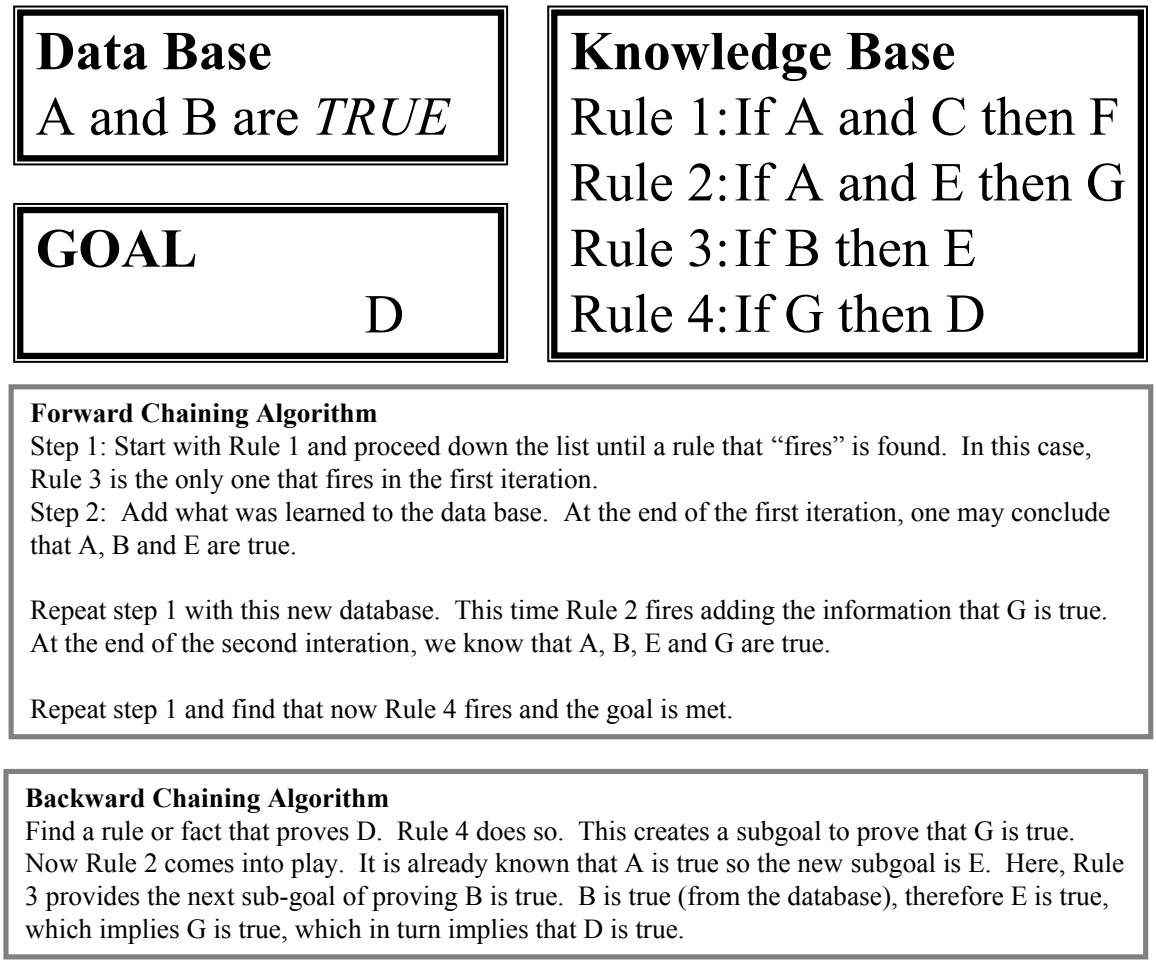

FIGURE 4 Reasoning examples.

a traffic disaster, and intelligent transportation systems (ITS). The sheer diversity of disciplines involved and complexities that may be encountered in the Transportation Engineering problem domain provides a rich environment for KBS development. Problems most amenable to KBS solution, as typified by the systems summarized in Figure 3, either suffer from lack of data—in which heuristics may be used to "fill in the holes"—or they are poorly defined or are too complex such that standard solutions using analytical or simulation tools may not be appropriate. For problems such as these last, heuristics are used as decision support—for example, design of a signal plan for a complex network of intersections and roads; or diagnosis of problems at a high crash signalized intersection; crash data collection; recommending speed limits in speed zones; and providing diagnostic safety reviews for intersection designs. These last three are all examples of systems that have actually been implemented (Thielman, 2007; Kindler, et al, 2002; Srinivasan, 2006).

Key questions that must be answered in helping to decide upon which type of tool to use include: Is there an analytical or simulation tool that could be used to solve the problem at hand? Would the problem best be solved using these more traditional techniques? For example, the determination of queue length at a signalized intersection or even the LOS of that intersection

would be more amenable to analytical models than to KBS. Determination of the operational parameters of a complex network of intersections and roads would probably best be done using simulation models. Design of that complex network or diagnosis of its problems or its real-time control on the other hand may best be conducted using a KBS since these types of problems are characterized by missing data, complexity, and time-criticality. In short, the type of problem to be addressed drives the decision as to type of tool to be used (for example, matching and optimization problems are not amenable to KBS solution whereas the others described above do benefit from the application of knowledge).

STRENGTHS

KBS offer many significant advantages over their traditional counterpart tools. It has already been mentioned that they allow engineers to work with uncertain problems. Most problems of any complexity involve some level of uncertainty—either from data quality or some other source. Many are such that we are willing to live with that uncertainty but for some we are not. KBS allow us to express concepts in ways in which we are more comfortable (the concepts of fairly good, somewhat old, and so on) and to avoid problems with crisp boundaries such as using delay levels to assess the LOS of highway intersections.

It is possible to consider problems requiring judgment and that are not amenable to a procedural approach. Design and evaluation problems are excellent examples of this type of problem. KBS are designed to improve with experience. By their nature, with knowledge separate from control, these systems are easily updated based upon experience.

Many of the applications listed in Figure 4 require that time-critical decisions, based upon copious, often simultaneous information, be made and disseminated quickly and accurately. This type of response requires one who is well versed in handling emergencies and who is able to make decisions quickly. This well-versed person has access to a tremendous amount of knowledge mainly derived from work experience in the area. Thus, the same problem faced in other application areas requiring special expertise must be faced here as well. Experts are rare, they are expensive and it is often difficult to retain enough of them for an adequate length of time for safe and effective operations. This means that valuable expertise is sometimes available only sporadically and at significant cost to the user. These are among the reasons KBS offers such potential. KBSs have been used extensively in a variety of different areas, most notably ITS, in attempts to help meet the goals of improved safety and efficiency. Perhaps one of the most compelling reasons for using knowledge-based tools is their ability to use all available knowledge, consistently and without error or misjudgment, and to work with uncertainty—thus providing more reliable and consistent decisions, more useful information, and improved reaction times. Finally, in some cases, these systems are used to actually codify "substantive insights in and assumptions of" problems.

KBS also hold great promise as educational tools, where even simple knowledge bases can have practical value for education. Work in developmental psychology indicates that actual "learning" must take place by "doing" (Piaget, 1970; Feigenbaum, 1982). Of course, such a system is not necessarily a good teacher of the material but nevertheless would expose students, in an interactive and nonthreatening way, to expert reasoning processes as well as to his or her domain knowledge. Another important advantage of using KBS as teaching aides is the capability of pooling heuristic knowledge into a common repository. This type of knowledge is

not normally published, and the only way it is shared is between teacher–student or master–apprentice.

WEAKNESSES

Unfortunately, many, especially in the early years of AI applications in transportation, have been carried away with all of this wonderful potential and have become enamored with the hype. Consequently, very often KBS have been used for all types of problems under all conditions. The fact is that these systems are indeed powerful problem solvers and they hold great promise for the solution of a plethora of problems. However, they are not a panacea and they have some major drawbacks in their application—mainly, that they often only have surface knowledge about the problem at hand. The best of these systems have a great deal of surface knowledge about a much focused subset of a problem—and very little about anything else. For example, IF car will not start, THEN check battery. The system has no information about the relationship the battery has with the ability of the car to start—it only has the heuristic to check the battery in this instance. The fact that they can be used to enhance our understanding of problems notwithstanding, there often exist a temptation to use these systems as "black boxes."

Additionally, obtaining the knowledge for the KBS is and always has been a major concern, sometimes the main bottleneck in developing such systems. Finding the expert and then figuring out how to elicit knowledge from him is often a difficult process, and can be extremely expensive.

Once implemented, the KBS model is often slow and unable to access or manage large volumes of information; and once implemented, it can be difficult to maintain. Solutions to these problems have been sought through better knowledge elicitation techniques and tools, better KBS shells and environments, improved development methodologies, knowledge modeling languages, facilitating the cooperation between KBS and databases in expert databases and deductive databases, and techniques and tools for maintaining systems.

Even using off the shelf "shells," implementing KBS is a difficult process requiring special skills and often taking many person-years. They can be very expensive. Perhaps this is why there have been so few actual implementations of the 200+ systems described in the transportation literature over the past 20 years.

GUIDELINES FOR APPLICATION

There are many steps in developing a successful KBS. The following three are a distillation of those that are critical to success.

1. Determine if your problem is appropriate for a KBS tool versus a conventional tool. Do conventional tools do what you need to do? Would an analytic or simulation model be better applied to the problem for example? In the case of modeling applications where viable methologies exist both in the mathematical and soft computing domains there are clearly trade offs to be evaluated in model selection. For example, there may be a trade-off between the potential for new insight versus ease of implementation or between the motivation to inform the modeling with accurate prior knowledge versus the aversion to biasing the results through

misconceptions and faulty assumptions. Explicit presentation of the evaluation of these kinds of trade-offs is often missing from papers on transportation modeling applications.

2. Establish an evaluation plan for the system at the outset. At a minimum, the plan should include system goals, specifications and constraints, and measures of effectiveness. This helps to assure that the system is designed to facilitate its own validation and verification.

3. Assure that you have the resource commitment for full development, implementation and maintenance. This will include staff requirements, developer salaries, time commitment of individuals knowledgeable about the domain of interest, software (and possibly hardware) costs, and so on.

ADVANCED SYSTEMS

KBS serves as an umbrella phrase representing a wide variety of systems whose common theme is the use of knowledge and heuristics to solve problems. In the foregoing pages the two most commonly used paradigms (rules and frames or object-oriented) were described. The area of AI continues to grow however, and thus there are always emerging paradigms to be discussed. Two such paradigms that are fast becoming part of the mainstream AI tools for transportation applications, and briefly discussed in the following pages are case-based reasoning (CBR) systems and agent-based systems (ABS).

Case-Based Reasoning

CBR systems use as their primary knowledge source a database of stored cases recording specific prior episodes rather than generalized heuristics. In CBR, new solutions are generated not by chaining, but by retrieving the most relevant cases from memory and adapting them to new situations. That is, CBR solves new problems by adapting previously successful solutions to similar problems. Thus in CBR, reasoning is based on remembering.

A complete CBR process can be represented as a cycle consisting of the following tasks: (*a*) retrieve; (*b*) reuse; (*c*) revise; and (*d*) retain (Figure 5). As previously mentioned, at the core of the CBR process is a case-base that stores previous instances of problems and their derived solutions. When faced with a new problem, a CBR system matches the new problem against cases in the case base, and retrieves the most similar case(s). Since the retrieved case is likely to be somewhat different from the current case, a CBR system typically adapts the retrieved solution to closely suit the new problem during the reuse step. The proposed solution is then implemented and tested for success; any revisions are then made, if needed. Finally, the new case is retained, allowing the system to learn and refine its knowledge with usage.

CBR systems are attractive because they directly address one of the most difficult and most costly problems described earlier, namely the elicitation of knowledge. CBR does not require an explicit domain model nor the involvement of expensive experts in system development. Elicitation therefore becomes a task of gathering case histories and implementation is reduced to identifying significant features that describe a case—an easier task than creating an explicit model. By applying database techniques, large volumes of information can be managed, and CBR systems can learn by acquiring new knowledge as cases are processed. This makes maintenance easier as well.

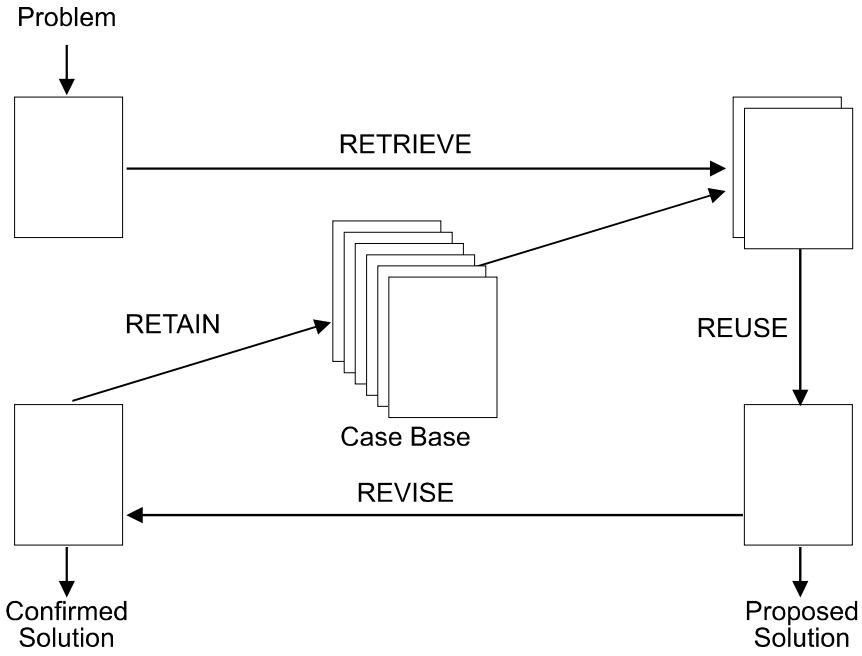

FIGURE 5 The CBR cycle.

Agents

A software agent is a computational entity that is capable of autonomous behavior by virtue of a small number of simple rules that make each agent aware of the options available to it when faced with a decision-making task related to its domain of interest. Furthermore, such an entity is seen as part of a community of similar software processes that are designed to interact with each other, often acting cooperatively to achieve mutual goals. Therefore, the two key features of agents are autonomy and communal interaction. ABM is explained in more detail in the last part of this circular.

Autonomy implies intelligence in that entities are directed toward specified goals; the level of intelligence required directly relates to the complexity of this goal and the associated heuristics involved. Of course since these entities use a form of knowledge base to direct their behavior they are also capable of reasoning about their behavior and interactions.

Communal interaction does not imply agents are actually sending messages to one another. They may merely be cooperating by carrying out a shared task without actually sending messages to one another. Cooperation without communication, however, may be seen as a special case.

Interfaces through which agents may interact must have common specifications. The agents must have an agreed upon architecture that would apply to all agents within a community. To these ends, the Foundation for Intelligent Physical Agents has developed specifications of four agent-based application areas:

- Personal travel assistance: individualized, automated access to travel services;

- Audiovisual entertainment and broadcasting: negotiating, filtering, and retrieving audiovisual information, in particular for digital broadcasting networks;
- Network management and provisioning: automated provisioning of dynamic virtual private network services where a user wants to set up a multimedia connection with several other users; and,
- Personal assistant: management of a user's personal meeting schedule, in particular in determining the time and place arrangements for meetings with several participants."

Numerous applications exist for such software agents, including air traffic control (Steeb, 1988).

CONCLUSION

As noted in this monograph, there have been few real-world KBS applications in the area of transportation—for reasons listed. However, KBSs are now routinely used in thousands of real-world applications. Most such applications involve relatively small knowledge bases, containing hundreds rather than thousands of units (objects, rules, frames, cases). The next generation of KBSs could involve knowledge bases containing hundreds of thousands or even millions of units. They will need to perform well in increasingly complex, time-critical environments. This is a daunting task, but it promises huge benefits in terms of safe and efficient transportation of our traveling public.

REFERENCES

Feigenbaum, E. A., A. Barr, and P. R. Cohen. *The Handbook of Artificial Intelligence*, Vol. 2, William Kaufman, Inc., 1982.

Fenves, S. J. What is an Expert System. In *Expert Systems in Civil Engineering*. Proceedings of a Symposium sponsored by the Technical Council on Computer Practices at the ASCE 1986 Annual Meeting, 1986.

Fikes, R. E., and T. Kehler. The Role of Frame-Based Representation in Knowledge Representation and Reasoning. *Communications of the ACM*, Vol. 28, No. 9, pp. 904–920, 1985.

Hayes, P. J. The Logic of Frames. In *Frame Conceptions and Text Understanding* (D. Metzing, ed.). deGruyter, Berlin, 1980, pp. 46–61.

Hayes-Roth, F., and N. Jacobstein. The State of Knowledge-Based Systems. *Communications of the ACM*, Vol. 37, No. 3, 1994.

Hopgood, A. A. *Knowledge-Based Systems for Engineers and Scientists*. CRC Press, 1992.

Kindler, C. E., D. W. Harwood, N. D. Antonucci, I. Potts, T. R. Neuman, and R. M. Wood. *Development of an Expert System for the Interactive Highway Safety Design Model*. FHWA, U.S. Department of Transportation, 2002.

Minsky, M. A Framework for Representing Knowledge. In *The Psychology of Computer Vision* (P. Winston, ed.). McGraw-Hill, New York, 1975, pp. 211–277.

Piaget, J., (D. Coltman, Trans.), *Science of Education and the Psychology of the Child.* Viking, l970.

Srinivasan, R. *Expert Systems for Recommending Speed Limits in Speed Zones*. In NCHRP Project 3-67, Transportation Research Board of the National Academies, Washington, D.C., 2006.

Steeb, R., S. Cammarata, F. A. Hayes-Roth, P. W. Thorndyke, and R. B. Wesson. Distributed Intelligence for Air Fleet Control. In *Readings in Distributed Artificial Intelligence* (A. H. Bond, and L. Gasser, eds.). Morgan-Kaufmann, 1988.

Thielman, C. Y. *Expert Systems for Crash Data Collection*. FHWA-RD-99-052. FHWA, U.S. Department of Transportation, 1999.

Neural Networks

SHERIF ISHAK
Louisiana State University

FRANCO TRIFIRÒ
University of Messina, Italy

Neural networks (NNs), or connectionist systems, have experienced a resurgence of interest in recent years as a paradigm of computational and knowledge representation. After a first surge of attempts to simulate the functioning of the human brain using artificial neurons in the 1950s and 1960s, this AI subdiscipline did not receive much attention until the 1990s. The resurgence has been due mainly to the appearance of faster digital computers that can simulate large networks and the discovery of new NN architectures and more powerful learning mechanisms. The new network architectures, for the most part, are not meant to duplicate the operation of the human brain, but rather to receive inspiration from known facts about how the brain works.

NNs are concerned with processing the information by a learning process and by adaptively responding to inputs in accordance with a learning rule. These powerful models are composed of many simulated neurons or simple computational units that are connected in such a way that they are able to learn in a manner similar to how human brains learn. This distributed architecture makes NNs particularly appropriate for solving nonlinear problems and input–output mapping problems. The usual application of NNs is in the area of learning and generalization of knowledge and patterns. They are not suitable for expert reasoning and they have poor explanation capabilities.

While there are several definitions for NNs, the following definition emphasizes the key features of such models. An NN can be defined as a distributed, adaptive, generally nonlinear learning machine built from interconnecting different processing elements (PEs) (Principe et al., 2000). The functionality of NNs is based on the interconnectivity between the PEs. Each PE receives connections from other PEs and/or itself. The connectivity defines the topology of NN and plays a role at least as important as the PEs in the NN's functionality. The signals transmitted via the connections are controlled by adjustable parameters called weights, w_{ij} .

A typical PE structure is depicted in Figure 6 as a nonlinear (static) function applied to the sum of all the PE's inputs. Due to the fact that NNs' knowledge is stored in a distributed fashion through the connection weights between PEs and also the fact that the knowledge is acquired through a learning process that involves modification of the connection strengths between PEs, NNs tend to resemble in functionality the human brain.

There are many types of NN architectures, each designed to address a particular class of problems such as system identification, function approximation, nonlinear prediction, control, pattern recognition, clustering, feature extraction, and others. NNs may also be classified as either static or dynamic. Static networks represent good function approximators with the ability to build long-term memory into their synaptic weights during training. On the other hand, dynamic networks have a built-in mechanism to produce an output based on more than one time instant in the past, establishing what is commonly referred to as short-term memory.

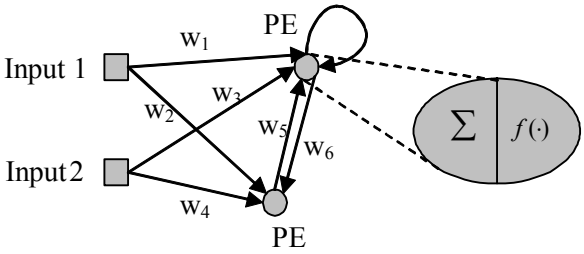

FIGURE 6 Example of a neural network.

The development process of NN models is typically carried out in two stages: training and testing. During the training stage an NN learns from the patterns presented in an existing dataset. The performance of the network is consequently evaluated using a testing dataset that is composed of patterns the network was never exposed to before. Because the learned knowledge is extracted from training datasets, NNs are considered both model-based and data-driven systems. Usually the learning phase uses an algorithm to adjust the connection weights, based on a given dataset of input–output pairs. Training patterns are presented to the network repeatedly until the error of the overall output is minimized. The presentation of all patterns once to the network is called an epoch and results in adjustment of the connection weights such that the network performance is improved. The training stage of NN is terminated when the error drops below a prespecified threshold value or when the number of epochs exceeds a certain prespecified limit. Another method to control the efficiency of the training stage is to monitor the network performance (errors) during the training stage on a cross-validation (CV) dataset, usually smaller than the learning dataset. The role of CV is to test for the network's generalization capabilities during the training process. If the network is overtrained a sudden degradation of the network based on the CV data will trigger the training process to stop.

THE BASIC PARADIGM: MULTILAYER PERCEPTRON

There are different types of NN. The most commonly used architecture of NN is the multilayer perceptron (MLP). MLP is a static NN that has been extensively used in many transportation applications due to its simplicity and ability to perform nonlinear pattern classification and function approximation. It is, therefore, considered the most widely implemented network topology by many researchers (see for instance, Duda et al., 2001; Ham and Kostanic, 2001). Its mapping capability is believed to approximate any arbitrary mathematical function.

MLP consists of three types of layers: input, hidden, and output. It has a one-directional flow of information, generally from the input layer, through hidden layer, and then to the output layer, which then provides the response of the network to the input stimuli. In this type of network, there are generally three distinct types of neurons organized in layers. The input layer contains as many neurons as the number of input variables. The hidden neurons, which are contained in one or more hidden layers, process the information and encode the knowledge within the network. The hidden layer receives, processes, and passes the input data, to the output layer. The selection of the number of hidden layers and the number of neurons within each affects the accuracy and performance of the network. The output layer contains the target output vector.

Figure 7 depicts an example of MLP topology. A weight coefficient is associated with each of the connections between any two neurons inside the network. Information processing at the neuron level is done by an "activation function" that controls the output of each one.

NNs train through adaptation of their connection weights based on examples provided in a training set. The training is performed iteratively until the error between the computed and the real output over all training patterns is minimized. Output errors are calculated by comparing the desired output with the actual output. Therefore, it is possible to calculate an error function that is used to propagate the error back to the hidden layer and to the input layer in order to modify the weights. This iterative procedure is carried out until the error at the output layer is reduced to a prespecified minimum or for a prespecified number of epochs. The back-propagation algorithm is most commonly used for training MLP and is based on minimizing the sum of squared errors between the desired and actual outputs.

Actual validation of an already trained NN requires testing the network performance on an exclusive set of data, called testing data, which is composed of data that was never presented to the network before. If the error obtained in both training and testing phases is satisfactory, the NN is considered adequately developed and thus can be used for practical applications.

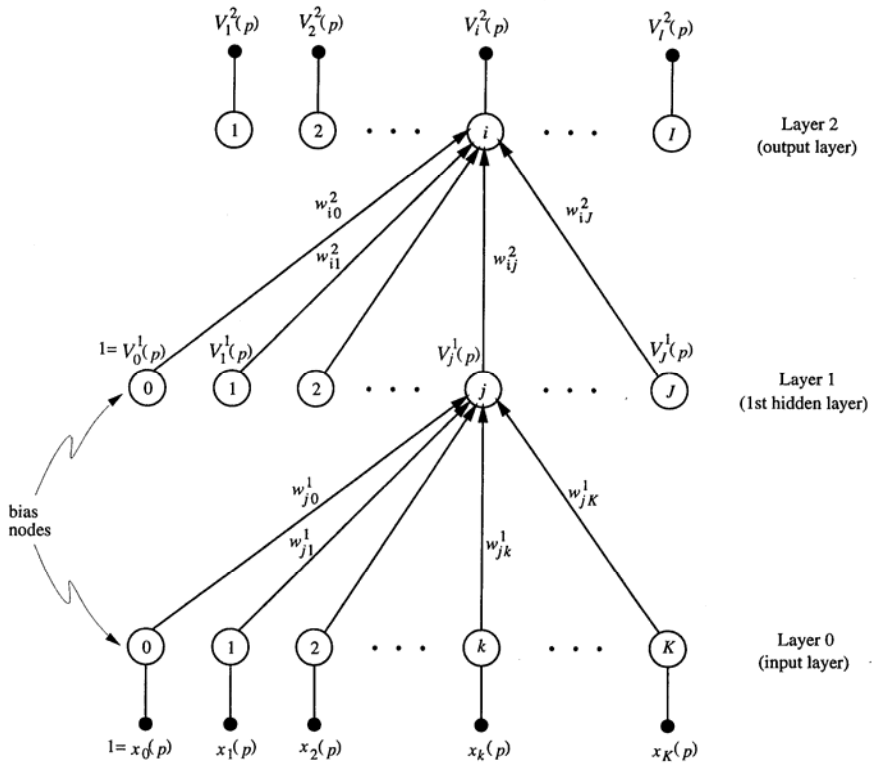

FIGURE 7 Example of MLP network topology.

ADVANCED TOPOLOGIES

In addition to the basic MLP architecture, several other advanced topologies have been developed in the past few years to meet the needs of different types of applications. Although NN and other soft computing constituents may perform exceptionally well when used individually, the development of practical and efficient intelligent tools may require a synergic integration of several topologies to form hybrid systems. In fact, computational intelligence and soft computing fields have witnessed in the past few years an intensive research interest towards integrating different computing paradigms such as fuzzy set theory, GAs, and NNs to generate more efficient hybrid systems. The emphasis is placed on the synergistic, rather than the competitive, way the individual tools act to enhance each other's application domain. The purpose is to provide flexible information processing systems that can exploit the tolerance for imprecision, uncertainty, approximate reasoning, and partial information to achieve tractability, robustness, low-solution cost, and close resemblance with human-like decision making (Pal et al. 2001).

For example, a combination of neural and fuzzy set, or neuro–fuzzy, model may consolidate the advantages of both techniques. When combined, they can be easily trained and have known properties of convergence and stability as NNs, and they can also provide a certain amount of functional transparency through rule dependency which is important to understand the solution of a problem. NN and GA could be combined to solve optimization problems. In fact, this hybrid approach could be applied using the properties of NN to define the observed functions with unknown shape, and the GA, to obtain the final result of an optimization problem. Examples of advanced and hybrid NN topologies include:

- Modular networks,
- Hybrid principal component analysis,
- Coactive neuro–fuzzy inference system (CANFIS),
- Jordan–Elman network,
- Partially recurrent network (PRN), and
- Time-lagged feed-forward network (TLFN).

Modular Network

Modular networks are a special class of multiple parallel feed-forward MLPs. The input is processed with several MLPs and then the results are recombined. The topology used specifically for this application is composed of two primary components: local expert networks and a gating network (Jang et al., 1997; Principe et al., 2000).

Figure 8 shows the topology of a modular network. The basic idea is linked to the concept of "divide and conquer," where a complex system is better attacked when divided into smaller problems, whose solutions lead to the solution of the entire system. Using a modular network, a given task will be split up among some local expert networks, thus reducing the load on each in comparison with one single network that must learn to generalize from the entire input space. Then, the modular NN architecture builds a bigger network by using modules as building blocks. A very common method is to construct an architecture that supports a division of the complex task into simpler tasks.

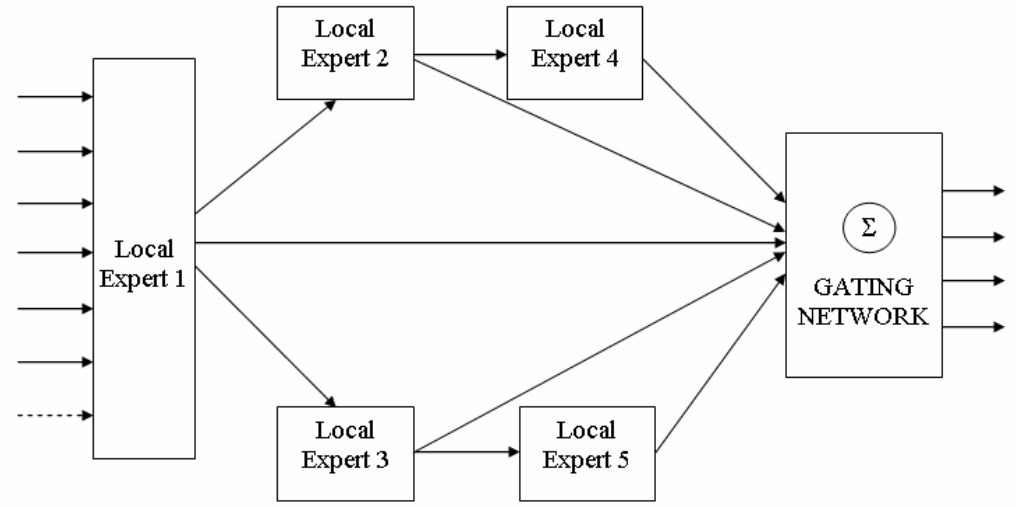

FIGURE 8 Example of the modular network topology.

All modules are NN. The architecture of a single module is simpler and the subnetworks are smaller than a monolithic network. Due to the structural modifications, the task the module has to learn is in general easier than the whole task of the network. This makes it easier to train a single module (SO). In a further step, the modules are connected to a network of modules rather than to a network of neurons. The modules are independent to a certain level which allows the system to work in parallel. This NN type offers specialization of a function in each sub-module and does not require full interconnectivity between the MLP's layers. A gating network eventually combines the output from the local experts to produce an overall output. For this modular approach, it is always necessary to have a control system to enable the modules to work together in a useful way. The evaluation using different real world data sets showed that the new architecture is very useful for high-dimensional input vectors. For certain domains, the learning speed and the generalization performance in the modular system is significantly better than in a monolithic multilayer feed-forward network (Ablameyko et al. 2003).

Hybrid Principal Component Analysis Network

Hybrid principal component analysis (PCA) is a technique that finds an orthogonal set of directions in the input space and provides a way to find the projections into these directions in an orderly fashion. The orthogonal directions are called eigenvectors of the correlation matrix of the input vector and the projections the corresponding eigen values. PCA has the ability to reduce the dimensionality of the input vectors, and therefore, can be used for data compression. When used in conjunction with MLP, the PCA can reduce the number of inputs to the MLP and improve its performance. The PCA projects the input vector onto a smaller dimensional space, thus compressing the input for the MLP network. It should be emphasized that PCA is a well-known statistical procedure that is used in feature extraction from high-dimensional space (see Duda et al., 2001; Ham and Kostanic, 2001; Jang et al., 1997). The topology of the hybrid PCA network is illustrated in Figure 9.

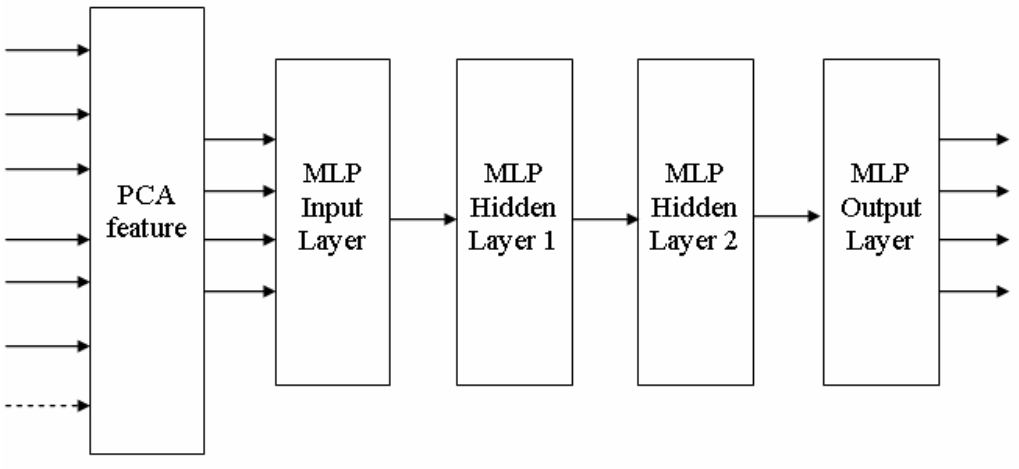

FIGURE 9 Example of the PCA network topology.

Coactive Neuro–Fuzzy Inference System

CANFIS belongs to a more general class of adaptive neuro–fuzzy inference systems (ANFIS) (Jang et al., 1997). CANFIS may be used as a universal approximator of any nonlinear function. The characteristics of CANFIS are emphasized by the advantages of integrating NN with fuzzy inference systems (FIS) in the same topology. The powerful capability of CANFIS stems from pattern-dependent weights between the consequent layer and the fuzzy association layer. The architecture of CANFIS is illustrated in Figure 10.

The fundamental component for CANFIS is a fuzzy neuron that applies membership functions (MFs) to the inputs (see the section on FS in this circular). Two membership functions are commonly used: general bell and Gaussian (Lefebvre, 2001). The network also contains a normalization axon to expand the output into a range of 0 to 1. The second major component in this type of CANFIS is a modular network that applies functional rules to the inputs. The number of modular networks matches the number of network outputs, and the number of processing elements in each network corresponds to the number of MFs. CANFIS also has a combiner axon that applies the MFs outputs to the modular network outputs. Finally, the combined outputs are channeled through a final output layer and the error is back-propagated to both the MFs and the modular networks.

The function of each layer is described as follows. Each node in Layer 1 is the membership grade of a fuzzy set (A, B, C, or D) and specifies the degree to which the given input belongs to one of the fuzzy sets. The fuzzy sets are defined by three membership functions. Layer 2 receives input in the form of the product of all output pairs from the first layer. The third layer has two components. The upper component applies the membership functions to each of the inputs, while the lower component is a representation of the modular network that computes, for each output, the sum of all the firing strengths. The fourth layer calculates the weight normalization of the output of the two components from the third layer and produces the final output of the network.

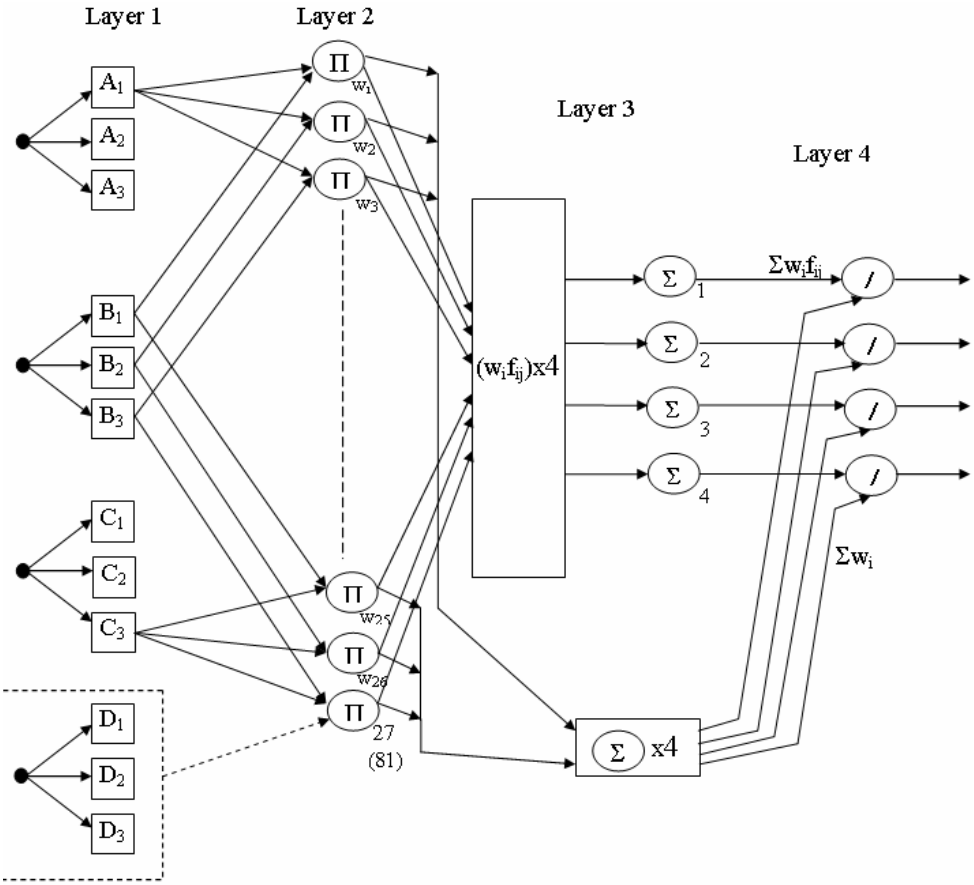

FIGURE 10 Example of CANFIS network topology.

Jordan–Elman Network

The Jordan–Elman network is also referred to as the simple recurrent network (SRN) (Ham and Kostanic, 2001). It is a single hidden-layer feed-forward network with feedback connections from the outputs of the hidden-layer neuron to the input of the hidden layer (Principe et al., 2000). It was originally developed to learn temporal sequences or time-varying patterns. As shown in Figure 11 the network contains context units located in the upper portion and used to replicate the hidden-layer output signals.

The context units are introduced to resolve conflicts arising from patterns that are similar, yet result in dissimilar outputs. The feedback provides a mechanism to discriminate between identical patterns occurring at different times. The context units are referred to as a low-pass filter that creates a weighted average output of some of the more recent past inputs. They are also called “memory units” since they tend to remember information from past events. The training phase of this network is achieved by adapting all the weights using standard back-propagation procedures. More details on this topology can be found in Ham and Kostanic (2001) and Lefebvre (2001).

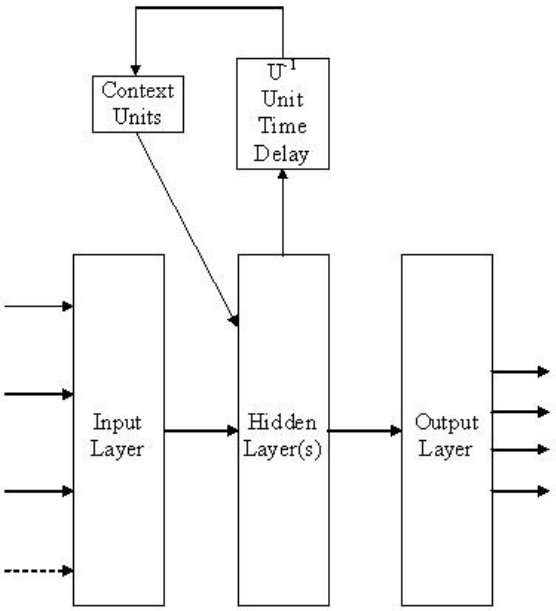

FIGURE 11 Example of Jordan–Elman network topology.

Partially Recurrent Network

PRN is considered a simplified version of the Jordan–Elman network without hidden neurons. It is composed of an input layer of source and feedback nodes, and an output layer, which is composed of two types of computation nodes: output neurons and context neurons. The output neurons produce the overall output, while the context neurons provide feedback to the input layer after a time delay. The topological structure of the network is illustrated in Figure 12. More details can be found in Haykin (1998) and Lefebvre (2001).

Time-Lagged Feed-Forward Network

In dynamic NN time is explicitly included in mapping input-output relationships. As a special type, TLFN extends nonlinear mapping capabilities with time representation by integrating linear filter structures in a feed-forward network. The type of topology is also called focused TLFN and has memory only at the input layer. The TLFN is composed of feed-forward arrangement of memory and nonlinear processing elements. It has some of the advantages of feed-forward networks such as stability, and can also capture information in input time signals. Figure 13 shows a simplified topological structure of the focused TLFN. The figure shows that memory PEs are attached in the input layer only. The input-output mapping is performed in two stages: a linear time-representation stage at the memory PE layer and a nonlinear static stage between the representation layer and the output layer. Further details underlying the mathematical operations of TLFN can be found in Ham and Kostanic (2001), Principe et al. (2000), and Lefebvre (2001).

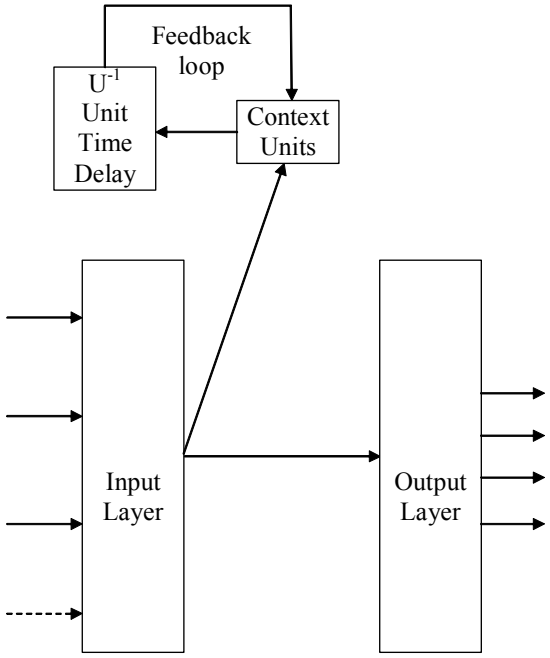

FIGURE 12 Example of PRN topology.

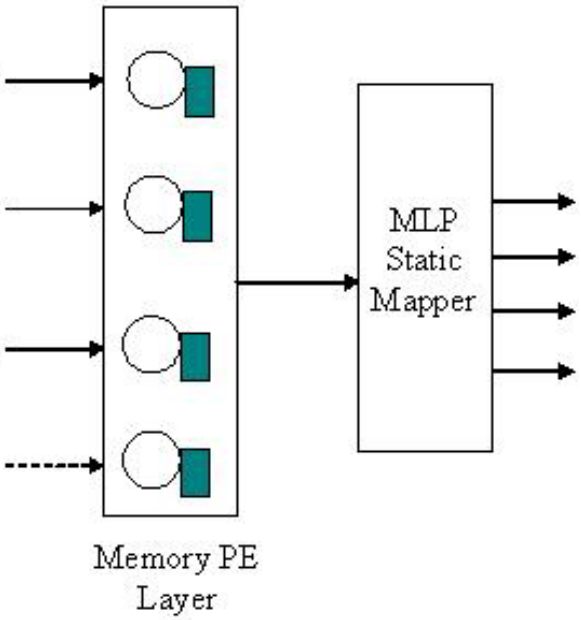

FIGURE 13 Example of TLFN topology.

NEURAL NETWORK APPLICATIONS DOMAIN

The inherent parallel architecture and the fault tolerance nature of NN are appealing to address problems in variety of application areas. NNs find their application in pattern recognition (classification, clustering, feature classification), image compression, image processing, system identification, and prediction. Neural models appear to have great potential for enhancing condition assessment and performance prediction modeling. In this section, we focus on just two representative transportation application domains (namely pavement management and engineering, and short-term traffic prediction) and provide a review of recent applications of NN to these two fields.

Pavement Management and Engineering

NNs have been used in a wide range of applications in the field of pavement management and engineering. Several models have been developed to predict pavement's conditions as well as to recommend appropriate maintenance strategies. Some examples are provided below.

An NN was used for roughness prediction of a flexible pavement, expressed as International Roughness Index (IRI) (La Torre et al., 1998). The application was performed using simulation data to calibrate the NN. Network performance was then verified using data obtained from experimental surveys. NNs were also used to predict the present serviciability rating (PSR) of pavements (Shekharan, 1998). The input variables were structural number, age and cumulative equivalent single-axle loads. Moreover, a partitioning method of connection weights was used to determine the relative contribution of each input variable to PSR prediction. Several NNs were used to determine the general visual condition index (VCI) of flexible pavements using distress data collected through visual assessments of the pavement surface (Van der Gryp et al., 1998). The networks were compared with classical methods. The results indicated the feasibility of using NNs for determining the VCI of pavement surfaces.

A dynamic NN was used to perform a reliable and accurate time-dependent roughness prediction model for newly constructed Kansas jointed plain concrete pavements (Felker et al., 2003). To achieve this objective, relevant data was obtained from the historical Kansas pavement condition database. The developed model produced output values very close to the measured IRI values. An overall pavement condition prediction methodology using NN was implemented (Yang et al., 2003). In particular, three individual NN models were developed to predict the three fundamental parameters used by Florida Department of Transportation (FDOT) for pavement evaluation purposes: crack rating, ride rating, and rut rating. The NNs were trained and tested using data from the FDOT pavement condition database.

A decision analysis framework based on past experience of rubber removal operations at Singapore airport was realized using NNs (Fwa et al., 1997). This was carried out with the aim of reducing the reliance on a few experienced maintenance staff for such an operation, and for improving the consistency and continuity of the rubber removal decision-making process. NNs were used to develop an automatic procedure for screening and recommending roadway sections for pavement preservation (Flintsch et al., 1998). The NN was used to learn the knowledge from past project selections. It was then trained using data representing the pavement's condition, the characteristics at the time of selection, and the sections selected for pavement preservation program for several years. NNs were used for selecting more appropriate strategies for repairing pavement distress of an airport rigid pavement to ensure optimum pavement performance (Lee et

al., 2002). In that study, experts were surveyed to compile expert knowledge that was then used to train the network.

A methodology to derive the optimal weights for known sets of pavement condition and operating parameters of a given road was proposed (Fwa et al., 2002). It consisted of two phases. The first phase used a GA (see the section on GA in this circular) to determine the optimal weights for the specified inputs of pavement condition and environmental operation. The second phase consisted of training a NN for speedy selection of priority weights for any given pavement condition under given operating environment. An NN was used to develop a sideway force coefficient (SFC) prediction model (Bosurgi and Trifirò, 2005). Their results demonstrate that NNs were capable of correctly interpreting the phenomenon modeled and capturing the internal correlations existing between the variables.

Short-Term Traffic Prediction

The short-term traffic prediction problem, which is concerned with attempting to forecast future traffic volumes, speeds or travel times, has been receiving increased attention in the last few years, especially given the interest in ITS and real-time traffic management and control. Lately, several studies have investigated the use of NNs for this problem. For instance, Park and Rilett (1998) proposed two modular NN models for forecasting multiple-period freeway link travel times. One model used a Kohonen Self Organizing Feature Map (SOFM) while the other utilized a fuzzy c-means clustering technique for traffic patterns classification. Rilett and Park (1999) proposed a one-step approach for freeway corridor travel time forecasting rather than link travel time forecasting. They examined the use of a spectral basis neural network with actual travel times from Houston, Texas.

Another study by Abdulhai et al. (1999) used an advanced time delay neural network (TDNN) model, optimized using a GA, for traffic flow prediction. The results of the study indicated that prediction errors were affected by the variables pertinent to traffic flow prediction such as spatial contribution, the extent of the loop-back interval, resolution of data, and others. Lint et al. (2002) presented an approach for freeway travel time prediction with state-space NNs. Using data from simulation models they showed that prediction accuracy was acceptable and favorable to traditional models. Several other studies applied NNs for predicting speed, flows, or travel times For instance, Park et al. (1999) used a spectral basis NN (SNN) to predict link travel times for one to five time periods ahead (of 5-min duration). They used traffic data collected from the TransStar system implemented in Houston. They found that the NN approach outperformed other statistical and heuristic approaches the Kalman filtering model, exponential smoothing model, and historical profile.

In a study by Maschavan Der Voort et al. (1996) a hybrid method of short-term traffic forecasting is introduced. The technique uses a Kohonen SOFM as an initial classifier and each class has an individually tuned ARIMA model associated with it. It was therefore was called KARIMA. It is believed that the explicit separation of the tasks of classification and functional approximation improves the forecasting performance, as compared to either a single ARIMA model or a backpropagation neural network. The model is tested with data from a French motorway, by forecasting traffic flow at horizons of 30 and 60 min.

Zhang et al. (1997) trained a multilayer feed-forward NN to address a freeway traffic system state identification problem. For this purpose, the authors used simulated traffic data from an artificially generated freeway. Several scenarios were generated, such as different demand

patterns and randomly generated incidents. The speed was predicted at a one time-step prediction horizon of 15 s duration. The solution was developed with the purpose of building an improved freeway traffic model that could be used for developing real-time predictive control strategies for dynamic traffic systems.

Zhang (2000) developed a recursive traffic flow prediction algorithm using NNs. The system prediction model is specified based on the understanding of how disturbances in traffic flow are propagated. Although the methodology presented has the advantage of its applicability to other linear and nonlinear function approximation predictors than NNs, it also has a shortcoming. The prediction is made at one-time step horizon of 30-s duration. The practicability of using such short prediction horizons or the effect of increasing the time step size was not considered.

In a study by Yasdi (1999) the effectiveness of a NN model for prediction of traffic volume based on time series data is presented. A dynamic NN, namely a Jordan–Elman recurrent network, was employed in this study to predict weekly, daily, and hourly based traffic volume. Fu and Rilett (2000) presented an NN-based method for estimating route travel times between individual localities in an urban traffic network. The methodology developed in this study assumes that route travel times are time-dependent and stochastic and their means and standard deviations have to be estimated.

In a study by Ishak et al. (2003 and 2004) an optimized NN-based methodology for short-term prediction horizons of traffic conditions was presented. It was found that the performance of different NNs families can be improved if traffic conditions and the number and type of the input parameters are considered. Up to 20-min point speed predictions are performed using the real traffic data and significant improvements were demonstrated.

STRENGTHS AND WEAKNESSES

The main advantages of NNs are their learning capabilities and their distributed architecture that allows for highly parallel implementation. When used for function approximation or for input-output mapping, a unique advantage of NNs lie in the fact that they do not require the user to specify the model form a priori (although the user still needs to decide upon the network architecture and the number of hidden layers and hidden nodes, for example). NNs are also excellent pattern classifiers and can be very effectively used for pattern recognition and classification problems. Finally, NNs allow the cause–effect relationships that are at the basis of complex multivariable systems to be reconstructed; they can make generalizations, and are particularly appropriate in those cases in which there is a significant amount of examples available.

On the negative side, the major criticism of NNs has always been that they are black boxes. The knowledge stored with the network structure is not transparent, but rather stored in the form of the weights of the network's connectors. Therefore, the use of NNs requires the availability of enough data to allow for the correct training and testing of the network. The problem has to be precisely characterized by the selected inputs and outputs. Otherwise this model cannot be used or can result in significant errors. Another problem with NNs is the relative difficulty of applying them compared to other more traditional approaches such as regression analysis. In fact, some agencies and transportation engineers still have reservations about implementing them. One possible approach to facilitate acceptance is to provide NN

methodology as an alternative to traditional analysis in available software. In this case, users will be able to test the new technologies for themselves, and may then adopt them if they prove to be more effective than the traditional tools for a particular application.

GUIDELINES—OR PITFALLS TO AVOID

The following main steps should be distinguished in every network design:

- Collection of prior information;
- Construction of examples;
- Selection of model structure;
- Model parameter estimation; and
- Model validation.

The first phase is characterized from the correct interpretation of phenomenon to examine. In fact, the construction of input–output data depends on prior knowledge about the problem. Afterwards, it is necessary to individuate the input and output variables. The choice of these variables is very important because the exactness and accuracy of analysis depends on this phase. The second phase consists of selecting the examples. Sometimes it could be appropriate to carry out specific measurement surveys to construct the examples. It is important that the acquired input–output data cover all the important factors of the problem.

Many different approaches can be applied in model identification depending on the prior information available, the goal of modeling, what aspects are to be considered, etc. Moreover, it is necessary to decide upon the network's architecture and the characteristic parameters.

To construct a neural network, its architecture must be first selected, and then the free parameters of the architecture must be determined. To select the architecture, the type, the numbers of neurons and their organization have to be determined. The values of the free parameters can be determined using the network's adaptive nature, which is their learning capability. In particular, it is necessary to divide the examples into two sets (training and testing). This makes it possible to train and to test the network; testing verifies the strength of the trained network to make generalization.

After the network has been trained, the final step of model identification is validation. For validation a proper criterion as a fitness of the model has to be used. The choice of this criterion is extremely important because it determines the measure of quality for the model. Validation tests typically address the following measures: mean square error, R^2 coefficient, and error autocorrelation in the two phases.

From the result of the validation it can be decided if the model is good enough for the intended purpose. If it isn't good enough, an iterative cycle of selection of model structure, selection of the network structure, model parameter estimation, and model validation must be repeated until a suitable representation is found. Then the model identification is an iterative process.

Since NNs are data-driven systems, the training patterns must cover the entire solution space to ensure sufficient representation of the data, and consequently, improve the network ability to generalize from the training data. Caution must also be exercised during training to avoid overtraining, which may result in a continuous improvement of performance with the

training dataset and degradation of performance with the validation dataset. If overtrained, NN tends to behave as a lookup table (i.e., memorize from training patterns) and its generalization ability is negatively impacted. Overtraining typically occurs when the same data is presented to the network at the learning stage for too many epochs. To avoid overtraining, a CV dataset should be used to monitor the network performance during training. Once the CV performance begins to deteriorate, the training process should stop since training beyond this point will cause the network to begin to memorize.

SUMMARY

In the last couple of decades NNs have been widely used to solve various transportation problems that defy traditional modeling approaches. A plethora of research efforts have shown that NNs can be most efficient and effective when addressing complex problems for which an accurate and complete analytical description is often too difficult to obtain, and yet can be easily represented by examples or patterns. NNs are particularly useful in applications of function approximation, pattern recognition, and pattern classification, to name a few. There exists a wide spectrum of architectures as presented earlier, each suited for specific applications (e.g., pavement management, short-term traffic prediction, incident detection, etc.) Flexibility and adaptability are two of the most powerful features in neural network architectures, which continue to expand this computational paradigm and its potential to tackle a large number of problems in the area of transportation engineering.

REFERENCES

Abdulhai, B., H. Porwal, and W. Recker. Short-Term Freeway Traffic Flow Prediction Using Genetically Optimized Time-Delay-Based Neural Networks. Presented at 78th Annual Meeting of the Transportation Research Board, Washington, D.C., 1999.

Ben-Akiva, M., A. de Palma, and I. Kaysi. Dynamic Network Models and Driver Information Systems. *Transportation Research A*, Vol. 25, No. 5, 1991, pp. 251–266.

Bosurgi, G., and F. Trifirò. A Model Based on Artificial Neural Networks and Genetic Algorithms for Pavement Maintenance Management. *International Journal of Pavement*, Vol. 6, No. 3, 2005, pp. 201–209.

Croney, D., and P. Croney. *Design and Performance of Road Pavements*. McGraw-Hill Professional, New York, 1997.

Duda, R., P. Hart, and D. Stork. *Pattern Classification* (2nd Edition). John Wiley and Sons, Inc., New York, 2001.

Felker, V., M. Hossain, Y. Najjar, and R. Barezinsky. Modeling the Roughness of Kansas PCC Pavements: Dynamic ANN Approach. Presented at 82nd Annual Meeting of the Transportation Research Board, Washington, D.C., 2003.

Flintsch G. W., J. P. Zaniewski, J. Delton, and A. Medina. Artificial Neural Network Based Project Selection Level Pavement Management System. In *Proc., Fourth International Conference on Managing Pavements*, Durban, South Africa, 1998, pp. 451–464.

Fu, L., and L. R. Rilett. Estimation of Time-Dependent, Stochastic Route Travel Times Using Artificial Neural Networks. *Transportation Planning and Technology*, Vol. 24, No. 1, 2000, pp. 25–36.

Fwa, T. F., W. T.Chan, and C. T. Lim. Decision Framework for Pavement Friction Management of Airport Runways. *Journal of Transportation Engineering,* November–December 1997.

Fwa, T. F., W. T.Chan, and Y. Liu. Priority Weighting Factors for Pavement Maintenance Management. Presented at 81st Annual Meeting of the Transportation Research Board, Washington, D.C., 2002.

Ham, F. M., and I. Kostanic. *Principles of Neurocomputing for Science and Engineering*. McGraw Hill, New York, 2001.

Haykin, S. *Neural Networks: A Comprehensive Foundation*. Macmillan College Publishing Company, Inc., 1994.

Ishak, S., and C. Alecsandru. Optimizing Traffic Prediction Performance of Neural Networks under Various Topological, Input, and Traffic Condition Settings. *ASCE Journal of Transportation Engineering*, Vol. 130, No. 1, 2004.

Ishak, S., P. Kotha, and C. Alecsandru. Optimization of Dynamic Neural Networks Performance in Short-Term Traffic Prediction. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1836*, Transportation Research Board of the National Academies, Washington, D.C., 2003, pp. 45–56.

Jang, J. S. R., C. T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing*. Prentice-Hall, N.J., 1997.

Kaysi, I., M. Ben-Akiva, and H. Koutsopoulos. Integrated Approach to Vehicle Routing and Congestion Prediction for Real-Time Driver Guidance. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1993,* Transportation Research Board of the National Academies, Washington, D.C., 1993, pp. 66–74.

La Torre, F., L. Domenichini, and M. I. Darter. Roughness Prediction Based on the Artificial Neural Network Approach. In *Proc., Fourth International Conference on Managing Pavements,* Vol. 2, Durban, South Africa, 1998, pp. 599–612.

Lee, C., C. F. Chen, S. Huang, and C. Hsu. Application of Neural Network for Selection of Airport Rigid Pavement Maintenance Strategies. Presented at 81st Annual Meeting of the Transportation Research Board*,* Washington, D.C., 2002.

Lefebvre, C. *Neuro Solutions, Version 4.10*. NeuroDimension, Inc., Gainesville, Fla., 2001.

Lint, V., S. P. Hoogendoorn, and H.J. Zuylen. Freeway Travel Time Prediction with State-Space Neural Networks: Modeling State-Space Dynamics with Recurrent Neural Networks. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1811*, Transportation Research Board of the National Academies, Washington, D.C., 2002, pp. 30–39.

Mascavan Der Voort, M., M. Dougherty, and S. Watson. Combining Kohonen Maps with ARIMA Time Series Models to Forecast Traffic Flow. In *Transportation Research Part C*, Vol. 4, No. 5, 1996, pp. 307–318.

Pal, S. K., T. S. Dillon, and D. S. Yeung. *Soft Computing in Case-Based Reasoning*. Springer-Verlag London Limited, Great Britain, 2001.

Park, D., and L. Rilett. Forecasting Multiple-Period Freeway Link Travel Times Using Modular Neural Networks. In *Transportation Research Record 1617*, 1998, p. 163–170.

Park, D., L. R. Rilett, and G. Han. Spectral Basis Neural Networks for Real-Time Travel Time Forecasting. *ASCE Journal of Transportation Engineering*, Vol. 125, No. 6, November–December 1999, pp. 515–523.

Principe, J. C., N. R. Euliano, and W. C. Lefebvre. *Neural and Adaptive Systems*. John Wiley and Sons, Inc., New York, 2000.

Rilett, L., and D. Park. Direct Forecasting of Freeway Corridor Travel Times Using Spectral Basis Neural Networks. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1752*, TRB, National Research Council, Washington, D.C., 2001, p. 140–147.

Shekharan, A. R. Effect of Noisy Data on Pavement Performance Prediction by Artificial Neural Networks. In *Transportation Research Record 1643*, TRB, National Research Council, Washington, D.C., 1998, pp. 7–13.

Ablameyko, S., L. Goras, M. Gori, and V. Piuri. *Neural networks for Instrumentation, Measurement and Related Industrial Applications*. IOS Press, Series III: Computer and Systems Sciences, Vol. 185, 2003.

Van der Gryp, A., S. J. Bredenhann, M. G. Henderson, and G. T. Rohde. Determining the Visual Condition Index of Flexible Pavements using Artificial Neural Networks. In *Proc., Fourth International Conference on Managing Pavements*, Durban, South Africa, 1998, pp. 115–129.

Yang, J., J. J. Lu, M. Gunaratne, and Q. Xiang. Overall Pavement Condition Forecasting Using Neural Networks: Application to Florida Highway Network. Presented at 82nd Annual Meeting of the Transportation Research Board, Washington, D.C., 2003.

Yasdi, R. Prediction of Road Traffic Using a Neural Network Approach. *Neural Computing and Applications*, Vol. 8, 1999, pp. 135–142.

Zhang, H., S. G. Ritchie, and Z. P. Lo. Macroscopic Modeling of Freeway Traffic Using an Artificial Neural Network. In *Transportation Research Record 1588,* TRB, National Research Council, Washington, D.C., 1997, pp. 110–119.

Zhang, H. M. Recursive Prediction of Traffic Conditions with Neural Network Models. *ASCE Journal of Transportation Engineering*, Vol. 126, No. 6, 2000, pp. 472–481.

Fuzzy Sets Theory Approach to Transportation Problems

SHINYA KIKUCHI
Virginia Tech

Many topics in transportation planning and engineering can be characterized as subjective, ill-defined, ambiguous, and vague. Decision-making processes for transportation investment, traveler's choice of routes and modes, and driver's behavior are typical examples that are not entirely based on the clear-cut decision criteria. Often, the difficulty of dealing with these decision problems are caused by the analyst's attempt to view the answers in the frame of binary logic; in other words, to seek an answer in one of two worlds, yes or no, or wrong or right, and nothing between with no uncertain term. As the scope of transportation analysis proliferates and as the consequences of transportation decisions pose far-reaching impacts on many non-transportation aspects, the analysis of transportation must inevitably deal with the types of uncertainty that are different from the traditional form, which has been handled by probability theory.

This part of the circular introduces fuzzy sets theory as a paradigm to deal with some of the difficulties that are related to the concepts or numbers that have vague boundaries. Fuzzy sets theory defines such concepts and numbers as fuzzy sets. Operations of fuzzy sets are the formal mechanism to operate on fuzzy sets to define new concepts. The theory facilitates modeling of situations that are observed as approximate or vague; most often a fuzzy set represents a concept associated with the natural language. Therefore, this theory is useful in analyzing qualitative or descriptive information and modeling a system whose properties are known or expressed only in natural language. There are many areas in which fuzzy set theory can be applied; they include inference, control, classification, decision making, and optimization.

Since introduced by Zadeh, more than a quarter century ago, fuzzy sets theory has gained an increasing level of acceptance in science and engineering. Gupta (1977) as quoted in *Fundamentals of Fuzzy Sets* (Dubois and Prade, 2000, page 25) defines fuzzy sets theory as "a body of concept and techniques aimed at providing a systematic framework for dealing with the vagueness and imprecision inherent to human thought process." Because the subjects in transportation engineering and planning cannot be separated from human perception and decision processes, transportation problems are a good domain for fuzzy theory application. In fact, a sizeable number of attempts have already been made to apply fuzzy set theory to transportation problems.

This part presents the following topics with the purpose of providing an introduction to application of fuzzy sets theory to transportation problems. First, we examine the nature of the transportation problems and identify the need for a new paradigm to deal with complex problems that we face today. Second, we introduce the basics of fuzzy sets theory. Third, we explore the possible general application areas for fuzzy sets theory, as well as specific transportation applications.

THE NATURE OF TRANSPORTATION PROBLEMS

While the situations in which transportation analysis take place may be characterized in many different ways, in the following we characterize them in the context of four basic elements of systems analysis, which are input, model or knowledge base, output, and goals and objectives.

Input

Information used for making projection and decisions in transportation is usually imprecise. Accuracy of input is often inconsistent among different data sets, and the lack of data is often supplemented by interpolation and extrapolation based on the available coarse data. Data pertaining to perception and feeling are difficult to handle because the boundaries are vague and unclear. Traditionally such data are treated as a single value or a rigid interval. Examples of information and data that have vague boundaries include the following:

- Notion of desire, goal, and target (e.g., desired cost, desired time-saving, desired arrival time);
- Notion of satisfaction and acceptability level (e.g., satisfactory level of achievement, acceptable air pollution level, acceptable cost, acceptable delay, acceptable error, willingness to pay);
- Perception and quantities based on memory (e.g., time spent for an activity, distance traveled, prices paid);
- Description of perceived condition or quality (traffic congestion, comfort level, acceptable safety level);
- Imprecise and hard to measure quantities (e.g., sight distance, reaction time, value of time, capacity of roadway, adjustment factors in highway capacity calculation);
- Performance as a combination of attributes that are interacting one another (LOS of highway or transit, aesthetic quality, concept of livability); and
- Cushion value—a padded value to absorb uncertainty (safety factor).

Model or Knowledge Base

The knowledge about causalities or relations is generally not very clear in the case of human and societal affairs, and it is expressed often in languages rather than in a precise equation and formula; although, often, such human phenomena have been modeled in a rigid mathematical formula in social science. Further, the domain to which a particular knowledge-base applies is not clear. Traditional rule-based models, such as expert systems, have a clearly defined domain in which each rule applies. A typical example is the relationship between different socioeconomic characteristics in an area and the number of trips generated from the area (cross-classification-based trip generation approach) in a cross-classification form. In general, a transportation system is a complex system in which many elements are interacting in a complicated manner so that the input is affected by the output (i.e., feedback loops); hence, it is not possible to reproduce the phenomena precisely and also to generalize the phenomena in a precise format.

Output

If either (or both) input or (and) the knowledge base is vague, then the output inevitably becomes vague. Such an output is the reflection of the propagation of uncertainty. In reality, however, the uncertainty in the output is often presented with a mask of certainty. This is typically seen in the forecasting situation. During the analysis process, uncertainty must propagate, but the initial uncertainty in the input and that in the knowledge base mysteriously disappears along the way, and, as a result, the output gives an illusion of certainty. In fact, presenting uncertainty would show more credibility in the analysis.

Traditionally, testing the effects of uncertainty has been dealt with by the sensitivity analysis, in which the range of input values is used to generate a range of possible output. This approach is certainly better than the single input and single output approach, but still, it does not present the range of outcome with the degree of possibilities. What we need is an output that is consistent with the degree of uncertainty of the input and knowledge base. This is not easy to perform in the traditional framework of uncertainty, particularly when language-based information is involved.

Goals

The goals of the decision or control problem are often not clearly defined in transportation; usually they are stated qualitatively, such as to reduce congestion and to improve air quality. Therefore, whether the outcome of an action satisfies the goals or not is not clearly determined. Further, transportation plans intend to achieve many objectives, but the priorities and weights among them are not certain.

The compounding effects of these characteristics lead to a conclusion of an analysis whose validity requires many qualifying statements. In summary, difficulty of transportation analysis is related largely to handling and interpreting uncertainty in (*a*) the data, (*b*) model or knowledge basis, (*c*) output, and (*d*) goals (or what we want to achieve); further, how to present the uncertainty in a credible manner is another important topic, because in the end the analysis is presented to the decision makers who generally do not like to face uncertainty.

VOID IN THE CLASSICAL APPROACH

Traditionally, probability theory has been the only avenue to analyze the uncertain situations. It is used to obtain the degree of likelihood that a particular event occurs. In this approach, the probability distribution function represents the information (in frequencies) that each of possible events occurs, and this distribution becomes the evidence for testing the hypotheses. Many problems of transportation have been dealt with by probability theory. A typical example is the application of queuing theory, in which the number of service channels is determined in order to limit the probability that the arriving units are delayed more than a certain tolerable time, or the probability that the length of the queue becomes more than a tolerable value. Another typical application of probability theory is the random utility models. In this theory, one chooses an alternative based on comparison of utility associated with each alternative. Utility is measured by the attributes of choice and elements of random terms, which accounts for all the unknown factors related to calculation of utility. This expression of the unknown factor has been the focus

of debate for many years. Depending on what probability distribution is assumed for this random term, either normal or Gumble distribution, the stochastic choice model has been grouped into probit and logit models, respectively. The question has been whether all unknown situations can be treated by a probability distribution. Probability theory is the correct approach when one is dealing with the random events in which all the possible outcomes are accounted for clearly and the phenomena are "observable and repeatable." However, its use for all types of unknown situations is a point of debate.

Consider the situation of tossing a die. If the outcome of each toss of a die is clearly observable, a value between 1 and 6, then the frequency distribution of the outcome provides the information (or evidence) about the system (in this case, about the die's property). Given this evidence, one assesses the degree of truth that the outcome of the next throw will be a particular value in probability.

What if the outcome is not clearly observable; say, if the toss was conducted under the feeble light, and one could observe the outcome only vaguely, "maybe 1 or 2," "a large number," "a small number," etc. In this case, the probability distribution cannot be established easily, and the axiom of probability measure, the sum of the probabilities of all events is equal to unity, is not established. Even if the observation is clear, what if the hypothesis is, "the outcome is a 'large' number"? In this case, without the clear definition of a "large number," one cannot obtain the probability. In other words, vagueness in observation or proposition can disable the use of probability theory; when the language, such as "large" or "small," is used to describe the outcome of the trial.

Given the nature of transportation problem as described in the previous section, the important issue is how to formalize the uncertainty associated with the linguistic expression in the mathematical structure so that the observation and knowledge can be processed systematically and put in the form of a mathematical model. Such a scheme would allow the computer to simulate the situation and also replicate the human control and inference processes. Particularly useful would be the ability to model the human decision and control processes that are based on language, like driving a car, making a choice of path in a crowed pedestrian environment, diagnosing the cause of the environmental damage, and automating these processes.

To discover and describe underlying relationships and to apply them for prediction, diagnosis, and control has always been at the core of the engineering profession. The relationships are normally formalized in the form of y=f(x). However, in human control and inference process, relationships are usually expressed in language in the form of "if… x..., then..y..," like, "if the vehicle in front decelerates fast, then decelerate fast," or "if traffic is congested, then leave home early." In these situations, the issue is not simply whether the relationship between x and y does exist or not exit, but the strength of association between particular x and y values is important. Often in a description of a relationship, one says, "somewhat related" or "very much related." This is usually the kind of description of relationships that are involved in the transportation analyses.

If, for example, we formalize a police officer's manual traffic control rules, which are based on his experience, at an intersection in a mathematical form, we may be able to develop a traffic control that emulates the police officer's control. Similarly, if we can put the driving rules in mathematical form, we may be able to automate some of the driving tasks. Further, a complex transportation system's workings, the relationships between supply and demand that are expressed in natural language, are put into a more formal expression, then predicting and diagnosing the situations may be systematized.

In summary, the analysis of transportation engineering and planning needs a mechanism that allows for a formalized treatment of vagueness in human expression and human-based information. Fuzzy sets theory is a paradigm that expresses linguistic uncertainty by the mathematical formalism. This theory helps to establish mathematical integrity in the analysis process, but also creates new avenues for control and inference.

FUZZY SET THEORY AND MEMBERSHIP FUNCTIONS

A set is a collection of elements (or items) that has some common characteristics and functions, reference, or features. For example, different parts of a car can form a set which may be called a set of car parts, or the alternatives for a decision may be a set called the decision alternatives set. A set is the basic building block of mathematics, and it is also the building block of the thought development and process.

Crisp Set

The population of a country can be divided into two groups, those who are 65 and older, and those younger than 65. These subsets are called the crisp set, meaning that the boundary of the set is crisply definable. Formally, the elements of set A is characterized by the characteristic function, which assigns a value, either 0 or 1, to each element, 0 being not belonging to the set, and 1 belonging to the set.

Crisp set: $C_A(x) = 0$, if x is not included in A; $C_A(x) = 1$, if x is included in A.

Figure 14 shows the characteristic function for set A.

Fuzzy Sets

The natural language based expression, such as “high speed” or approximately 100”, cannot be clearly defined for their boundaries, because we cannot easily tell whether a particular speed, say 60 mph, is “high speed,” or a value, say 120, is within the “approximately 100,” without knowing the situation or context in which they are discussed. Actually, on the city streets, 60 mph is certainly a high speed, but this may not be the case on a rural road.

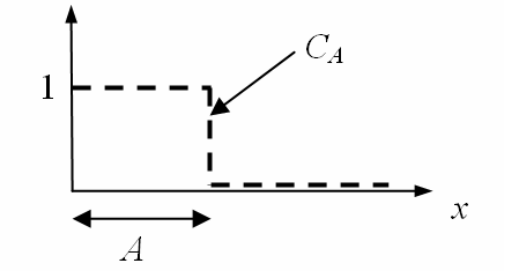

FIGURE 14 Characteristic function for set *A*.

A fuzzy set is a set whose boundary is fuzzy so that one cannot easily state whether the element matches the notion of the set or not because the transition is gradual. Nearly all natural language based expressions, e.g., "late arrival," "heavy traffic volume," and "long distance," can be considered as fuzzy sets with their elements being time, traffic volume, and distance, respectively.

The counterpart of the characteristic function in fuzzy sets theory is called the membership function. This is a function that maps the degree of compatibility between the element and the notion that the set A represents. It has the following form.

Fuzzy set A: $A(x)$: $X \rightarrow [0,1]$

where $A(x)$ is the membership function, which assigns an element x of the universal set X into a value between 0 and 1. The value of $A(x)$ is called the grade of membership of x in set A; this indicates the degree of association, compatibility, and closeness to the notion of A.

The membership function of a fuzzy set, "approximately 15," $A(x)$ can be in the following form:

Continuous case: $A(x) = 3e^{(x-15)^2}$
Discrete case: $A(x) = 0.5/12 + 0.7/13 + 0.8/14 + 1/15 + 0.9/16 + 0.6/17 + 0.3/18$

where a/b, a shows the membership grade, and b is the element of the set.

The shapes of these membership functions are shown in Figure 15.

A fuzzy set does not have to be a set of numbers, as shown above. It may be a set of acceptable alternatives in a decision set, in which the elements are decision alternatives, A, B, C, and D as shown below:

$Z(x) = 0.5/A + 0.4/B + 0.8/C + 0.6/D + 0.7/E$

where A through E are alternatives, and Z can be called the decision set with "acceptable" alternatives. $Z(x)$ is the membership function, or the degree of acceptability of alternative x.

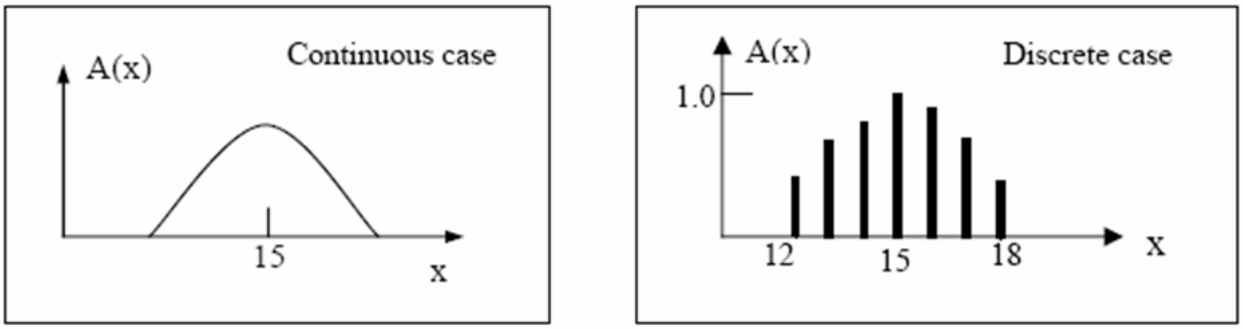

FIGURE 15 Examples of membership functions, continuous, and discrete cases.

APPLICATION ENVIRONMENTS OF FUZZY SETS THEORY

This section discusses the environment in which fuzzy sets theory is the suitable approach for analysis. However, before we begin explaining application of fuzzy set theory to various transportation problems, we must understand the domains in which fuzzy set theory is useful and also in which it is not appropriate.

The Problem Domain Where Fuzzy Set Theory Is Applicable

Generally, fuzzy sets theory is applicable to situations in which uncertainty is related to perception and interpretation due to the lack of the clear definition or boundaries. Consider the following. When one goes to an ophthalmologist for a visual acuity check up, one is asked to "read" letters on the Schellen chart from a distance. At a certain band of distance, one usually encounters difficulty in answering whether a particular letter is "readable" or not. Often, one takes time to decipher the letter. In this situation, the distance from which a particular letter is readable is fuzzy because there is no exact distance at which one can say "I can read it" and beyond which "I cannot read it." Dubois and Prade (1991) characterizes fuzziness by the situation when one takes time to answer yes or no to a question. If one plots the degree of clarity at each distance from the chart, then the plot could constitute a membership function for a set of "readable" distance.

Let us again think about the case of the Schellen chart above. When the experiments are performed for a number of people and each person specifically states yes or no (although this may be difficult for each), and the statistics with regard to the distance from which the test subjects can or cannot read the letter is compiled, then this information is the frequency distribution of the distances from which different persons can read the chart. The distribution, although it may be similar to the shape of the membership function, is not a fuzzy quantity; it is the statistical information based on the yes or no responses by the subjects.

On the other hand, when one is presented with a crisp number of travel time, say 78 min, and this number may be based on the average value obtained from a frequency distribution of travel times. Most people consider the travel time as approximately 80 min, then, this rounded number is a fuzzy number because it is no longer the statistical value, but it is a convenient number that represents the sense of approximation. This 80 min is usually used to make the decision for the departure time or for normal conversation, meaning it is "possible" to travel in 80 min.

It is important to separate the situations in which the classical statistical treatment is needed and that in which fuzzy set theory is appropriate. Probability density function and the membership function of fuzzy set are fundamentally different. The probability density function represents a summary of the information obtained from random experiments, and it is used to measure the truth of a proposition in probability. On the other hand, the membership function represents the definition of a set that represents a person's understanding, interpretation, feeling or disposition regarding the notion of a problem at hand, most often related to language, naturally, a subjective quantity. As a result, unlike the probability density distribution, the values of the membership functions are operated in the fuzzy set operations.

Fuzzy Inference

For most daily activities, human behavior and decision is structured on the language-based inference system. For example, when one sees a long queue of vehicles on the highway, one would infer a traffic incident ahead. If the queue is very long, then one would infer a very large incident. If it is a short queue, one infers a relatively small incident.

In this case, one perceives the queue length ahead, and also one has the general knowledge regarding the relationship between the queue length and the severity of traffic incident. Then, one builds a knowledge-base rule, which associates the queue length and the severity of accident. Then, if one has the information on the current queue length, one can infer the severity of the incident.

The most fundamental structure of an inference is the following type:

Premise 1: x is A
Premise 2: If x is A, then y is B
Consequent: y is B

In this form of logic, if the two premises (Premises 1 and 2) are true, then the consequent is always true. This logical structure has been the foundation of reasoning from the time of Aristotle, and it is called modus ponens.

This structure is generalized into a generalized modus ponens, in which A', A, and B are expressed in fuzzy terms:

Premise 1: x is A' (fuzzy input)
Premise 2: if X is A then Y is B (fuzzy relation)
Consequent: y is B' (fuzzy output)

While A' and A are not the same fuzzy set, but they are drawn from the same universal set. The degree of match between A and A' defines the degree of validity of the consequent "y is B'." Premise 1 is a fuzzy set of input (or data), and Premise 2 is a fuzzy relation expressed by the form shown above.

Defuzzification is a step that defuzzifies the consequent. A situation may arise such that one needs to reduce the fuzzy outcome into a single value representation; after all a decision is binary. For this, we need a process to convert the fuzzy output into a single value. Several approaches are proposed for this step. The most popular approach is to take the center of gravity of the shape of the membership function of B' (z) with respect to z.

The modeling situation described above is called the fuzzy system. In a fuzzy system, input is fuzzy and the behavior of the system is known only fuzzy. The output becomes inevitably fuzzy. The description above is summarized in an illustrated form in Figure 16.

The process is useful for many situations of transportation analysis, in which the causalities are represented by the linguistic rule basis, and the observed data is fuzzy. A typical example is the human choice process.

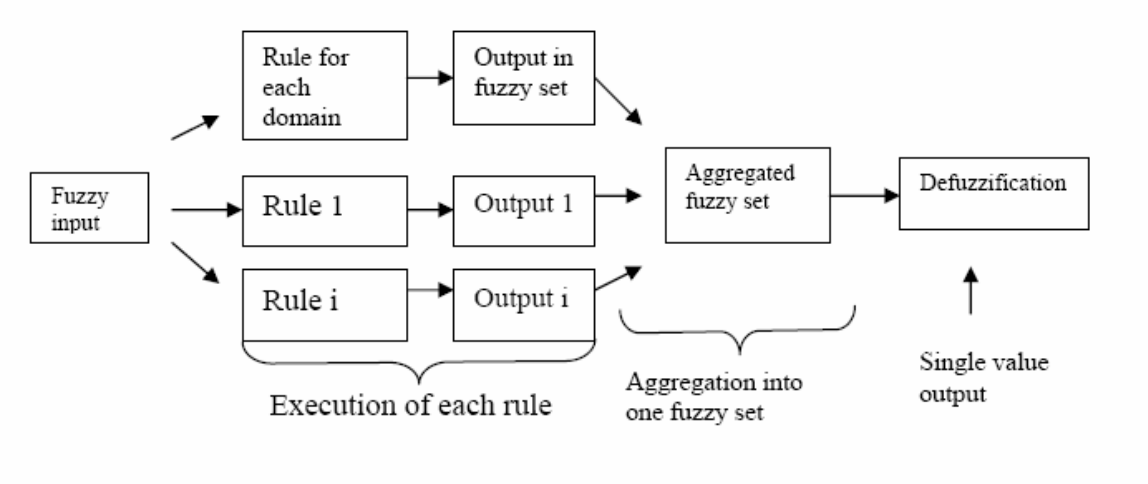

FIGURE 16 Fuzzy system input and output process.

Fuzzy Controls

Most human-based control problems, such as adjusting the hot water and the cold water faucet when taking a shower, adjusting the steering angle driving a car, and adjusting design parameters when remodeling a kitchen, are in essence fuzzy controls. There are no specific rigid formulas for control, but humans learn to exercise control by experience, and over time developed a set of rules. Cooking is another example of human control problem—fuzzy measurement of ingredients, mixing, and timing.

Mathematically, the operation of the fuzzy sets for control is the same as what was demonstrated for fuzzy inference above, but an important difference is the derivation of a single value. In other words, in the case of control, usually, the parameter value for control must be specified by a single value; thus, it is a normal practice that output from the fuzzy inference is defuzzified to a single value. Defuzzification is performed by one of several methods, among them the most popular and intuitive approach is the center of gravity, which is the value corresponds to the center of gravity of the shape defined by the membership function.

Fuzzy Optimization

The Concept

The traditional optimization processes finds the values of the decision variables so as to minimize or maximize the objective function, within the domain of values of variables that are defined by a set of constraints. Both the objective and the constraints are a function of the decision variables. The formulation requires the strict satisfaction of the constraints; thus, sometimes, the optimum value may not be found.

In the fuzzy optimization, the process is to find the values of the decision variable so as to achieve the maximum "satisfaction" or "compatibility" with the objective and the constraints. The degree of satisfaction with the objective and constraints is measured by the membership

grade between 0 and 1. The approach is based on the idea that the optimum solution is the *best* compromise between the objective and constraints.

In this formulation, the mathematical process is to find the set of parameter values in the intersection of the fuzzy sets of objectives and constraints. This approach follows the Bellman–Zadeh principle of optimization [Bellman and Zadeh (1970)], which states that the optimum solution lies in the confluence of objective and the constraints. Thus, both the objective and the constraints perform the same purpose of defining the solution sets, and the notions of objective and constraint are interchangeable.

Consider the following optimization problem. In the process of negotiating the salary, the labor wants the salary as high as possible, and the management wants the salary to be as small as possible. Say the labor wants the hourly salary to be greater than \$18/h, and the management wants to contain the salary to be less than \$14/h. Then, there is no window to negotiate. What usually happens, however, is that an arbitrator offers a compromise from both the labor and the management: say, an "acceptable" minimum salary to be somewhat greater than \$16/h from the labor, and "acceptable" maximum salary to be somewhat less than \$16/h from the management. By drawing the membership function of the "acceptable," along the axis of the hourly pay, the intersection of the two sets defines the area of feasible solution. In the feasible solution, the value that maximizes the acceptability to both the management and the labor is found. This is the point where the minimum satisfaction is maximized, max–min. This is illustrated in Figure 17.

In the classical optimization approach, the solution may become infeasible (Figure 17*a*); in real life, however, through a compromise, a solution is usually found. This is a result of fuzzifying the boundaries of the objective and the constraints; that is compromising. Such fuzzification of boundaries is seen in Figure 17*b*. The compromised solution is found the following.

$$D(x^*) = \text{Max Min } \{G(x), C(x)\} \text{ for all } x$$

where x is the decision variable; x^* is the optimum solution, $G(x)$ and $C(x)$ are respectively, the membership function of the objective and the constraint. $D(x^*)$ represents the maximum degree of satisfaction of both the objective and constraint. Actually, we now see that there is no difference between objective and constraints as far as the computation is concerned.

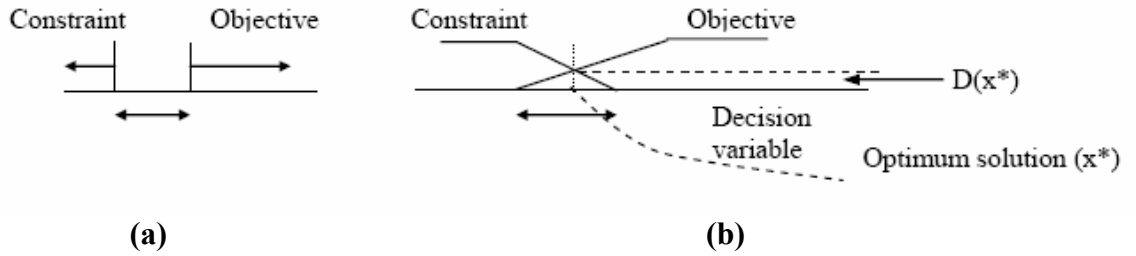

FIGURE 17 Illustration of fuzzy optimization concept: (*a*) no feasible solution and (*b*) feasible solution.

Because the intersection can be formed by many objectives and constraints, this formulation solves the multiobjective and multiconstraint programs, in which the solution is found in the intersection of the objectives and constraints, by maximizing the value of the membership grade of intersecting membership functions. Thus, the generalized expression of the optimum solution is the following.

$$D(x^*) = \text{Max Min } \{G_1(x), G_2(x), C_1(x), C_2(x), C_3(x)\}$$

In this formulation, if the membership functions are in linear form as shown in Figure 17*b*, then the problem can be solved using linear programming. This is shown in the next section.

It should be noted that *x* does not mean to be a single variable. It can be a vector of decision variables, *X*. If the decision set's elements are discrete then the optimization concept is depicted as shown in Figure 18, where each dot indicates the degree of satisfaction of an alternative for each objective and constraint. Among the alternatives (*X*s) the one whose minimum degree of satisfaction is maximum is chosen as the optimum solution; in this case, alternative 3.

The fuzzy optimization technique discussed above is suited for problems in which both the objective and constraints are not clearly defined or flexible; for example, when the objective is to control cost "around" a certain value to meet the budget level, or when the constraints are not known exactly and a certain level of tolerance is acceptable. Therefore, this approach can be most useful for the strategic planning level. The trade-off between the costs of obtaining precise information to find the optimum versus the cost of obtaining less accurate data to find somewhat less optimum solution must be evaluated before using fuzzy optimization. In some problem, the objectives and constraints are inherently vague; in this case, a fuzzy optimization approach is suitable. In some cases, the constraints may be a mix of fuzzy and crisp equations and inequalities.

APPLICATION PROBLEMS

In the following, we present a set of subjects in transportation to which fuzzy sets theory can be considered as a viable approach. What is common to these situations is the involvement of human interpretation, rather than completely mechanical or physical phenomena. The problem on hand should have the following features.

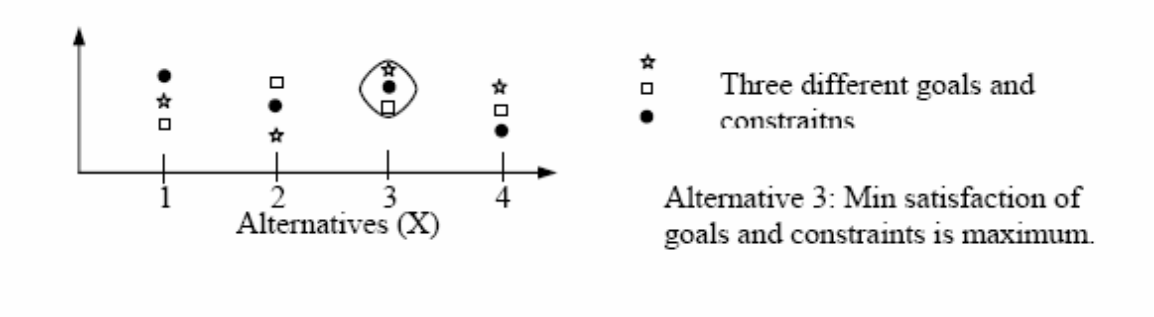

FIGURE 18 Illustration of discrete fuzzy optimization.

- Fuzziness or approximation in the data values;
- Fuzziness in the knowledge base, relations, and rules;
- Fuzziness in the reference (e.g., goals and desire); and
- Fuzziness in the model outcome is acceptable.

Data Handling Problems

The data used for transportation analysis are basically two types: numerical data and language-based data. The nature of the numerical data may be the observed or measured values, perceived values, and the values set for reference, such as limiting value for safety or regulatory reasons. The language-based data are words, such as "large," "small," etc., whose meaning depends on the context.

Fuzzy set theory is useful for dealing with the data that is approximate and also the language-based data in the problems, to show how the uncertainty associated with the individual values propagate as they are operated.

Arithmetic Operations of Fuzzy and Crisp Values

This is useful for estimating the cost of a large-scale project when individual component costs are known only in approximate terms and the relationships that connect the cost parameters are not well known. It is also applicable for estimating travel time between two points, when the travel times for different segments between the points are known approximately.

Line Fitting Problem

Line fitting problem to fuzzy data, in particular, when both the x–y values are fuzzy. For example, when a set of data points are approximate and with a range (both x and y values), and when a line that fits all the data points is needed.

Adjustment of Data for Consistency

Examples of applications include: (*a*) making the consistent traffic volume counts from the inconsistent observed counts on the transportation network; (*b*) making consistent transit ridership counts from the inconsistent counts, e.g., nonequal boarding and alighting sums; and (*c*) adjusting the desired design values while making sure the design values to confirm to a set of pre-established relationship.

Optimization Methods

The concept of fuzzy optimization has been applied to the traditional optimization algorithm, such as fuzzy linear programming transportation problem and fuzzy dynamic programming. In the case of fuzzy linear programming transportation problem, supply and demand are known only in fuzzy numbers and it determines the amount to be shipped between each demand–supply node pair (x_{ij}). Mathematically it is solved according to the following.

Max h

Subject to: $G(x_{ij}) \le h$, $Si(x_{ij}) \le h$, and $Dj(x_{ij}) \le h$, $\sum_j x_{ij} \approx Si$, $\sum_i x_{ij} \approx D_j$

where $G(x_{ij})$ represents the goal, such as the minimum total cost or time of shipment, $Si(x_{ij})$ and $Di(x_{ij})$ represents the approximate total supply and demand at i and j, respectively. It is possible to introduce additional either fuzzy or rigid constraints.

- Fuzzy linear programming for resource allocation at the strategic planning level. This is applicable when the objectives and constraints to an optimization problem are fuzzy.
- Fuzzy dynamic programming for network optimum path analysis, sequential decision-making process. In this case, the cost function is not exact but fuzzy.
- Compromise among different interest groups during planning stage, e.g., compromise of desires of different groups. One group's desire is a constraint to another group.

Reasoning and Inference

Examples include

- Traffic signal timing and phasing controls based on fuzzy data and fuzzy rules;
- Dispatching control of public transportation, air traffic control, land use controls, and traffic flow controls;
- Rule-based decision problem, e.g., choice of travel modes, choice of routes, and choice of departure time; and
- Representation of a large-scale cause–effect relationship and measurement of the justification of the process for alternative investment analysis, traffic impact analysis.

Fuzzy Systems

Application examples include

- Large-scale transportation planning and investment modeling as a fuzzy system;
- Fuzzy input;
- Fuzzy knowledge base–fuzzy output structure for travel demand forecasting; and
- Diagnostic analysis, including accident analysis, evaluation of the cause of environmental damage, and causes of traffic congestion.

Comparison of Fuzzy Numbers

This includes

- Comparing two approximate numbers for ranking and ordering;
- Evaluation of driver perception of safe separation of vehicles and stopping distance;
- Evaluation of vehicle design, passenger comfort, and safety; and
- Highway capacity analysis and LOS determination, e.g., comparing the passenger car equivalent volume with the LOS demarcation.

CONCLUSIONS

The essence of fuzzy set theory lies in the following.

1. Its ability to model natural and human-related phenomena, in which no clear demarcation from one state to the next exists. In particular, it is suited when the data and the relationships are approximate or expressed in language, and the approximate output is sufficient. With the use of proper set operators, the set operations can preserve uncertainty in the computation process and reaches the output whose uncertainty is consistent with that of input and the relationships.
2. Its ability to incorporate the traditional mathematical optimization. Many of the traditional operations research techniques can be modeled with fuzzy sets theory, e.g., fuzzy linear programming or fuzzy dynamic programming. For these applications, the theory can deal with approximate goals and constraints.
3. Its ability to evaluate the truth of a proposition using fuzzy data. It can be used for evaluation of alternatives when there is more than one goal, and the goals are defined in language. This ability complements the classical probability theory, because the uncertainty involved is not random nature but rather perceptive.

It should be noted that the theory does not explain the reason why the system is fuzzy; rather it describes the phenomena. The analyst must examine rigorously if in fact the phenomena being analyzed are fuzzy phenomena. It should be warned that fuzzy theory should be used only when the merits exist. For many transportation problems, the traditional deterministic assumptions or probabilistic approach may suffice. Probability theory is the legitimate approach when the phenomena are random and repeatable, such as the physical phenomena of vehicle arrivals. It should be emphasized, however, that fuzzy sets theory has no one-on-one match correspondence with probability theory. Therefore, fuzzy theory should be applied only when the benefits exist and the phenomena being analyzed meets the definition of fuzziness.

Terms such as fuzziness, vagueness, and ambiguity have not been considered as the "desirable" features in the traditional analytical approach. The tendency has been to eliminate such uncertainty by assumptions. Definitive crisp statements and logic have been preferred to "fuzzy" reasoning for many years. Such a mindset, however, has not been effective in modeling when the subject involves human judgment and human decision process. Fuzzy sets theory offers a legitimate mathematical approach that complements the traditional approach in transportation engineering and planning by preserving uncertainty in the analysis process, which is abundant in this field.

We conclude this section by saying

> Language changes based on what the society gives the meaning. Uncertainty and fuzziness is not the devil, it is a positive force to sustain the society. To make life interesting two basic elements are randomness and fuzziness. Those are indispensable unpredictability and the language cannot be singly interpreted. We are now accepting the fact that we need randomness and fuzziness.
>
> —Radhakrishna Rao
> *Statistic and Truth: Putting Chances to Work* (2nd Edition), 1989

REFERENCES

Bellman, R., and L. A. Zadeh. Decision Making in a Fuzzy Environment. *Management Science*, Vol. 17, 1970, pp. 141–164.

Berkan, R. C., and S. L. Trubatch. *Fuzzy Systems Design Principles: Building Fuzzy IF_THEN Rule Bases*. IEEE Press, Piscataway, N.J., 1996.

Blanchard, D. *Intelligent Systems: Applications and Analysis*. 1994–1995 Edition, Order 2555. Lionheart Publishing, Inc., Atlanta, Ga.

Chin-Teng, L., and G. C. S. Lee. *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*. Prentice Hall, Upper Saddle River, N.J., 1995.

Cox, E. *The Fuzzy Systems Handbook: A Practical Guide to Building, Using, and Maintaining Fuzzy Systems*. ISBN 0-12-194270-8. AP Professional, Cambridge, Mass.

Delgado, M., J. Kacprzyk, and M. A. Verdegay. *Fuzzy Optimization*. Physica-Verlag, c/o Springer-Verlag, GmbH $ Co. KG, Auftragsbearbeitung, Postach 31 13 40

DuBois, D., and H. Prade. *Fundamentals of Fuzzy Sets: The Handbooks of Fuzzy Sets Series*. Kluwer Academic Publishers, Norwell, Mass., 2000.

Dumitrescu, D., B. Lazzerini, and L. C. Jain. *Fuzzy Sets and Their Application to Clustering and Training*. CRC Press, Boca Raton, Fla., 2000.

Kandel, A., and G. Langholz. *Fuzzy Control Systems*, CRC Press, 1993.

Kandel, A. *Fuzzy Expert Systems.* CRC Press, Boca Raton, Fla., 1991.

Kandel, A. *Fuzzy Mathematical Techniques with Applications*. Addison–Wesley Publishing Company, 1986.

Kaufmann, A., and M. M. Gupta. *Introduction to Fuzzy Arithmetic Theory and Applications.* Van Nostrand Reinhold, 1984.

Klir, G. J., U. H. St. Clair, and B. Yuan. *Fuzzy Set Theory: Foundations and Applications.* Prentice Hall, Upper Saddle River, N.J., 1997.

Klir, G., and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall, Upper Saddle River, N.J., 1995.

Klir, G. J., and T. A. Folger. *Fuzzy Sets: Uncertainty and Information*. Prentice Hall, Upper Saddle River, N.J., 1988.

Klir, G. J., and M. Wierman. *Uncertainty-Based Information: Elements of Generalized Information Theory*. Physica-Verlag, a branch of Springer-Verlag (ISBN number:3-7908-1242-0).

Kosko, B. *Neural Networks and Fuzzy Systems*. Prentice Hall, Upper Saddle River, N.J., 1992.

Li, H.-X., and V. C. Yen. *Fuzzy Sets and Fuzzy Decision Making*. CRC Press, Boca Raton, Fla., 1995.

Mamdani, E. H., and B. R. Gains. *Fuzzy Reasoning and Its Applications*. Academic Press, 1981 and reprint 1987.

Mares, M. *Computation over Fuzzy Quantities.* CRC Press, Boca Raton, Fla., 1994.

McNeill, M. F., and E. Thro. *Fuzzy Logic: A Practical Approach.* AP Professional, Cambridge, Mass., 1994.

Orlovski, S. *Calculus of Decomposable Properties, Fuzzy Sets, and Decisions*. Allerton Press, Inc., New York.

Wang, P. P. Advances in Fuzzy Theory and Technology Vol. 1. Bookwrights, Fairfax, Va.

Pedrycz, W. *Fuzzy Sets Engineeiring*. CRC Press, Boca Raton, Fla.

Wang, P.-Z., and K. F. Loe. Between Mind and Computer: Fuzzy Science and Engineering. *World Scientific*, 1994.

Reghis, M., and E. Roventa. *Classical and Fuzzy Concepts in Mathematical Logic and Applications.* CRC Press, Boca Raton, Fla., 1998.

Sugeno, M. *Industrial Applications of Fuzzy Control.* North-Holland, 1985.

Terano, T., K. Asai, and M. Sugeno. *Applied Fuzzy Systems.* AP Professional, Cambridge, Mass.

Wang, Z., and G. J. Klir. *Fuzzy Measure Theory*. Plenum Press, New York, 1992.

Wang, Z., and G. J. Klir. *Fuzzy Measure Theory*, Plenum Press, New York.

Welstead, S. T. *Neural Network and Fuzzy Logic Applications in C/C++*. Wiley and Sons, Inc.
Yager, R. R., and D. P. Filev. *Essentials of Fuzzy Modeling and Control.* John Wiley and Sons, Inc. 1994.
Yager, R. R. and D. P. Filev. *Essentials of Fuzzy Modeling and Control*. John Wiley and Sons, New York, 1994.
Zadeh, L., and J. Kacprzyk. *Fuzzy Logic for the Management of Uncertainty*. John Wiley and Sons, New York.
Zimmermann, H.-J. *Fuzzy Set Theory and Its Applications*, Third Edition. Kluwer Academic Publishers, 1996.

Genetic Algorithms

GHASSAN ABU-LEBDEH
Michigan State University

Nature-inspired computational paradigms are becoming more common in an increasing number of applications. While none of those paradigms work magic, they have uniqueness and power that make them a first choice in many application domains including transportation. This part of the circular aims at providing a one-stop overview of one such group of algorithms: GAs.

GAs are part of a broader class of evolution-inspired algorithms (Figure 19). Alternatively, one can classify them as one of a derivative-free optimization group of algorithms. Their applications in transportation go back to the early 1990s and since then have been growing steadily. In all applications, GAs were instrumental in solving problems that had been either difficult to solve, or the solutions had been of modest quality.

This part of the circular serves two broad objectives: (*a*) provide a brief introduction to GAs that covers their fundamental components and mechanisms, and a brief account of GA applications in transportation, and (*b*) provide a concise coverage of means to improve performance of GAs with special emphasis on parallel GAs. The first section is a brief overview of GAs including coverage of the primary components and functional steps, along with a brief note on why GAs work well. Section 2 discusses the general characteristics of problems for which GAs are particularly suited, presents a summary of GA applications in transportation, and concludes with a discussion of the general pitfalls users should be aware of. Means to improve GA performances and discussion of some advanced topics are the subject of section 3. Section 4 is on parallel GAs.

BRIEF OVERVIEW OF GENETIC ALGORITHMS

GAs are one of the derivative-free stochastic optimization methods which have their foundation in the concepts of natural selection and evolutionary processes. While these two functional characteristics summarize the essence of GAs, more details are necessary to fully understand how GAs function and the secret to their strength. GAs use the mechanics of natural selection and natural genetics. The basic operation of a GA is simple. First a population of possible solutions to a problem is developed. Next, the better solutions are recombined with each other to form some new solutions. Finally, the new solutions are used to replace the poorer of the original solutions and the process is repeated. Here, GAs will be introduced through an example where the fundamentals and different functionalities are noted. This way, the reader will be able to follow the logic and the specific steps taken to put a GA to use for a given optimization problem. Later, we will discuss GA properties, how they function, what makes them unique, and what the sources of their strength are. Limitations of GAs will also be discussed.

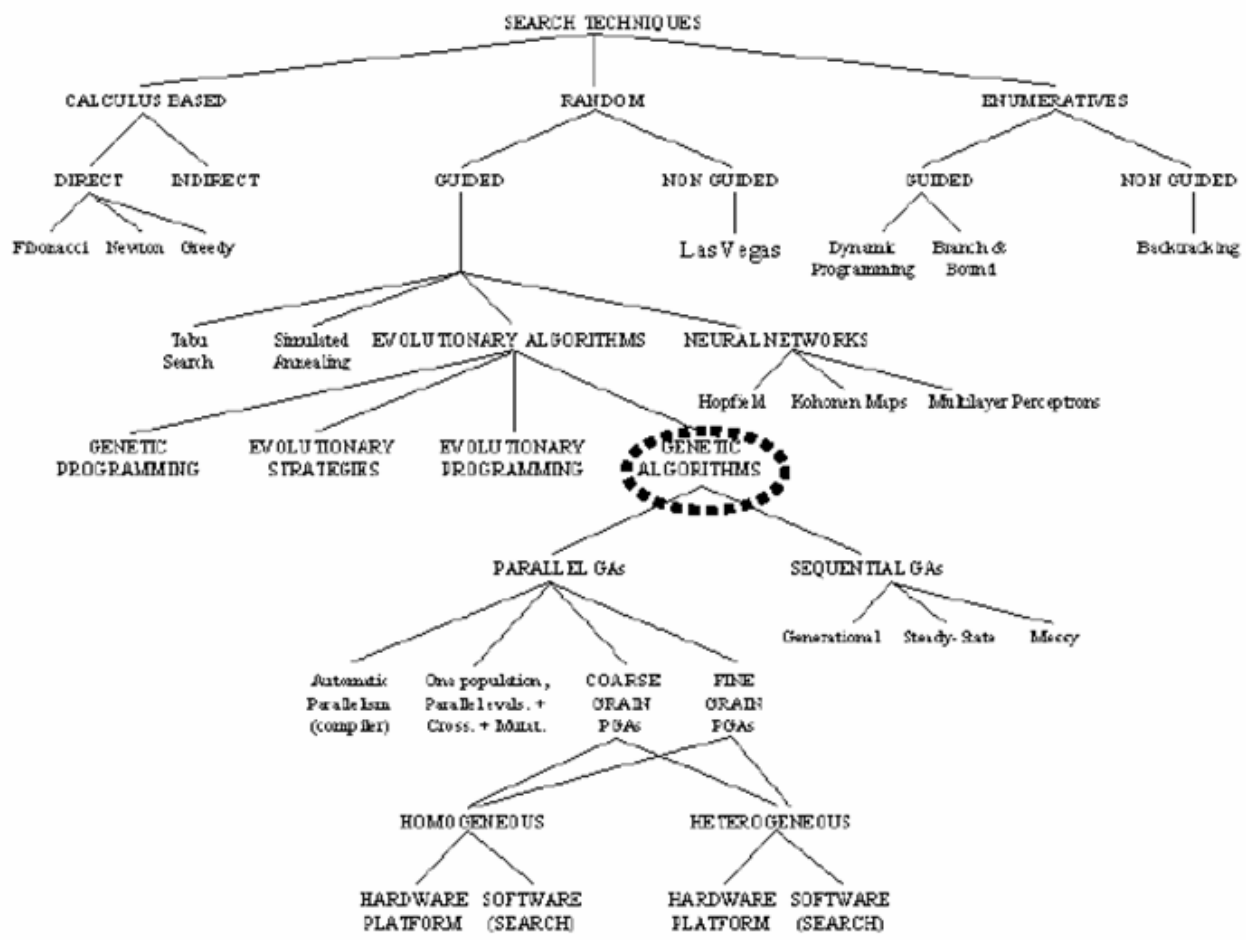

FIGURE 19 GAs among family of search algorithms.

How Does a GA Work?

The following are the main functional steps that take place in a GA: (*a*) selection, (*b*) cross-over (or recombination), and (*c*) mutation. At the end of this sequence of steps the population is ready for the next evaluation (which is done using the fitness or evaluation function). In some respect, evaluation may be though of as a fourth step. Here is a pseudo code for a generic GA. *Pop* is the population and *t* is time measured by number of generations:

```
Begin
t = 0
Initialize Pop(t)
Evaluate Pop(t)
While (t < maximum # of generations) do
        Begin
                t = t +1
```

```
                Select Pop(t) from Pop(t-1)
                Reproduce Pop(t)                    } Main phenomenon of
                        Crossover                   } evolution takes place
                        Mutate                      } here
                Evaluate Pop(t)
        End
End
```

GA Through an Example[a]

Assume we want to find x in the range [–1 … 2] that maximizes the following function:

$$f(x) = x \cdot \sin(10\pi \cdot x) + 1.0$$

This is a simple problem and the derivative can be found easily then made equal to zero to find the value(s) of x at which the value of the function f is maximum:

$$f' = \sin(10\pi \cdot x) + 10\pi \cdot \cos(10\pi \cdot x) = 0$$

This is equivalent to

$$\tan(10\pi \cdot x) = -10\pi$$

This equation has an infinite number of solutions.

Let us assume that we want to use a GA to optimize the above problem. This will involve six distinct components:

1. Genetic representation;
2. A way to create a pool of chromosomes (individuals, solutions);
3. An evaluation function;
4. Genetic operators to alter the composition of children;
5. Selection procedure; and
6. Values of the different parameters, namely the population size, probabilities of crossover and mutation.

Representation

We will use binary representation (other presentations are possible; more on that later). The length of the chromosome will depend on the precision (how many decimal points) we desire. Use 6 places after the decimal. Note that the domain of x has a length of 3 (–1 to 2). Given the desired precision, the domain length has to be divided into at least 3 × 1,000,000 equal size ranges. Since each bit can have two possibilities, a 0 or a 1, the number of bits needed to represent this is 22 bits ($2^{21} = 2097152 < 3{,}000{,}000 < 2^{22} = 4194304$). Mapping the binary string into real number in [–1, 2] is done in two steps: 1) convert the string from base-2 to base-10. For example, the base-2 string **01011** can be converted into base-10 as follows: $2^4 \times \mathbf{0} + 2^3 \times \mathbf{1} + 2^2 \times \mathbf{0} + 2^1 \times \mathbf{1} + 2^0 \times \mathbf{1} = 11$, and 2) find a corresponding real number x such that

$$x = -1.0 + (\text{base-2 value of the string}) \times \frac{3}{2^{22} - 1}$$

The –1.0 in this equation is the lower boundary of the domain and 3 is the length of the domain. For example, the following chromosome (1000101110110101000111) represents the number 0.637197 since the base-2 value of the string, $(1000101110110101000111)_2 = 2288967$, and $x = -1.0 + 2288967 \times \frac{3}{4194303} = 0.637197$. And, the chromosomes (0000000000000000000000) and (1111111111111111111111) represent –1 and 2.0 (the domain boundaries), respectively.

Initial Population

A GA starts its search from a population of chromosomes (possible solutions) and not just one point. How this initial population is created and how many should be in is largely dependent and the GA literature has some ad-hoc rules for deciding that. A randomly created population is not a bad start unless the user has some prior knowledge as to where within the search space the solution is, in which case the initial population would be created accordingly. Otherwise all bits of all chromosomes would be initialized randomly.

Evaluation Function

The role of the evaluation function is to evaluate and rate the different chromosomes (solutions). If v is the binary vector (chromosome) that corresponds to x, then the evaluation function, EF, is related to the objective function f as follows $EF(v) = f(x)$. Using the above example, $v1$= (1000101110110101000111), $v2$= (0000001110000000010000), and $v3$= (1110000000111111000101) respectively correspond to $x1$= 0.637197, $x2$= –0.958971, and $x3$= 1.627888. EV will rate those chromosomes as follows:

$EF(v1) = f(x1) = 1.586345$
$EF(v2) = f(x2) = 0.078878$
$EF(v3) = f(x3) = 2.250650$

Chromosome $v3$ is the best since its evaluation value is the highest.

Genetic Operators

There are two classical genetic operators: mutation and cross-over. Besides these two, others have also been discussed and used in the GA literature. For binary representation, the mutation operator flips one or more bits from 0 to 1 or from 1 to 0 with a predetermined probability (Pm). For real value representation (not emphasized in this section) mutation takes place by perturbing values by adding some random noise. The crossover operator recombines pieces of chromosomes from different parent chromosomes to form a new “baby” chromosome, or an offspring. What pieces from the parents is decided by what crossover point or points are selected. It is possible to cut a parent chromosome into n pieces. If n is 1 then a chromosome is cut into two pieces and the location of this cut is determined randomly. The selection of one or more

cutting points has profound impact on the GA performance; too many cutting points, for example, disrupt the ability of the GA to piece together the necessary pieces of a potentially good solution. Use the above example and assume that the crossover point was randomly selected to be after the fifth bit of chromosomes (parents) $v2$ and $v3$:

$v2$= (00000|01110000000010000)
$v3$= (11100|00000111111000101)

The two resulting offspring are:

$v2'$ = (00000000001111111000101)
$v3'$ = (11100011100000000010000)

and the values of those offspring are:

$f(v2') = f(-0.99813) = 0.940865$
$f(v3')= f(1.666028) = 2.459245$

Note that the best of those two offsprings is better (more fit) than the best of the parents ($v2$ and $v3$).

Selection

Here the new population is selected. Individuals are selected based on their relative fitness as follows: (*a*) calculate the fitness value of each chromosome (potential solution); (*b*) calculate the total fitness of the population (this is the summation of fitnesses of all chromosome in the population); and (*c*) calculate the portability of a selection for each chromosome. This equals to the ratio of the fitness of the chromosome to that of the total fitness of the population. It is noted here that some chromosomes will be selected more than once and hence will get more copies (these would the more fit ones), and others will not be selected at all hence will die off (these are the less fit ones). The average ones stay even.

Crossover (Recombination) The chromosomes to be crossed over and then mate are selected randomly and so is the point (location within the string) of the crossover. Two chromosomes are replaced by a pair of their offsprings.[b]

Mutation This is formed on a bit-by-bit basis and every bit in the population has an equal chance of undergoing mutation. The probability of mutation gives the likelihood that a given bit will be mutated. If P_m is the probability of mutation, the population size is *Pop,* and each chromosome is n bits long, then the number of bits to be mutated, on average, is equal to $P_m \times Pop \times n$. The specific bits to be mutated are selected randomly.

The end of the mutation step marks the end of one iteration, or one generation (and the beginning of the selection step, which is also the beginning of the next generation).

Value of Parameters The last GA component to be discussed is the values of the different parameters. There are many of those parameters but the most critical are the population size,

number of generations, portability of crossover, and the probability of mutation. A suitable choice of values of these parameters is important for the proper functioning of the GA. Later discussion will show that the particular choice of any of these values can have a profound impact on the performance of the GA. As will be noted later, there are some guidelines for how to choose those values although much is left to the user's judgment.

Michalewicz (1994) used a population of 50 chromosomes, 0.25 for Pc, 0.01 for Pm, and 150 generations and found the best chromosome of (11110011010001000000101), which corresponds to $x = 1.850773$, to maximize the value of the function to 2.850227.

WHY DO GAs WORK?

In order to understand why GAs work, some fundamental properties should be recognized and their impact on survival of a chromosome understood. First, we introduce the concept of schema (Holland, 1975) as it is fundamental to the understanding of GAs and how they work. A schema is a similarity template describing a subset of chromosomes with similarities at certain string positions (Goldberg, 1989). For example, in a population of five-bit long chromosomes, the schema *0000 matches two chromosomes 10000 and 00000. A "0" or "1" is a fixed position; "*" is not since it can be either a 0 or a 1. A schema matches a particular chromosome if at every location in the schema a 1 matches a 1 in the chromosome, a 0 matches a 0, or a * matches either. Most GA literature refers to the "*" as a "do not care" symbol. The location of the "*" within a schema determines two important properties: Defining length and order, which both in turn impact the survivability of the chromosome during the GA working process. The Defining Length is the distance in bits between the first and last fixed position. It defines the compactness of information contained in a schema. The order of the schema is the number of 0 and 1 positions (number of fixed positions). The defining length impacts a schema's survivability against crossover. And the order of a schema impacts it survivability against mutation. Needless to say, we would like a good schema to survive. Think of a good schema as a building block to a good solution. Another relevant property of the schema is its fitness at a given time t.

Key to understand why a GA works efficiently [Holland (1975) calls it implicit parallelism] is to know how a schema grows from generation to generation. Knowing that a schema's selection into a following generation is based on the ratio of its fitness to the total fitness of the population, an above average schema receives an increasing number of chromosomes (copies) into the next generation; a below average schema receives decreasing number of chromosomes, and an average schema stays on the same. The long-term effect of this is that the above average schema receives an *exponentially* increasing number of chromosomes in the next generations. As it turns out, a short, low-order, above-average schema receive exponentially increasing number of chromosomes. The immediate result of this is that "a GA is able to explore the search space using short, low-order schemata, which, subsequently, are used for information exchange during crossover. Goldberg (1987) noted that a GA seeks near-optimal performance through the juxtaposition of short, low-order, high-performance schemata—which he called the building blocks. Put simply, a GA works wonderfully because by working with the building blocks it reduces the complexity of the optimization problem: instead of building high-performance chromosomes by trying every conceivable combination, the GA constructs better and better chromosomes from the best partial solutions of past samplings.

TYPES OF PROBLEMS TO WHICH GAs ARE MOST SUITED

In brief, a GA is a suitable choice for problems that are difficult to formulate and solve using derivative-based and other traditional optimization techniques. Problems that are characterized by complex objective functions including multiobjective problems, problems with no-closed form objective function, and ones with large number of variable and mixed solution space are particularly suited for optimization with GAs.

A GA has no requirement of the function being optimized; it does not need to be differentiable, continuous, monotonic, etc. In fact it can be as ill-behaved as it may and that would not have any bearing on its implementation in a GA—all the user needs from the function is to be able to calculate a value of the function given values of the different decision variables. In more practical terms, a GA is a good choice to optimize complex and combinatorial problems. Of course the term "complex" is a loaded one that can mean discontinuous, nondifferentiable, nonclosed form, multiobjective, etc. As a side point, GAs are particularly suited for multiobjective optimization problems.

Strengths of GAs

There are a number of specific attributes of GAs that give them an edge over other traditional optimization techniques. These are:

1. A GA works from a population, not a single point, and hence it is less likely to be trapped at a local optimum.
2. Derivative freeness. A GA does not need the objective function's derivative to do its work.
3. Flexibility. A GA can function just fine regardless of how complex the objective function; the only thing it requires of the function is that it be executable (i.e., its value can be calculated given the values of the decision variables).
4. Because of its implicit parallelism, a GA can handle combinatorial problems efficiently. It was shown that as the size of the search space or number of solutions increases exponentially, the time requirements for the GA grow only linearly. This feature is particularly useful for on-line optimization of transportation problems such as traffic control (Abu-Lebdeh and Benekohal, 1999, 2000).
5. A GA naturally lends itself to parallel implementation. This follows from its functional components–structure.
6. Intuitive guidelines and flexible structure and use of parameters. GAs are for the most part based on intuitive notions and concepts.[c]

Weaknesses of GAs

GAs have limitations as well:

1. Analytical opacity (murkiness). One cannot do much analytical studies on GAs. This follows from the randomness inherently in GAs and the fact that much of how the GA is designed depends on the problem at hand. Most of what we understand about GAs is based on empirical studies

2. Slowness. Because GAs work without derivatives, they are bound to be slower.
3. Global optima not guaranteed. Because of the nature of GAs, finding the global optima is not guaranteed. A GA is a heuristic search method and although randomness is a characteristic of GA's search, the search is not purely random; it is "directed."
4. Iterative nature. There is no clear rule on when the GA should stop. In some cases users run the GA for a pre-specified number of runs, or generations. In other cases one may want to observe the generational rate of improvement and decide accordingly. In other cases computational issues may dictate when to stop.

Besides the above, to most users it is still not clear how to "optimally" select the appropriate type of operators and values of parameters (population size, number of generations, crossover probability, etc.). This is ironic given that GAs themselves are an optimization tool.

EXAMPLES OF TRANSPORTATION PROBLEMS

The example applications of GAs in transportation noted below are a sample of the diverse uses of GAs in transportation problems. As such details of the different applications are not noted. This review is also not intended to be a critique of those applications. The applications are presented by specific application area within transportation, but this classification is strictly a convenience than anything else since once a problem is coded in a GA framework, domain differences simply disappear.

Traffic Signal Timing and Control

Of the first GA applications in this area was of Foy et al. (1992). They used a GA to minimize delay in a four-intersection network. Binary coding was used to encode all signal timing parameters. A population size of 50 was used and run for 60 generations. A simple microscopic traffic simulator performed the evaluation. Results indicated an improvement in system performance. Hadi and Wallace (1993) added a GA component to TRANSYT-7F so that all four signal timing variables could be optimized. A two-stage optimization was used with binary representation of variables: first, a simple GA (SGA) optimizes the phase sequence and cycle lengths, and both are then used as input to TRANSY-7F for determining the optimal green splits and offsets. A variant of their work that improved performance but required more computational time was to use a GA to optimize offsets. They experimented on three networks of 7 to 12 intersections and used a population and number of generations of 50 each. Crossover and mutation probabilities were 0.9 and 0.01, respectively. Results showed that with the GA-based phase sequence and cycle length optimization, system performance improved by 15% to 44%.

Abu-Lebdeh and Benekohal (1997, 2000) and Girianna and Benekohal (2002, 2004) presented formulations and solutions to control of oversaturated arterials and network control problems, respectively. Control was formulated as an optimization of dynamical problems and Micro-GAs[d] were used with binary coding to optimize green splits and offsets. Ceylan and Bell (2004) used GAs to optimize signal timing with consideration to traffic assignment. Chen and Abu-Lebdeh (2006) used GAs to simultaneously optimize signal control and dynamic speed selection in signalized networks. Park et al. (1999) used mesoscopic simulation with a GA-based optimizer to simultaneously optimize all four signal parameters with consideration to

oversaturated conditions. They accounted for hard constraints partially by using fractional value representation of parameters. The GA parameter values were: 250 and 10 for the population and number of generations, respectively, and 0.4 and 0.03 for the probabilities of crossover and mutation, respectively. A hypothetical network with three levels of demand was used to test the algorithm. The GA-based solutions outperformed those from TRANSYT-7F for both low and high demands, but were comparable for the medium demand. Later work (Park, 2000) considered multioptimization strategies.

Duerr (2000) used GAs with a microscopic traffic simulator as fitness evaluator to optimize signal coordination to reduce delay to buses, and other vehicles as possible. The simulator considered vehicle behavior at intersections and at transit stops. The GA parameter values were: 200 and 100 for the population and number of generations, respectively, and 0.5 and 0.1 for the probabilities of cross over and mutation, respectively. Results showed a 25% and 5% delay reduction for buses and cars, respectively, but the solution was too computationally demanding to be implemented in real time.

GA for Transit Network Design

The transit network design (TND) problem is about designing new routes or modifying existing ones given the roadway network, travel demand, and operating policies and objectives. The TND problem is commonly formulated as a minimization of the sum of user and operator costs (Baaj and Mahmassani, 1990; Ceder and Wilson, 1986). Especially for large networks, TND problems are typically characterized by computational complexity and inability to obtain optimal solutions.

Chakroborty et al. (1995) used GAs to solve a TND problem where transit routes are given and meet at a common transfer station. The objective is to determine the optimal schedules for these routes subject to minimum passenger waiting and transfer times as well as fixed fleet size and demand. The GA enabled a simpler formulation of this computationally intractable problem; as the GA does not use a derivative to guide its search, the problem was formulated with only one decision variable (arrival–departure times). Integer transfer variables were determined in a sub-procedure based on the solution of the reformulated bus arrival–departure times' problem. A three-route problem was solved. The GA parameter values were: 350 for the population and 0.95 and 0.005 for the probabilities of cross over and mutation, respectively. A sensitivity analysis showed the results to be reasonable but no other effectiveness indicators were given. Later work presented cases with multiple transfer stations (Chakroborty et al., 1998), and the case of schedule coordination between trains and feeder buses (Shrivastava and Dhingra, 2002).

Pattnaik et al. (1998) used GAs with other heuristic procedures to solve a general version of the TND problem. The main problem was broken into two sequential sub-problems. Candidate routes were generated heuristically based on origin–destination (O-D) patterns and policy constraints. A GA was then used to solve the TND problem with the candidate routes constituting the search space. The fitness of a solution was determined after the volume of passengers on each route had been determined using a specified assignment procedure. In depth analysis was used and the optimal GA parameter values were determined to be: 50 and 250 for the population and number of generations, respectively, and 0.6 and 0.05 for the probabilities of crossover and mutation, respectively. The GA was then used to solve the TND problem of 25 routes and 39 links. Tom and Mohan (2003) later extended Pattnaik et al.'s work to include frequency of routes. The work by Bielli et al. (2002) is similar to that of Tom and Mohan (2003).

Ngamchai and Lovell (2000) used a GA with heuristic procedures to identify the optimal routes, transfer points, and headways that minimize the total operator and passenger costs. Initial candidate routes (solutions) were constructed using a branching technique over a spider graph of a known set of demand points that must be connected by one or more routes. Routing decisions did not include how each route should be located on the underlying road network. One unique aspect of this work is the use of seven operators (not like commonly used ones) that can be used to generate new configurations and transfer points from the existing ones. The algorithm was tested on a hypothetical 19-node network. Optimal values for the proposed operators were determined through experimentation; all operators ranged between 0.1 and 0.3, and the number of generations was 40. Chien et al. (2001) used a GA to construct and evaluate alternative feeder bus routes where a single feeder route is to be located within a given service area. GA parameters were calibrated via simulation to yield ranges of population size (50-60), crossover probability (0.8-0.9), and mutation probability (0.05-0.1). This work applies only to rectangular network and directional demand to and from a transfer station or central business district.

Kruchten et al (2006) used GAs to estimate appropriate parameters so that within-household interactions were incorporated in the mode choice phase. Cevallos and Zhao (2006) used a GA to minimize transfer times in a public transit network. Jeon et al. (2006) used a special GA to ease the computational complexity of the discrete network design problem.

Other Transportation Applications

GAs were also used in scheduling of transportation crews and activities, and in transportation logistics (Kwan and Wren, 1996, Zhao et al., 1995, 1995; Kwan et al., 1999; Kwan and Kwan, 2000; Lourenço et al., 2001) and demand responsive transit (Uchimura et al., 2002; Marchiori and Steenbeek, 2000; Helena et al., 2001; Le and Lixin, 2006; Cao, 2006). Miester et al. (2005) used GAs to optimally generate household daily activity schedules based on the structure of a household and the activity agendas of its members. This was done to use activity-based analysis operational for transportation planning.

Calibration of parameters of microscopic traffic simulation models and parameters of other types of models is another area where GAs were used with some success. Lee and Yang (2001) used GAs to calibrate two Paramics parameters with some success (12% improvement over use of default parameter values). GA's parameter values were: 20 and 12 for population size and number of generations, respectively. Cheu et al. (1998) used a GA to calibrate FRESIM through optimizing values of 12 of its parameters so that it replicates observed traffic conditions. They experimented with different of population sizes and numbers of generations and selected 4 and 100, respectively. Significant improvement was noted over use of the default parameter values. Srinivasan et al. (2000) applied a GA to calibrate the parameters of an automated incident detection algorithm and reported improved performance over other established algorithms. Liu and Mahmassani (2000) used a GA to calibrate four of a multinomial probit (MNP) model using artificial data. The GA-based values were very close to the optimal values. Loizos et al (2005) used a GA to optimize the topology of a neural network that was used to estimate the elasticity modulus of the pavement granular layer. Dong et al. (2006) used GAs to calibrate departure time and route choice parameters of microsimulation models. Zhang and Xie used GAs to improve the predictive abilities of NNs for detection of accidents at signalized intersections in real time. Hegeman and Hoogendoorn (2006) used GAs to minimize the Kolmogorov–Smirnov distances between observed and simulated gaps in order to estimate a distribution of critical gaps.

Jha et al. (2001) used GAs with GIS to present an advanced approach to visualization and design of roadway alignments.

GUIDELINES—OR PITFALLS TO AVOID

While GAs have been shown to work well, they do not work magic and in fact they could easily be misleading. They have limitations, and shrewd use is necessary to ensure good results. The three primary areas where users can have difficulty are (*a*) GAs, especially SGAs, can easily get trapped at local optima of deceptive problems; (*b*) selecting of parameter values (and selecting some specialized operators) is still an art that requires significant offline work by users; and (*c*) finding the global optimum (optima) is not guaranteed. Each of these points is discussed below with some details.

Simple GAs and Deceptive Problems

For particularly difficult optimization problems (deceptive ones), SGAs can easily be trapped at local optima. Assuming that the problem at hand cannot be solved with other more traditional optimization methods, users should be aware of the fact that a GA can be trapped at a local optimum, a problem that GAs are often able to overcome, but not always. Complicating this is the fact that there is nothing inherent in a GA structure to indicate that it is in fact trapped at a local optimum, which makes it critical that users be aware of this limitation. If it is suspected that a GA is indeed trapped at a local optima, a user should explore, perhaps aggressively, widely varying values of the key parameters (population size, number of generations, and to a lesser extern the probabilities of crossover and mutation). Even then, this is not a guarantee that the problem is solved as will be discussed later. More recent generations of GAs address this but do not solve it entirely.

Selecting Parameter Values

While GAs are widely used as an optimization tool, selection of the best values of the different parameter is still an ad hoc process despite a significant volume of both theoretical and empirical research to address this very point. Most theoretical research used “toy” problems of known structure and complexity to derive some rules or guidelines on selection of parameter values (Thierens et al., 1999; Thierens and Goldberg, 1993; Goldberg, 1998a). The problem with that approach is that in most practical cases users have little knowledge of the inner working or structure of their problems. The empirical research on the other hand used practical problems and experimented with wide ranges of values for the parameters, and some preliminary rules were extracted (Abu-Lebdeh and Benekohal, 1999; Abu-Lebdeh and Al-Omari, 2003). The problem with this approach is that no two problems are the same, and even if they are, users are still burdened with having to compare structures of problems. Hence the results of the empirical approach are difficult to generalize. This whole issue is ironic since GAs are used as an optimization tool and yet we are not even close to knowing how to optimally select the values of the different parameters. One can argue that experimenting with a large enough pool of values solves the problem, but that is impractical to say the least.

Genetic Algorithms and Global Optima

Because of the nature of GAs, there is no guarantee that a GA will find the global optima (assuming there is one). That there is nothing in the GA structure or "behavior" to detect that makes this a potentially serious problem. Hence a less-than-prudent use of GAs can easily result in inferior results. This is a particularly relevant problem when the primary motivation for using a GA is the "novelty" of the approach; if that is the case, a GA may not be a good choice—actually it could be a poor one. That is one reason why the best result of GA should be called a "near-optimal" as opposed to "optimal."

IMPROVING GA PERFORMANCE

Virtually all of the GA applications in transportation have used SGAs. But SGAs may not be suitable for some problems because they are not easy to configure, can be too slow to converge, and have difficulty solving deceptive and hard problems. Recent advances in evolutionary computation can alleviate many of these shortcomings. In fact some of those advances are revolutionizing the use of GAs whereby hard problems can now be solved efficiently and in reasonable time. Although some of those advances are still at the research stage, several have matured enough to be used in real-life applications and they started to appear in non-evolutionary computation literature. Newer generations of GAs are upon us, and the issue now is not whether to use GAs but rather "how best" to use them.

Six courses may be taken to make the application of GAs efficient, rapid, and productive:

1. Selection of appropriate operators and parameter values;
2. Appropriate problem-specific representations of candidate solutions;
3. Faster and better evaluation of solutions (or individuals);
4. Structuring of individuals into subpopulations or various other classes that are treated separately with respect to application of various operators, etc.;
5. Division of workload among multiple loosely coupled processors (as in a cluster or network, for example); and
6. Hybridizing GAs with other non-evolutionary search methods.

It is intuitive that in order to get the most from the GA, the first three areas need to be "done" right. The fifth action can be done regardless of the structure of GA employed or any of the other actions. The sixth area is less fundamental to the functioning of the GA per se; it is more of a supplement, but for the right type of problem, it can be valuable. A GA user will almost always get results even with a basic problem representation and primitive selection of parameter values. Those results, however, can easily be inferior, and the user might not know it. As noted earlier, there is nothing about the GA's mechanics that will either assure optimal search or "warn" the user of bad search results.

The rest of the discussion primarily concerns points 3, 4, and 5. But before discussing those, the other areas are briefly discussed.

Appropriate Operators and Parameter Values

One approach to improving GA search performance is to simply test different values for mutation rates, population size, number of generations, etc. Selection of different values for those parameters has become easier since the work of Goldberg (1989 and 2002, books and later papers) and his students helps to provide some guidance regarding those parameters, given some characterization of the difficulty of the problem. The Schema Theorem (Holland, 1975) provides additional guidance. In many cases, however, the use of those rules and guidelines requires specialized knowledge of specific properties of the optimization problem, which may not be easy for typical nonevolutionary computation researchers or user. If changing the configuration parameters has no effect on the search performance then a more fundamental problem may be the cause.

Problem Representation

GAs can be used with either binary or real coding, and with many forms, depending on the nature of the problem to be solved. A simple method of improving GA performance is sometimes to change the genetic representation. Many researchers published results showing that binary coding worked better for their applications, while other researchers report different results (Deb, 2001). Particularly for combinatorial optimization problems, the topic of appropriate representations is the focus of a great deal of current research (Rothlauf and Goldberg, 2002). There are certain advantages to each representation scheme that can be exploited for certain applications.

Problem representation (binary versus nonbinary) becomes very important when the search space is inherently continuous, or even ranges over a number of integer values. For example, with binary coded GAs, Hamming cliffs are a common difficulty (Hamming cliffs appear where transition between neighboring individuals in the phenotypic space requires alteration of many bits of the genotype). This translates into an "artificial" difficulty to a gradual search in a continuous space (Deb, 2001). Gray codes or phenotype-based mutation operators are possible solutions. Also, with higher precision, longer strings are needed, and with longer strings, larger populations are needed, thus increasing the computation time. There are additional problems caused by the fixed coding scheme. Discussion of those issues is beyond the scope of this circular.

Real-parameter GAs are an option that has some advantages, since real-parameter values are easier to deal with. But there are also issues with real-number GAs. The crossover and mutation operators are not as easy or intuitive to apply as for binary codes. There is no universal resolution for binary versus real coding. Deb (2001) noted that there are many arguments for using binary codings, and at least one against it (i.e., for GAs that fundamentally represent real numbers).

Faster Evaluation of Solutions

Even when the GA is optimally configured and the problem is optimally represented, the user should ensure that the best solution is obtained as soon as possible. This is a particularly critical point for real-time online optimization such as real-time traffic control. In these cases, the system operator is usually interested in the GA's best-solution-so-far. Anything that can be done to

reduce the time to evaluate each solution can contribute enormously to reducing total time to identify good solutions. For example, rapid rejection (and assignment of poor fitness (objective function values) to solutions that perform poorly in an initial time period can reduce the average evaluation time dramatically.

Hybridizing GAs with Nonevolutionary Search Methods (Hybrid or Memetic Algorithms)

Many researchers have found that it is beneficial to augment the GA with additional search operators. Two types are commonly used—generic local search (such as using simulated annealing, nonlinear sequential quadratic programming, neural nets, etc.) or problem-specific heuristic search rules. In either case, it is possible to subject each individual generated by the GA to local search, or to apply local search only when particular "triggers" occur. It also is possible to replace the starting GA-generated genotype (solution) by the representation of the newly-created solution ("Lamarckian" approach) or merely to assign the locally optimized fitness to the original GA-generated genotype ("Baldwinian" approach).

PARALLEL GENETIC ALGORITHMS

A distinction should be made between parallel GAs and parallel "implementations" of a SGA. By parallel GA, this circular shall mean one in which the structuring of the population(s) is into some set of demes or subgroups which are treated separately, whether these groups are very large or as small as single individuals. A parallel GA can be implemented on a single processor or can be implemented across multiple, loosely connected processors, but in either case, the generation of new individuals is affected by the structuring of the population into multiple groups. This is a separate issue from whether or not the GA is implemented across multiple processors (e.g., a network of loosely coupled computers). Either a SGA or a parallel GA can be implemented on multiple processors, with no difference in the actual course of the search (unless asynchronous operation is allowed) [see Cantu-Paz (2000) for more discussion]. Below, parallel GAs are discussed along with considering the possibilities of implementing them on multiple processors, further speeding their search time, but without fundamentally affecting their search trajectory. But a GA which uses multiple processors to evaluate in parallel the individuals in a single GA population (below, it is called the "global parallel GA") will not be considered to be a parallel GA (PGA) in the strictest sense.

Global Parallel Genetic Algorithms

Parallel hardware can be utilized with a SGA in the master–slave architecture. Here, an overall node called the master initializes and contains the entire population and performs the selection operation and any needed rescaling of fitnesses. A number k of "slave" nodes perform at least the function evaluations, in a parallel fashion, hence improving the speed of execution of the GA (Figure 20).

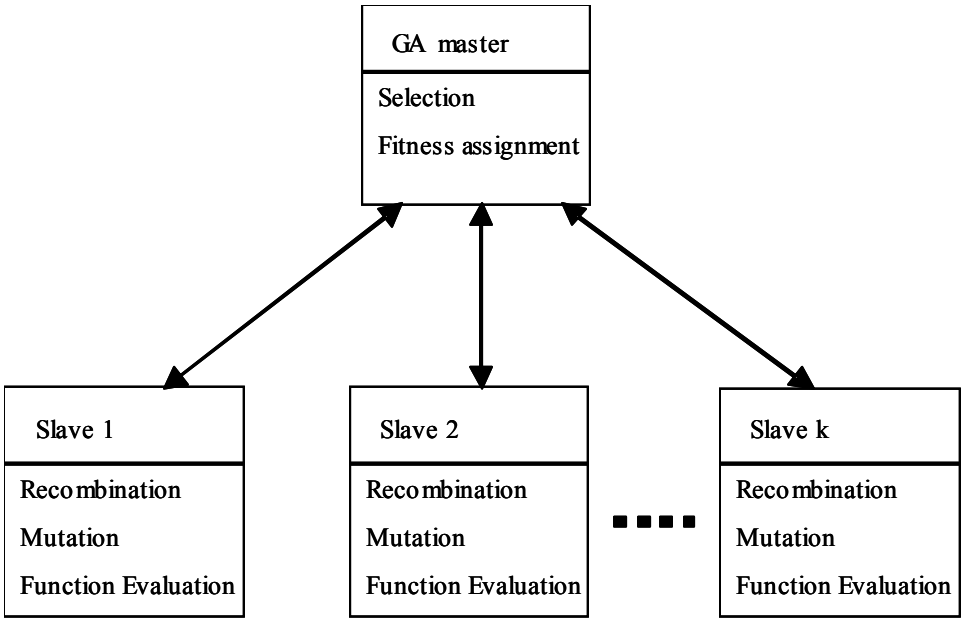

FIGURE 20 Global parallel GA.

If individuals are passed in a group to each slave, the slaves may also perform recombination and mutation in parallel, also relieving the master of some work. In this type of implementation the algorithm has to balance serial-parallel tasks to minimize bottlenecks hence the issue of synchronous–asynchronous operation is an important consideration. However, unless the imbalance among processors is large, either mode of operation can produce similar results. The distributed evaluation of GA is appealing when objective function evaluation is expensive, but the same advantages of parallel execution can be gained by PGAs using parallel hardware, as well.

Increasing the number of slaves, n, obviously will increase the efficiency of the GA; however, the communication requirements also increase. Therefore there is a point beyond which adding more slave nodes become counterproductive; however, for problems with long evaluation times for each individual, this limit is very large. Cantu-Paz and Goldberg derived a relationship that determines the optimal number of nodes (Cantu-Paz and Goldberg, 1999).

Parallel Genetic Algorithms

In contrast to the global parallel GA described above, PGAs are based on GAs, but instead of considering a single fully-mixed (panmictic) population (i.e., any individual can be crossed over with any other individual to produce offspring), PGAs treat individuals as being divided into groups, or as spatially distributed in some nonhomogeneous fashion.

Here, two types of PGAs—island PGAs and diffusion PGAs—are discussed. The main differences between those types are in the population structure and method of selecting individuals for reproduction. The following subsections briefly describe these two types of PGA.

Island Parallel Genetic Algorithms

In migration or island or coarse-grained PGAs (first introduced by Grosso in 1985), the population is divided into small clusters, each of which is treated as a separate breeding unit under the control of a conventional GA. As noted in Figure 21, the island PGA does not operate globally on a single population. Occasionally, individuals from the various subpopulations (islands) are permitted to migrate to other islands, where they may subsequently mate with other members of that island. There are many different implementations (topologies) of the island PGA scheme—some examples are shown in Figure 21.

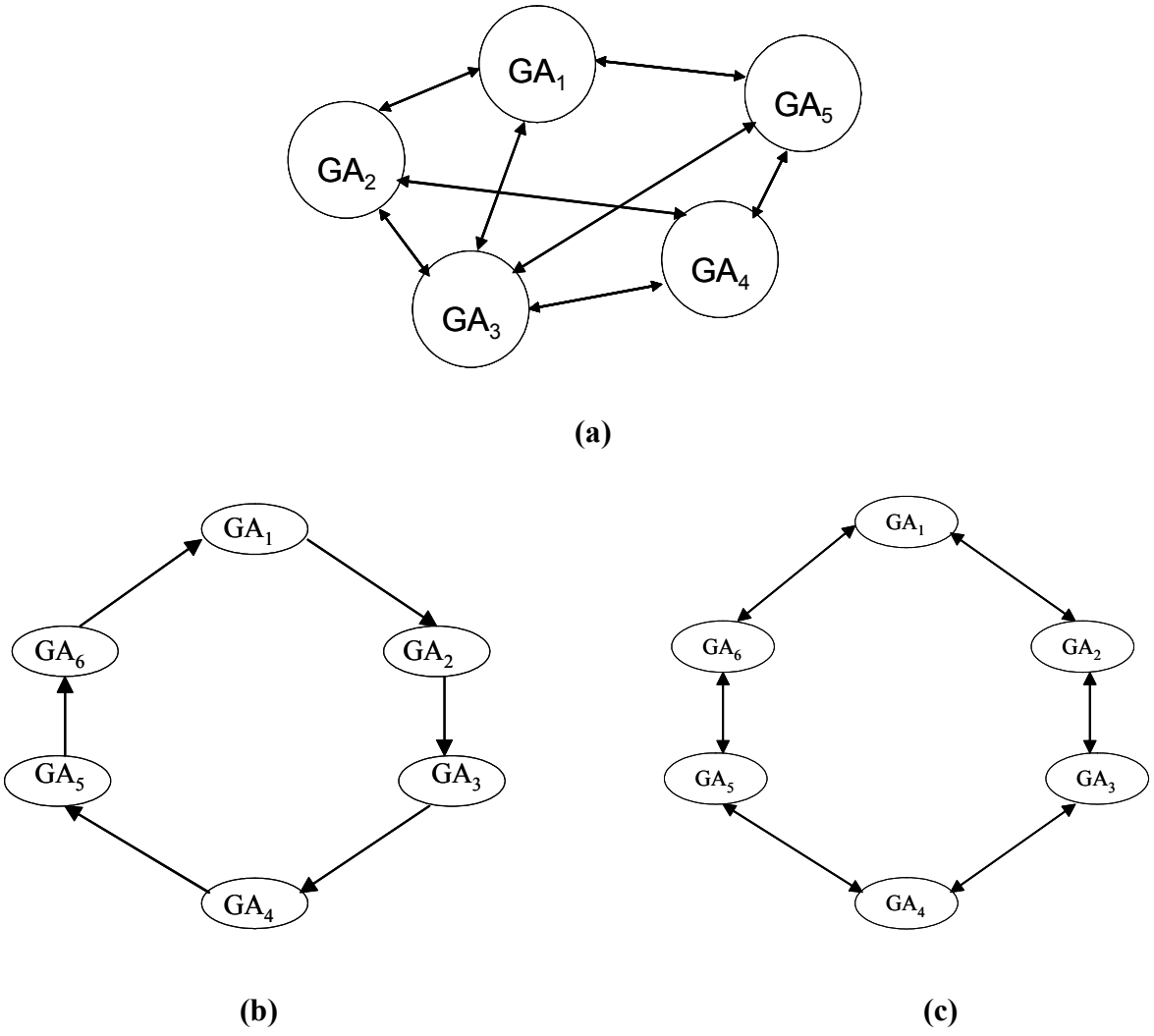

FIGURE 21 Topologies of migration PGA : (*a*) unrestricted migration; (*b*) ring migration; and (*c*) neighborhood migration.

A few additional key parameters need to be defined when using an island PGA. The interval between migrations and the number of individuals to migrate are the most notable. Additionally, one has to decide on which individuals are going to migrate. The topology is another decision which is determined by the target subpopulation. Traditionally, the topology is a ring and one individual—the best, or a randomly selected one—migrates at each migration step, which takes place at predefined cyclic points in time. Many other migration schemes have also been successfully employed. A typical pseudo-code for an island PGA follows:

```
-- For each node (GAi)
WHILE not finished
 SEQuential
        … Selection
        … Reproduction
        … Evaluation
PARallel
        … send emigrants
        … receive immigrants
```

A notable advantage of island PGAs is a reduction in takeover time by superior individuals. A classical problem, premature convergence, affects SGAs more strongly than island PGAs: when a superior individual is found, a SGA will tend to begin converging toward that individual, skewing all future search. In contrast, until that individual or its descendants propagate through all the subpopulations of an island PGA, search in the remaining islands goes on unaffected by that individual. It is possible that several superior individuals will exist in an island PGA at any time and migration allows for their recombination, but delays the convergence of the entire population to any of their genotypes.

The tradeoff is that to avoid an excessive number of function evaluations, each subpopulation's size must be smaller than the total population of the SGA. Thus the choice of number of subpopulations and subpopulation sizes is a subtle and problem-dependent issue, which introduces some additional complication for the user of this GA. However, the advantages of island parallel GAs have been demonstrated many times (Cantu-Paz, 2000).

Hierarchical GAs (HGAs) are another category of island PGA. They use a hierarchical topology for the layout of the subpopulations, as noted in Figure 22. In fact, if the GA subpopulations are also allowed to be heterogeneous (i.e., represent the problem or calculate fitnesses differently), this hierarchical topology makes it possible to have different layers performing different tasks. Such an architecture, the injection island GA, was first introduced by Lin et al. (1994). In one such implementation, the top layer concentrates on refinement while the bottom layer performs mostly exploration (Figure 22). This type of implementation addresses the dilemma of having to choose between complex modeling that requires a long time to compute a fitness function or coarser but faster models. HGAs with multiple models provides a way out of this problem through using distinct models for each level of the hierarchy.

A recent form of PGA uses the hierarchical fair competition principle (HFC) to structure a set of subpopulations (Hu, 2002). Subpopulations are stratified according to fitness brackets, and whenever a newly created individual has a fitness higher than the range of the bracket in which it originates, it is moved out to a subpopulation with a fitness bracket corresponding to the individual. This allows for rapid exploitation of high-fitness individuals through recombination

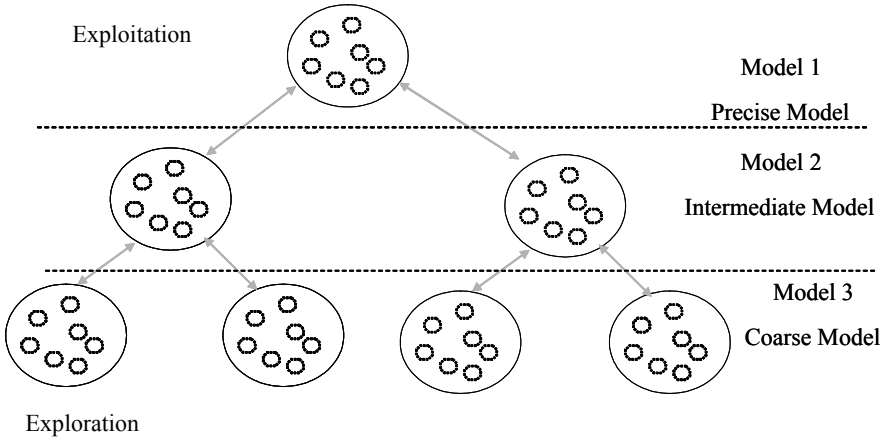

FIGURE 22 Three-level hierarchical GA.

with others of like fitness, but keeps high-fitness individuals from taking over subpopulations composed mostly of low-fitness individuals, thereby restricting future exploration.

Any of these forms of island PGA can be implemented almost trivially across multiple computers or processors: each processor is assigned one or more islands, and communicates migrants to/from other subpopulations (via a buffer or synchronously) at appropriate intervals.

Diffusion (Cellular) Parallel Genetic Algorithm

A diffusion PGA (also know as a cellular or massively parallel GA) is similar to the island PGA but overcomes the discontinuities generated by the island PGA. Here the diffusion PGA represents the population as a single spatially distributed population with individuals being assigned a location within some (typically two- or three-dimensional) space. Mating is allowed between individuals in the same or neighboring cells. Genetic operations take place in parallel (conceptually, at least) for every node of the population, and every individual interacts only with those in its neighborhood. The replacement policy typically destroys the considered individual by overwriting it with the newly computed chromosome. A typical structure of a diffusion PGA is shown in Figure 23.

In a diffusion PGA, the population is a single continuous structure, but each individual is assigned a spatial location. Mating is only allowed within a small local neighborhood. For example, in Figure 23, $I(2,2)$ can only mate with $I(1,2)$, $I(2,1)$, $I(2,3)$, $I(3,2)$. This isolation-by-distance property allows a high diversity, and the selection pressure is also weaker due to the local selection operator. The appearance of new species of solution in the grid and the refinement of existing solutions are both permitted and desired. In that respect, a diffusion PGA allows a well-developed balance between exploitation (of high quality chromosomes) and exploration (of the search space).

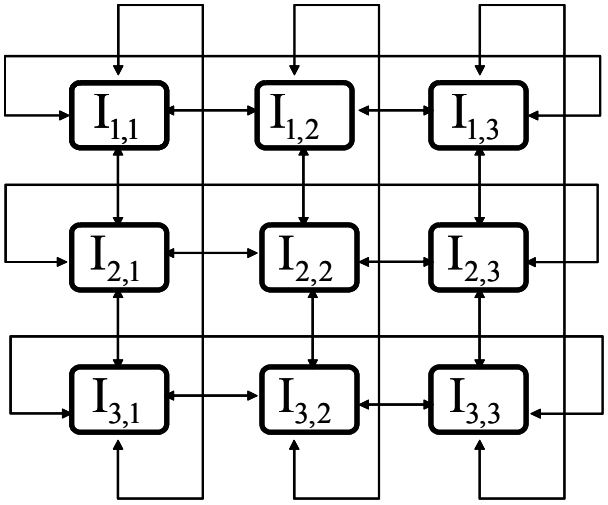

FIGURE 23 Diffusion or cellular PGA.

Diffusion PGAs have often been implemented on multiprocessors due to the close similarities between the model and the physical arrangement of central processing units. Another possible approach is to simulate the diffusion PGA in a network of workstations. However, the same architecture can be simulated on a single processor, at the cost of longer run times. A pseudocode for cellular PGA follows:

```
-- Each node (Ii,j)
WHILE not finished
SEQuential
… Evaluate
PARallel
        … send self to neighbors
        … receive neighbors
… select mate
… reproduce
```

Cantu-Paz (2000) discusses cellular PGAs in some detail, and provides more information on their advantages and disadvantages relative to island PGAs. In the limit, a large enough set of subpopulations and small enough neighborhood for migration in an island PGA can yield equivalence to a cellular PGA, so the distinction is largely conceptual, rather than a firm boundary.

SUMMARY

This section presented one of the most popular derivative-free optimization methods: GAs. This technique relies n modern high-speed computing and entails more computation as compared to

derivate-based approaches. However, it is more flexible as it makes uses of institutive guidelines and principles to formulate complex objective functions and account for necessary constraints.

NOTES

a. This example is largely after Michalewicz, 1994.
b. Other schemes where some of the parents are not replaced are possible.
c. Users who are not well-versed in the theory of GAs may argue that the last point is in fact a weakness in that it makes their use of GAs a little onerous.
d. These are GAs with very small population, no mutation, and repetitive injection of new chromosomes.

REFERENCES

Abu-Lebdeh, G., and B. H. Al-Omari. Configuring Micro-Genetic Algorithms for Solving Traffic Control Problems: The Case of Number of Generations. *Proc., 4th International Conference on Uncertainty Modeling and Analysis*, 2003, pp. 70–75.

Abu-Lebdeh, G., and R. F. Benekohal. Micro-Genetic Algorithms for Adaptive Signal Coordination. *Proc., 6th International Conference on Use of Advance Technologies in Transportation*, Singapore, June 2000a.

Abu-Lebdeh, G., and R. F. Benekohal. Development of Traffic and Queue Management Procedures for Oversaturated Arterials. In *Transportation Research Record 1603*, TRB, National Research Council, Washington, D.C., 1997, pp. 119–127.

Abu-Lebdeh, G., and R. F. Benekohal. Genetic Algorithms for Traffic Signal Control and Queue Management of Oversaturated Two-Way Arterials. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1727*. TRB, National Research Council, Washington, D.C., 2000b, pp. 61–67.

Abu-Lebdeh, G., and R. F. Benekohal. Convergence Variability and Population Sizing in Micro-Genetic Algorithms. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 14, 1999, pp. 321–334.

Cantu-Paz, E. Efficient and Accurate Parallel Genetic Algorithms. Kluwer-Academic Publishers, 2000.

Cantu-Paz, E., and D. E. Goldberg. Parallel Genetic Algorithms: Theory and Practice. *Computer Methods in Applied Mechanics and Engineering*, 1999.

Cao, Z., D.-H. Lee, and Q. Meng. Scheduling Two-Transtainer Systems for Loading Operation of Containers Using Revised Genetic Algorithm. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Cevallos, F., and F. Zhao. Minimizing Transfer Times in a Public Transit Network with a Genetic Algorithm. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Chakroborty, P., Deb, K., and Srinivas, B. (1998): Network-Wide Optimal Scheduling of Urban Transit Networks Using Genetic Algorithms. *Journal of Computer Aided Civil and Infrastructure Engineering*. 13, 363-376.

Chakroborty, P., K. Deb, and P. S. Subrahmanyam. Optimal Scheduling of Urban Transit Systems Using Genetic Algorithms. *Journal of Transportation Engineering,* Vol. 121, No. 6, 1995, pp. 544–553.

Chen, H., and G. Abu-Lebdeh. Development of a Framework for an Integrated Dynamic Signal-Dynamic Speed Traffic Management Algorithm for Signalized Networks. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Cheu, R. L., X. Jin, K. C. Ng, Y. L. Ng, and D. Srinivasan. Calibration of FRESIM for Singapore Expressway using Genetic Algorithm. *Journal of Transportation Engineering*, Vol. 124, No. 6, 1998, pp. 526–535.

Chien, S., Z. Yang, and E. Hou. Genetic Algorithm Approach for Transit Route Planning and Design. *Journal of Transportation Engineering*, Vol. 127, No. 3, 2001, pp. 200–207.

Ceylan, H., and M. Bell. Reserve Capacity for a Road Network Under Optimized Fixed-Time Traffic Signal Control. *Journal of ITS: Technology, Planning, and Operations*, Vol. 8, No. 2. 2004, pp. 87–99.

Deb, K. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley and Sons, New York, 2001.

Dong, H., J. Ma, and M. H. Zhang. Calibration of Departure Time and Route Choice Parameters in Microsimulation with Macro Measurements and Genetic Algorithm. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Duerr, P. 2000 Dynamic Right-of-Way for Transit Vehicles: Integrated Modeling Approach for Optimizing Signal Control on Mixed Traffic Arterials. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1731*. TRB, National Research Council, Washington, D.C., 2000, pp. 31–39.

Foy, M., R. F. Benekohal, and D. E. Goldberg. Signal Timing Determination Using Genetic Algorithms. In *Transportation Research Record 1365,* TRB, National Research Council, Washington, D.C., 1992, pp. 108–115.

Girianna, M., and R. F. Benekohal. Using Genetic Algorithms to Design Signal Coordination for Oversaturated Networks. *Journal of ITS: Technology, Planning, and Operations,* Vol. 8, No. 2, 2004, pp. 117–129.

Girianna, M., and R. F. Benekohal. Dynamic Signal Coordination for Networks with Oversaturated Intersections. *Transportation Research Record 1811,* TRB, National Research Council, Washington, D.C., 2002, pp. 122–130.

Goldberg, D. G. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer Academic Publishers, 2002.

Goldberg, D. G. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

Goldberg, D. E. Sizing Populations for Serial and Parallel Genetic Algorithms. Proc., 3rd International Conference on Genetic Algorithms, 1989a, pp. 70–79.

Grosso, P. B. Computer Simulations of Genetic Adaptation: Parallel Subcomponent Interaction in a Multilocus Model. Ph.D dissertation, University of Michigan, 1985.

Hadi, M. A., and C. E. Wallace. Hybrid Genetic Algorithm to Optimize Signal Phasing and Timing. *Transportation Research Record 1421,* TRB, National Research Council, Washington, D.C., 1993, pp. 104–112.

Hegeman, G., and S. Hoogendoorn. Simulation-Based Critical Gap Distribution Estimation Using Genetic Algorithms. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Lourenço, H. R., J. P. Paixão, and P. Portugal. Multiobjective Metaheuristics for the Bus Driver Scheduling Problem. *Transportation Science*, Vol. 35, No. 3, 2001, pp. 331–343.

Henderson, J., and L. Fu. Applications of Genetic Algorithms in Transportation Engineering. Presented at the 83rd Annual Meeting of the Transportation Research Board, Washington, D.C., 2004.

Holland, J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, Mich., 1975.

Hu, J., and E. D. Goodman. Hierarchical Fair Competition Model for Parallel Evolutionary Algorithms. Proc., Congress on Evolutionary Computation, CEC 2002, IEEE World Congress on Computational Intelligence, Honolulu, Hawaii, 2002.

Jeon, K., J. S. Lee, S. V. Ukkusuri, and S. T. Waller. New Approach for Relaxing Computational Complexity of Discrete Network Design Problem Using Selectorecombinative Genetic Algorithm. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Jha, M. K., C. McCall, and P. Schonfeld. Using GIS, Genetic Algorithms, and Computer Visualization in Highway Development. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 16, No. 6, 2001, pp. 399–414.

Kruchten, N., E. J. Miller, and M. J. Roorda. Incorporating Within-Household Interactions into Mode Choice Model Using Genetic Algorithm for Parameter Estimation. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Le, Y., and L. Miao. Genetic Algorithm for Bilevel Programming Model of Public Logistics Terminal Planning. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Lee, D. H., Y. Xu, and P. Chandrasekar. Parameter Calibration for PARAMICS Using Genetic Algorithm. Presented at 80th Annual Meeting of the Transportation Research Board, Washington, D.C., 2001.

Lin, S.-C., W. F. Punch, and E. D. Goodman. Coarse-Grain Genetic Algorithms, Categorization and New Approaches. *Sixth IEEE Parallel and Distributed Processing*, 1994, pp. 28–37.

Liu, Y. H., and H. S. Mahmassani. Global Maximum Likelihood Estimation Procedure for Multinomial Probit Model Parameters. *Transportation Research Part B,* Vol. 34, 2000, pp. 419–449.

Loizos, A., A. Karlaftis, and M. G. Karlaftis. A Genetically Optimized Neural Networks Approach for Estimating the Moduli of Unbound Pavement Materials. Presented at the 84th Annual Meeting of the Transportation Research Board, Washington, D.C., 2005.

Lourenço, H., J. P. Paixão, and R. Portugal. Multiobjective Metaheuristics for the Bus Driver Scheduling Problem. *Transportation Science*, Vol. 35, No. 3, 2001, pp. 331–343.

Marchiori, E., and A. Steenbeek. An Evolutionary Algorithm for Large-Scale Set Covering Problems with Application to Airline Crew Scheduling. Lecture Notes in Computer Science, Springer Berlin, Heidelberg, Vol. 1803, 2000, pp. 367–381.

Meister, K., M. Frick, and K. W. Axhausen. A GA-based household scheduler. Presented at the 84th Annual Meeting of the Transportation Research Board, Washington, D.C., 2005.

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

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

Ngamchai, S., and D. J. Lovell. Optimal Time Transfer in Bus Transit Route Network Design Using a Genetic Algorithms. 8th International Conference on Computer-Aided Scheduling of Public Transport, Berlin, 2000.

Park, B., J. Messer, and T. Urbanik. Traffic Signal Optimization Program for Oversaturated Conditions. In *Transportation Research Record 1683,* TRB, National Research Council, Washington, D.C., 1999, pp. 133–142.

Park, B., J. Messer, and T. Urbanik. Enhanced Genetic Algorithm for Signal-Timing Optimization of Oversaturated Intersections. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1727*, TRB, National Research Council, Washington, D.C., 2000, pp. 32–41.

Pattnaik, S. B., S. Mohan, and V. M. Tom. Urban Bus Transit Route Network Design Using Genetic Algorithm. *Journal of Transportation Engineering*, Vol. 124, No. 4, 1998, pp. 368–375.

Rechenberg, I. Cybernetic Solution Path of an Experimental Problem. Royal Aircraft Establishment, Library Translation No. 1122, Farnborough, Hants, U.K., 1965.

Rothlauf, F., and D. E. Goldberg. Representations for Genetic and Evolutionary Algorithms. Physica-Verlag, 2002.

Shrivastava, P., and S. L. Dhingra. Development of Coordinated Schedules using Genetic Algorithms. *Journal of Transportation Engineering*, Vol. 128, No. 1, 2002, pp. 89–96.

Srinivasan, D., R. L. Cheu, Y. P. Poh, and A. K. C. Ng. Development of an Intelligent Technique for Traffic Network Incident Detection. *Engineering Application of Artificial Intelligence*, Vol. 13, 2000, pp. 311–322.

Tom, V. M., and S. Mohan. Transit Route Network Design Using Frequency Coded Genetic Algorithm. *Journal of Transportation Engineering*, Vol. 129, No. 2, 2003, pp. 186–195.

Tzeng, G. H., and Y. W. Chen. The Optimal Location of Airport Fire Stations: A Fuzzy Multi-Objective Programming and Revised Genetic Algorithm Approach. *Transportation Planning and Technology*, Vol. 23, 1999, pp. 37–55.

Thierens, D., D. E. Goldberg, and A. Pereira. Domino Convergence, Drift, and the temporal Salience Structure of Problems. *Proc., IEEE International Conference on Evolutionary Computation*, 1999, pp. 535–540.

Thierens, D., and D. E. Goldberg. Mixing in Genetic Algorithms. *Proc., 5th International Conference on Genetic Algorithms*, 1993, pp. 38–45.

Uchimura, K., H. Takahashi, and T. Saitoh. Demand Responsive Services in Hierarchical Public Transportation System. *IEEE Transactions of Vehicular Technology*, Vol. 51, No. 4, 2002, pp. 760–766.

Yunlong Zhang, Y., and Y. Xie. Application of Genetic Neural Networks in Real-Time Intersection Accident Detection Using Acoustic Signals. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Agent-Based Modeling in Transportation

KRISTEN L. SANFORD BERNHARDT
Lafayette College

Agent-based modeling (ABM) is a relatively new paradigm compared to some of the other advanced computing paradigms discussed in this document. Researchers and practitioners in many disciplines, from biology to business, have developed agent-based models, and the number of applications continues to rise. Bonabeau (2002b) provides an accessible overview of the variety of applications of ABM.

The purpose of this chapter is to explain briefly what ABM is and what it can be used for, as well as to review some of the many applications developed so far in the transportation domain. Because interest in ABM continues to grow, the number of applications continues to grow as well. The applications described present examples of the type of work that can be done rather than a comprehensive review. This chapter provides a primer of sorts for those wondering whether ABM could be a useful tool for a particular transportation problem.

The following sections define ABM, describe the types of problems to which ABM can be applied, discuss its strengths and weaknesses, provide some examples of transportation applications, and suggest some guidelines for those interested in developing an ABM for a transportation application.

WHAT IS AGENT-BASED MODELING?

Despite the disaggregate nature of transportation systems, we historically have attempted to model them from the top-down. That is, we have started with the behavior we expect to see and have created models that produce this end behavior. ABM takes the opposite approach. It begins with the individual actors (agents) and defines their potential interactions; the simulated interactions of the actors generate the system-level (end) behavior. As described by Bonabeau (2002a), ABM "consists of describing a system from the perspective of its constituent units." He likens ABM to microscopic modeling, as contrasted with macroscopic modeling. Another perspective is to think of ABM as "bottom-up" modeling, as contrasted with "top-down" modeling (Figure 24).

ABM is particularly appropriate for exploring the behavior of complex systems. Complex systems comprise collections of interrelated elements in which the simple behaviors of the basic agents or elements combine in unprogrammed ways to produce sometimes unexpected results. These larger system dynamics are said to emerge from the interactions of the individuals, or the system produces "emergent behavior." That is, system behavior emerges as agents react to other agents and the environment in which they operate or function. As defined by Holland (1988), complex systems are characterized by (*a*) many agents or decision makers with dispersed control, (*b*) many organizational levels, (*c*) the ability of agents to adapt, and (*d*) the use of internal models to anticipate the future.

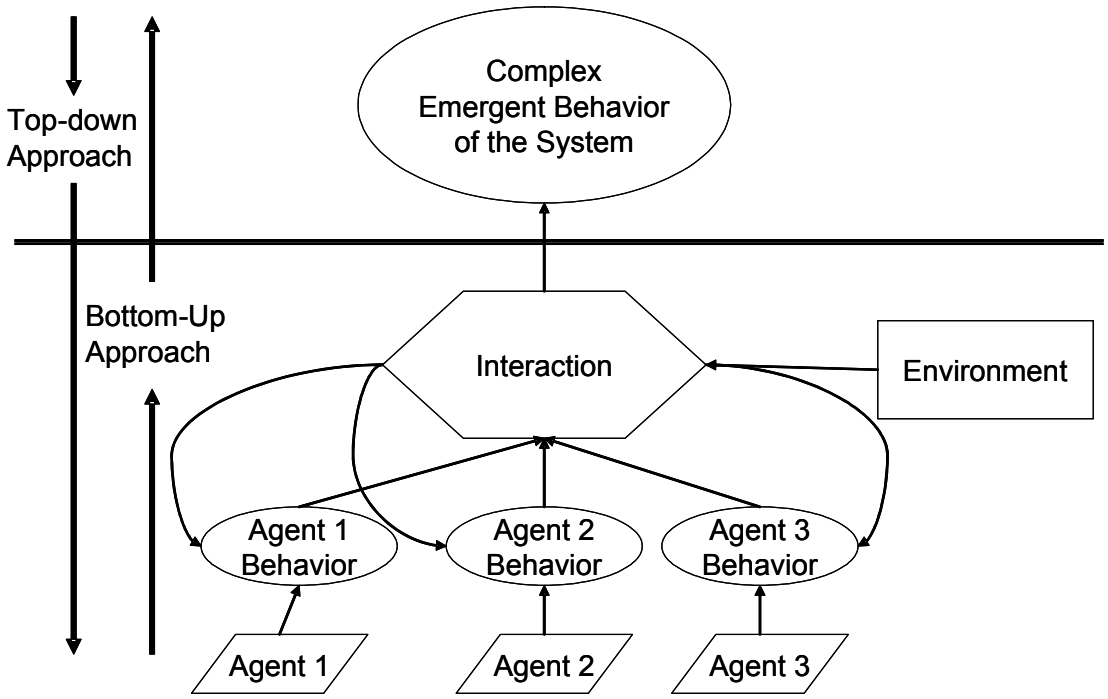

FIGURE 24 A complex systems approach.

Toroczkai and Eubank (2005) describe the characteristics of an agent using traffic modeling as an example. An agent has characteristics that describe its state, how it perceives its environment, and an objective. It also has a set of "allowable actions" and a set of strategies for choosing among them. Finally, an agent has a set of additional characteristics pertaining to its history and constraints on the function it is trying to optimize. Note that each agent optimizing its own objective may not (and often will not) lead to an optimal state for the system as a whole. Following Toroczkai and Eubank's (2005) traffic modeling example:

- The agent's state would include position, speed, driver health, etc.;
- The agent's perception of the environment might include ice on the pavement surface or an upcoming traffic jam;
- Allowable actions might include accelerating and decelerating;
- Strategies determine how the driver will respond to an event—for example, a driver might either brake or swerve if the vehicle in front slows suddenly;
- Characteristics might include number of speeding tickets and constraints might include remaining on the road; and
- The agent's objective might be to minimize its travel time.

Parunak et al. (1998) distinguish between "observables [which] are measurable characteristics of interest," and "behaviors through which individuals interact with one another." A traditional model, then, develops equations to describe the observed characteristics, while an ABM uses rules or equations to describe the individual behaviors. As a result, "Direct

relationships among the observables are an output of the process, not its input" (Parunak et al. 1998).

Axelrod (1997) explains ABM in relation to the commonly understood methods of induction, or learning from empirical data, and deduction, in which one "[specifies] a set of axioms and [proves] consequences that can be derived from the assumptions." An agent-based model actually starts with a set of rules (like deduction), but it uses those rules to generate data that can be analyzed (like induction). According to Axelrod (1997), "the purpose of agent-based modeling is to aid intuition."

PROBLEM TYPES FOR WHICH AGENT-BASED MODELING IS APPROPRIATE

Axelrod and Tesfatsion (2005) suggest that ABM is most appropriate "for studying systems exhibiting the following two properties: (*a*) the system is composed of interacting agents; and (*b*) the system exhibits emergent properties, that is, properties arising from the interactions of the agents that cannot be deduced simply by aggregating the properties of the agents." These researchers suggest that appropriate goals for using ABM are empirical understanding, normative understanding, and heuristic. For example, we might empirically investigate the causes of recurring traffic congestion. Normative understanding might project the effect a new policy, such as implementing an high-occupancy vehicle lane, is likely to have on the system as a whole. Heuristics can help us understand how individual choices lead to counterintuitive consequences—for example, why everyone choosing their perceived shortest travel time fails to yield the lowest overall minimum travel time.

Within the transportation domain, ABM is particularly appropriate for modeling systems in which human decision making and action are a critical component. Examples include highway traffic, pedestrian movements, and demand modeling; these are discussed further in a subsequent section.

Strengths of Agent-Based Modeling

Bonabeau (2002a) characterizes the strengths of ABM as its ability to capture emergent phenomena, provide a natural description of the system, and provide flexibility in modeling. He notes that, of these, the ability to capture emergent phenomena is most significant. He goes on to suggest that ABM is best applied to situations in which

- The interactions between the agents are complex, nonlinear, discontinuous, or discrete;
- Space is crucial and the agents' positions are not fixed;
- The population is heterogeneous, when each individual is (potentially) different;
- The topology of the interactions is heterogeneous and complex; and
- The agents exhibit complex behavior, including learning and adaptation.

Parunak et al. (1998) describe strengths of ABM as including:

- The ability to define both physical space and "interaction space"—that is, the ability of agents to interact across a distance via electronic or other communication.

- Validation at two levels—are the observables consistent with our experience and are the individual behaviors reasonable?
- Ease of experimentation—users can "think directly in terms of familiar … processes, rather than having to translate them into equations relating observables."
- Ease of implementation of changes in practice—if behavior modifications are determined to produce a desirable outcome, they can be translated into real-world strategies.

In summary, ABM allows the user to develop a deeper understanding of the causes underlying system-level behavior by examining the emergent behavior of the complex systems model.

Weaknesses of Agent-Based Modeling

The advantages described above notwithstanding, like other advanced computing paradigms, ABM is not a magic bullet.

Agent-based models require significant quantities of data and can be computationally intensive. While neither of these is unique to ABM, agent-based models often require behavioral data, which may be difficult and/or costly to obtain, and the simulation of individual agents requires significant computational power. In addition, it is often unclear how to validate such a model, particularly if the goal is prediction of behavior in an untested system. For example, if one is modeling lane changing behavior on a highway, the model results can be compared with observed results. However, if one is predicting passenger behavior in the presence of new information, there are no observed results to which the model results can be compared.

Bonabeau (2002a) warns that, as with any model, an ABM must be designed at an appropriate level of detail. Further, because many of the agent characteristics are "soft," outcomes are more often appropriately viewed at a qualitative rather than a quantitative level. Finally, ABM tends to be time-consuming and resource-intensive because it is working at the component level and simulating the interactions among many agents rather than scripting macroscopic behavior.

In their review of transportation applications of ABM, Kikuchi et al. (2002) describe the fact that "…the individual agents do not make the globally optimal decisions" as a limitation of ABM. It is a limitation if we are trying to determine a globally optimal state. If, however, we are trying to gain a better understanding of the system dynamics (in order to identify appropriate intervention points, for example), this becomes a strength rather than a limitation.

AGENT-BASED MODELING APPLIED TO TRANSPORTATION PROBLEMS

ABM has been applied to a variety of transportation problems. In 2002, *Transportation Research C* published a special issue on "Intelligent Agents in Traffic and Transportation." The volume comprises nine papers, each detailing applications of agents in transportation. Four of the papers deal explicitly with ABM, or simulation, while the others address a variety of other agent applications.

In addition, a literature review by Kikuchi et al. (2002) showed that most transportation-related applications of ABM were to vehicle or pedestrian flow. Additional applications cited included ITS, aircraft arrivals at an airport, travel behavior, vehicle routing, and land use

patterns. More recently, an informal review showed that the most common applications described in the literature are traffic or pedestrian simulation and demand modeling efforts.

Microscopic traffic simulation is becoming an increasingly important analysis tool. One of the only ways to test new operational schema, such as those proposed in ITS initiatives, is through simulation. Agent-based models lend themselves particularly well to microscopic simulation because they are disaggregate and can account for human behavior. In fact, Hidas (2002) observes that "most microscopic simulation models—at least those developed in an object-oriented framework—would correspond to the specifications of multi-agent systems."

A number of researchers have explored ABM for traffic simulation. Dia (2002) applied ABM to analyze the effects of advanced traveler information systems on traffic congestion to evaluate the impacts of providing drivers with travel information. He applied discrete choice models to behavioral data to determine agent characteristics, and the agents' route-choice behavior was based on these frameworks. The model yielded promising results for more sophisticated ABMs in this domain. Wahle et al. (2002) also studied the effects of traveler information on driver choices as it ultimately affected traffic congestion. In their model, agents operate at a tactical layer, which controls the "task of driving," and a strategic layer, which controls route choice. Agents choose between two routes and are either "static," meaning they maintain their preferred route regardless of travel time information provided, or "dynamic," meaning they choose the route with the lower travel time. The simulation showed that, when provided with travel time information and two route choices, "the concentration of drivers on the recommended routes is intrinsic to many systems and leads to a negative impact on the traffic pattern, e.g., oscillations."

In addition to route choice, researchers have used ABM to model lane-changing and car-following behaviors. Hidas (2002) describes Simulation of Intelligent TRAnsport Systems (SITRAS), an agent-based traffic simulation. SITRAS is designed to provide a test-bed for ITS technologies to evaluate situations, such as driver response to incidents, which are not feasibly tested in the real world. He describes an agent-based approach to modeling lane changing behavior in congested conditions. Specifically, the model demonstrates that simulation of forced lane changing behavior is required for a realistic representation of traffic when an incident occurs. Peeta et al. (2004) model the interaction between cars and trucks to examine the effects of changes in car-following and lane-changing behavior on aggregate flow. They use the results of stated preference surveys to characterize the behavior of non-truck drivers in the vicinity of trucks. The model allowed the authors to investigate the effects of mitigation strategies for car-truck interactions, and they conclude that for the best combination of safety, traffic flow, and car-truck interactions, restricting trucks to the right lane is preferred. Louisell et al. (2006) model driver behavior approaching a work zone, distinguishing between driver interactions with the roadway and driver interactions with other drivers. The latter is modeled with strategic game theory. Inputs include driver behavior characteristics and work-zone configuration; the output is how these interactions generate micro-shockwaves and how they affect recovery.

Two of the more widely known agent-based models for traffic simulation are TRANSIMS (http://transims.tsasa.lanl.gov/) and MATSIM (http://www.matsim.org/). TRANSIMS was developed by Los Alamos National Labs with funding from a variety of government agencies. It is an agent-based model for large metropolitan areas that "is designed to give transportation planners more accurate, complete information on traffic impacts, energy consumption, traffic congestion, land use planning, traffic safety, intelligent vehicle efficiencies, and emergency evacuation." Rilett (2001) notes that TRANSIMS requires large amounts of data

as it models the large-scale dynamics of a system. Henson and Goulias (2006), in a bit of a shift, explore how TRANSIMS might be used for Homeland Security modeling applications, and they also conclude that the key for this type of modeling is large quantities of detailed data. Finally, Balmer et al. (2006) describe MATSIM, a more recently developed large-scale agent-based traffic simulator. They note similarities to TRANSIMS and other activity-based demand generation packages. However, they also identify differences; for example, MATSIM uses XML file formats in contrast to TRANSIMS flat files, and MATSIM runs more quickly than TRANSIMS because its traffic flow simulation is more simplified.

Several researchers have applied ABM to pedestrian simulation. Markowski and Kikuchi (2004) propose a framework for modeling pedestrian flow that is based on hierarchical "groups," such as a single person, a family, or several colleagues. This is in contrast to models that are based on individual pedestrians of particular types. Object-oriented structures are used, and rules are defined, the most important of which determines velocity. In addition, messages are passed between agents that can influence their behavior. Kukla et al. (2001) developed an ABM that "reliably mimics the movement decisions of pedestrians negotiating the walking environment." It includes a set of rules that applies to all agents, but the parameters differ among agents. The rules are derived from video recordings of real-life pedestrian movement.

Travel demand modeling is another area that has generated significant interest in ABM. For example, Devisch et al. have developed a residential choice model for use in integrated land use/transportation models. The agents are buyers and sellers, each of which acts "strategically based on their beliefs under conditions of uncertainty to achieve the best price." More traditionally, Zhang (2006) developed a route-choice model to explore route choice decisions made in real life. He notes that "normative assumptions, such as perfect information and unlimited human abilities to maximize utility, may produce serious prediction biases." His model is based on individual behavior and takes into account users having imperfect information. The output is network flow patterns. He points out that because the model is based on behavior, the behavioral rules require empirical data for both development and validation. Zhang and Levinson (2004) describe an agent-based travel demand model for trip distribution and route assignment. They define three types of agents: nodes, arcs, and travelers. The agents communicate with one another—for example, arcs inform travelers about travel times, and nodes provide information about adjacent nodes. Agents also share information about shortest paths and therefore learn from one another. The model is demonstrated on a simple grid network and also on the Chicago area sketch network. For the latter, "the trip length distribution [is] reasonably close to the observed one and [the model] assigns most traffic to the shortest routes." Rossetti et al. (2002) have extended the DRACULA (Dynamic Route Assignment Combining User Learning and microsimulAtion) model, an existing microscopic simulation model, to include intelligent agents that choose their departure times and routes.

The concept of agents has been used in transportation not just for modeling but also for control. Software agents are designed to function autonomously and control tasks. For example, Choy et al. (2003) and Manikonda et al. (2001) use software agents to optimize traffic signal timing.

GUIDELINES FOR AGENT-BASED MODELING IN TRANSPORTATION

Bonabeau (2002a) suggests that ABM is appropriate

- When there is potential for emergent phenomena—that is, individual behavior is nonlinear and changes over time, and fluctuations are more important than averages;
- When describing the system from the perspective of its constituent units' activities is more natural—that is, "activities are a more natural way of describing the system than processes"; and
- When the appropriate level of description or complexity is not known ahead of time.

As in all modeling initiatives, it is important first to define the problem clearly and second, to determine whether it is appropriate for an agent-based approach. If an agent-based approach is appropriate, the modeler must define agents, their interaction rules, and their environment, as well as select a modeling tool and/or programming language. In defining agents, the modeler needs to determine:

- The types of agents (e.g. drivers, traffic signals);
- The attributes of the agents;
- The allowable values for the attributes; and
- The initial values for the attributes.

Once agents are defined, the modeler needs to specify interaction rules for the agents. Agents will interact with one another and with their environment. Because the results will often need to be validated intuitively (at least in the short term), experts recommend keeping the rules as simple as possible, at least initially. This facilitates a "reality check" in which the modelers can quickly assess whether the results seem to make sense, given the rules, as well as the opportunity to focus on the resulting complex system-level behavior.

An agent-based model can be programmed in the developer's programming language or software package of choice. While programming from scratch in C, C++, Java, or any other language is possible, there are several modeling tools that can minimize the programming effort required. These modeling tools range from free and widely available to proprietary. Tobias and Hofmann (2004) compared several of the freely available tools that had been designed for social-science simulation. They concluded that, of the four tools reviewed (RePast, Swarm, Quicksilver, and VSEit), RePast (http://repast.sourceforge.net/) was best suited for this type of modeling and had the potential to save modelers significant programming time and effort. Other available tools include the SimAgent Toolkit (http://www.cs.bham.ac.uk/research/projects/poplog/packages/simagent.html) and ABLE (http://www.alphaworks.ibm.com/tech/able).

ADVANCED TOPOLOGIES

One of the things that makes the behavior of agent-based models so interesting is that agents typically have the ability to learn. This can be accomplished in a variety of ways, including through explicit knowledge updating rules. However, agents can use any advanced computing technique that facilitates learning, including artificial NNs (ANNs) and GAs. In fact, Sadek et al.

(2003) note that "CI [computational intelligence] is the study of the design of intelligent agents, in which an agent is regarded as something that acts in an environment. … Commonly used techniques to implement intelligent agents in CI include ANNs, fuzzy logic, KBS, CBR, and probabilistic reasoning (including genetic algorithms)."

CONCLUSIONS

As demonstrated by the results of research in many different aspects of transportation systems, ABM provides a reasonable and promising approach to modeling transportation phenomena. Specifically, it is a method for bottom-up modeling in which individual behavior, which is often well understood, yields system-level emergent behavior. The system-level behavior may also be easily observable, but the analytical links between the individual and system-level behavior are not; therein lies the contribution of ABM.

ACKNOWLEDGMENTS

The author gratefully acknowledges the assistance of Lafayette College students Christine Moore and Martin Tjioe in preparing this section of the circular, as well as the helpful suggestions of Shinya Kikuchi and Jeffrey Pfaffman. However, any errors or misstatements are the author's.

REFERENCES

Axelrod, R. *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*. Princeton University Press, 1997.

Axelrod, R., and L. Tesfatsion. A Guide for Newcomers to Agent-Based Modeling in the Social Sciences. In *Handbook of Computational Economics, Vol. 2: Agent-Based Computational Economics* (L. Tesfatsion and K. L. Judd, eds.). Handbooks in Economics Series, Elsevier/North-Holland, Amsterdam, Netherlands. 2006.

Balmer, M., K.W. Axhausen, and K. Nagel. An Agent-Based Demand-Modeling Framework for Large Scale Micro-Simulations. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Bonabeau, E. Agent-Based Modeling: Methods and Techniques for Simulating Human Systems. PNAS, Vol. 99, No. 3, 2002a.

Bonabeau, E. Predicting the Unpredictable. *Harvard Business Review*. March 2002b, pp. 109–116.

Choy, M. C., R. L. Cheu, D. Srinivasan, and F. Logi. Real-Time Coordinated Signal Control Through Use of Agents with Online Reinforcement Learning. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1836*, Transportation Research Board of the National Academies, Washington, D.C., 2003, pp. 64–75.

Devisch, O., T. Arentze, A. Borgers, and H. Timmermans. Bi-Level Negotiation Protocal for Multi-Agent Simulation of Housing Transactions and Housing Market Clearing Processes. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Dia, H. An Agent-Based Approach to Modeling Driver Route Choice Behavior Under the Influence of Real-Time Information. *Transportation Research Part C*, Vol. 10, 2002, pp. 331–349.

Henson, K., and K. Goulias. A Preliminary Assessment of Activity Analysis and Modeling for Homeland Security Applications. Presented at 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Hidas, P. Modeling Lane Changing and Merging in Microscopic Traffic Simulation. *Transportation Research Part C*, Vol. 10, 2002, pp. 351–371.

Holland, J. The Global Economy as an Adaptive Process. *The Economy As a an Evolving Complex System* (P. W. Anderson, K. J. Arrow, and D. Pines, eds.). Addison Wesley, Redwood City, Calif., 1988.

Kikuchi, S., J. Rhee, and D. Teodorovic. Applicability of an Agent-Based Modeling Concept to Modeling of Transportation Phenomena. *Yugoslav Journal of Operations Research*, Vol. 12, No. 2, 2002.

Kukla, R., J. Kerridge, A. Willis, and J. Hime. PEDFLOW: Development of an Autonomous Agent Model of Pedestrian Flow. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1774,* Transportation Research Board of the National Academies, Washington, D.C., 2001, pp. 11–77.

Louisell, C., J. Collura, M. Knodler, and K. Heaslip. A Simple Algorithm for Quantifying the Impact of Driver Behavior on Traffic Flow and Safety During Forced Merges in Work Zones. Presented at the 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Manikonda, V., R. Levy, G. Stapathy, D. Lovell, P. C. Chang, and A. Teittinen. Autonoumous Agents for Traffic Simulation and Control. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1774*, Transportation Research Board of the National Academies, Washington, D.C., 2001, pp. 1–10.

Markowski, M., and S. Kikuchi. Simulating Pedestrian Interactions. *Information Technology in Civil Engineering 2004*, ASCE, 2004, pp. 52–56.

Parunak, H. V. D., R. Savit, and R. L. Riolo. Agent-Based Modeling Versus Equation-Based Modeling: A Case Study and Users' Guide. *Proceedings of Multi-Agent Systems and Agent-Based Simulation*, Springer, 1998, pp. 10–25.

Peeta, S., W. Zhou, and P. Zhang. Modeling and Mitigation of Car-Truck Interactions on Freeways. In *Transportation Research Record: Journal of the Transportation Research Board, No, 1899*, Transportation Research Board of the National Academies, Washington, D.C., 2004, pp. 117–126.

Rilett, L. R. Transportation Planning and TRANSIMS Microsimulation Model. *Transportation Research Record: Journal of the Transportation Research Board, No. 1777*, Transportation Research Board of the National Academies, Washington, D.C., 2001, pp. 84–92.

Rossetti, R. J. F., R. H. Bordini, A. L. C. Bazzan, S. Bampi, R. Liu, and D. V. Vliet. Using BDI Agents to Improve Driver Modeling in a Commuter Scenario. *Transportation Research Part C*, Vol. 10, 2002, pp. 373–398.

Sadek, A. W., G. Spring, and B. L. Smith. Toward More Effective Transportation Applications of Computational Intelligence Paradigms. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1836,* Transportation Research Board of the National Academies, Washington, D.C., 2003, pp. 57–63.

Tobias, R., and C. Hofmann. Evaluation of Free Java-Libraries for Social-Scientific Agent-Based Simulation**.** *Journal of Artificial Societies and Social Simulation*, Vol. 7, No. 1, 2003. http://jasss.soc.surrey.ac.uk/7/1/6.html.

Toroczkai, Z., and S. Eubank. Agent-Based Modeling as a Decision-Making Tool. *The Bridge*, Vol. 35, No. 4, 2005.

TRANSIMS. 2005. http://transims.tsasa.lanl.gov/.

Wahle, J., A. L. C. Bazzan, F. Klugl, and M. Schrekenberg. The Impact of Real-Time Information in a Two-Route Scenario Using Agent-Based Simulation. *Transportation Research Part C*, Vol. 10, 2002, pp. 399–417.

Zhang, L. An Agent-Based Behavioral Model of Spatial Learning and Route Choice. Presented at 85th Annual Meeting of the Transportation Research Board, Washington, D.C., 2006.

Zhang, L., and D. Levinson. Agent-Based Approach to Travel Demand Modeling. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1898,* Transportation Research Board of the National Academies, Washington, D.C., 2004, pp. 28–36.

THE NATIONAL ACADEMIES

Advisers to the Nation on Science, Engineering, and Medicine

The **National Academy of Sciences** is a private, nonprofit, self-perpetuating society of distinguished scholars engaged in scientific and engineering research, dedicated to the furtherance of science and technology and to their use for the general welfare. On the authority of the charter granted to it by the Congress in 1863, the Academy has a mandate that requires it to advise the federal government on scientific and technical matters. Dr. Ralph J. Cicerone is president of the National Academy of Sciences.

The **National Academy of Engineering** was established in 1964, under the charter of the National Academy of Sciences, as a parallel organization of outstanding engineers. It is autonomous in its administration and in the selection of its members, sharing with the National Academy of Sciences the responsibility for advising the federal government. The National Academy of Engineering also sponsors engineering programs aimed at meeting national needs, encourages education and research, and recognizes the superior achievements of engineers. Dr. William A. Wulf is president of the National Academy of Engineering.

The **Institute of Medicine** was established in 1970 by the National Academy of Sciences to secure the services of eminent members of appropriate professions in the examination of policy matters pertaining to the health of the public. The Institute acts under the responsibility given to the National Academy of Sciences by its congressional charter to be an adviser to the federal government and, on its own initiative, to identify issues of medical care, research, and education. Dr. Harvey V. Fineberg is president of the Institute of Medicine.

The **National Research Council** was organized by the National Academy of Sciences in 1916 to associate the broad community of science and technology with the Academy's purposes of furthering knowledge and advising the federal government. Functioning in accordance with general policies determined by the Academy, the Council has become the principal operating agency of both the National Academy of Sciences and the National Academy of Engineering in providing services to the government, the public, and the scientific and engineering communities. The Council is administered jointly by both the Academies and the Institute of Medicine. Dr. Ralph J. Cicerone and Dr. William A. Wulf are chair and vice chair, respectively, of the National Research Council.

The **Transportation Research Board** is a division of the National Research Council, which serves the National Academy of Sciences and the National Academy of Engineering. The Board's mission is to promote innovation and progress in transportation through research. In an objective and interdisciplinary setting, the Board facilitates the sharing of information on transportation practice and policy by researchers and practitioners; stimulates research and offers research management services that promote technical excellence; provides expert advice on transportation policy and programs; and disseminates research results broadly and encourages their implementation. The Board's varied activities annually engage more than 5,000 engineers, scientists, and other transportation researchers and practitioners from the public and private sectors and academia, all of whom contribute their expertise in the public interest. The program is supported by state transportation departments, federal agencies including the component administrations of the U.S. Department of Transportation, and other organizations and individuals interested in the development of transportation.
www.TRB.org

www.national-academies.org

TRB
TRANSPORTATION RESEARCH BOARD
500 Fifth Street, NW
Washington, DC 20001

THE NATIONAL ACADEMIES™
Advisers to the Nation on Science, Engineering, and Medicine

The nation turns to the National Academies—National Academy of Sciences, National Academy of Engineering, Institute of Medicine, and National Research Council—for independent, objective advice on issues that affect people's lives worldwide.
www.national-academies.org