Proceedings of the 1989 Winter Simulation Conference
E.A. MacNair, K.J. Musselman, P. Heidelberger (eds.)

TUTORIAL: ARTIFICIAL INTELLIGENCE AND SIMULATION

Jeff Rothenberg
The RAND Corporation
1700 Main Street
Santa Monica, CA 90406

ABSTRACT

The term "simulation" is often interpreted quite narrowly: as a way of making predictions by running a behavioral model to answer questions of the form *"What-if...?"*. The major impact of Artificial Intelligence (AI) research on simulation is to encourage the use of additional kinds of modeling based on inferencing, reasoning, search methods, and representations that have been developed in AI. This natural—though long overdue—extension of simulation can produce behavioral models that answer questions *Beyond "What-if...?"*. The result is sometimes referred to as "Knowledge-Based Simulation". This tutorial presents some of the major concepts of Artificial Intelligence and illustrates their applicability to simulation using examples drawn from recent Knowledge-Based Simulation research. It focuses on the present state-of-the-art, current problems and limitations, and future directions and possibilities.

1. A BRIEF OVERVIEW OF AI

Artificial Intelligence (AI) has defined many of the frontiers of computer science since the 1950s (Klahr and Waterman 1986). It is a vast, loosely defined area encompassing various aspects of pattern recognition and image processing, natural language and speech processing, robotics, symbolic computation, automated reasoning, expert systems, neural nets, and a host of other disciplines.

Throughout its history, AI has been concerned with problems whose solution seemed impossible using conventional computer science. This attempt to make computers intelligent has two distinct motivations, referred to here as *modeling* and *engineering*. The modeling approach seeks to model the way humans (or other intelligent beings) perform tasks that require intelligence: It attempts to identify problems that require intelligence and to elucidate the mechanisms we employ in our own solutions of those problems. The engineering approach, in contrast, is concerned with producing systems that solve useful problems, regardless of whether those problems require intelligence or their solutions involve mechanisms parallel to our own. In practice these two approaches are often merged or even confused; but the distinction is useful for understanding the different emphasis of different AI efforts and the various roles of AI in modeling (and modeling in AI).

The modeling approach to AI has a psychological or philosophical premise: can we use computers to build models of how we believe intelligence works? That is, given a conceptual theory of intelligence, can we embody that theory in a computer model? Computer models ideally make such theories concrete, allowing them to be tested, validated, and refined much more effectively than if they remained purely conceptual. The modeling approach to AI therefore views the implementation of computerized models as a primary technique for advancing our understanding of intelligence. In addition, the implemented models often suggest novel mechanisms that may in turn become part of new conceptual theories. For example, it is this kind of "metaphor feedback" that has led to the popular conception of the brain as a computer. Insights gained from AI models (both from their failures and their successes) have contributed to revisions of theories in areas ranging from linguistics to cognitive psychology.

The engineering approach to AI has a different premise: computers are not organisms, so why not use them to their own best advantage to try to solve useful problems, without worrying about whether they are solving them the way we would? This focuses on solving interesting and useful problems rather than on defining or understanding intelligence. In practice, this approach works

symbiotically with the modeling approach; if a model fails to work, engineering may suggest a solution. While such solutions are often *ad hoc*, and therefore unlikely to provide direct insight from a modeling point of view, they may nevertheless reveal fundamental flaws or alternative possibilities in the conceptual theory that produced the model, thereby suggesting revisions to the theory. Whenever the engineering approach succeeds in solving a problem (or even approaches success), its results tend to be appropriated by conventional computer science or engineering, so that AI often receives no credit for the eventual solution. This has occurred repeatedly in AI's history, contributing to the partially facetious adage that "AI never *solves* any problem". A more constructive interpretation of this phenomenon is that it represents successful technology transfer.

The lack of distinction between these two different motivations often leads to confusion about how AI research should be evaluated and judged. Ideally, research stemming from a modeling motivation should be judged according to the insight it produces into how natural intelligence works; mechanisms designed for such AI programs should be evaluated with respect to how closely they parallel and illuminate their corresponding biological mechanisms. The behavior of such programs should be judged according to how well they mimic the behavior of humans (or other intelligent entities), rather than how well they solve particular problems. In contrast, research stemming from an engineering motivation should be judged solely according to its problem-solving performance; mechanisms designed for these AI programs should be evaluated according to standard software engineering principles.

Unfortunately, since these two motivations tend to be combined and confused, AI programs often ask to be evaluated and judged by whichever criterion provides a more generous answer. Poor problem-solving performance is often excused on the basis that a program provides interesting modeling insights, whereas *ad hoc* models are often excused in the interests of performance. By adopting a somewhat schizophrenic personality, some AI programs simultaneously attempt to justify their poor performance on the basis of their modeling while justifying their *ad hoc* models on the basis of their performance. On the other hand, many AI programs have produced interesting modeling insights, many have achieved excellent problem-solving performance, and some have combined the two to a degree that entirely eliminates the need for excuses.

AI has made many contributions to computer science and software engineering. Often the problems that AI attacks are also attacked from other quarters of computer science, and it is not always easy to assign credit for the solutions that eventually emerge. AI has had at least some part in producing—or is currently attempting to produce—a number of advances that have direct bearing on simulation and modeling. These include the object-oriented programming paradigm, demons, dynamic planning, goal-directed heuristic search, spreading-activation search, taxonomic inference (as implemented by class/subclass, or "IS-A", inheritance hierarchies), forward and backward chaining, qualitative reasoning, truth maintenance, proof procedures for formal logic, simulated annealing, neural nets, and the representation of spatial and temporal phenomena, uncertainty, plans, goals, beliefs, and so-called "deep structures".

The following sections outline some of the most important areas of overlapping research and cross-fertilization between AI and simulation.

2. AI AND SIMULATION

In any discussion of AI and simulation, the term "simulation" must be freed from the confines of its own tradition, where it often denotes a very limited form of modeling. There is a strong tendency in simulation circles to view simulation narrowly as a way of making predictions by running an encoded behavioral model ("winding it up and letting it run") to answer "*What-if?*" questions. This may be thought of as the "toy duck" view of simulation (Rothenberg, et al. 1989).

A great deal of effort is required to encode the knowledge needed to build a simulation; having gone to all this trouble, one should attempt to derive the maximum benefit from this knowledge. In particular, in addition to "running" a simulation to answer "*What-if?*" questions, one should be able to utilize the full range of inferencing, reasoning, and search methods that are available in AI. These methods should be able to explain why a given

sequence of events occurred and answer definitive questions such as "Can this event ever happen?" and goal-directed questions such as "Which events might lead to this event?". This broad view of simulation is referred to as *Knowledge-Based Simulation.*

The major impact of AI on simulation is (or should be) to encourage simulation to make use of a wider range of modeling techniques: the result will still be a phenomenological model, but one that can take full advantage of additional techniques to answer a wider range of questions that are of interest to its users. This natural, long-overdue extension of simulation can be thought of as going *"Beyond What-if?"*.

Discrete-state simulation has derived great benefit from many of the techniques developed in AI. The object-oriented paradigm, which first appeared in Simula (Dahl and Nygaard 1966), owes its present state of refinement to AI language efforts like Smalltalk (Goldberg and Kay 1976) and ROSS (McArthur, Klahr, and Narain 1984). The object-oriented approach has many advantages, despite its shortcomings (Rothenberg 1986). For example, the appropriate use of inheritance hierarchies (or lattices) greatly simplifies the specification of a complex simulation, producing highly comprehensible models (Klahr 1985). Searching and planning techniques developed in AI have made feasible models that simulate the behavior of human decision makers in environments involving "command and control", while backward chaining can help answer questions about how to achieve a given result. Techniques for representing goals and beliefs have helped build simulations that can explain the behavior of simulated entities. Some of the current outstanding problems in discrete-state simulation, such as the problem of representing and computing continuous information like weather and terrain, may also yield to AI solutions.

Analytic simulation has tended to look to mathematics rather than AI for its methods, but here too AI offers some new approaches. One example is recent work at The RAND Corporation in sensitivity analysis (a sorely neglected problem in simulation), where AI techniques are used to represent and propagate sensitivity information through a computation, so that it need not be recomputed for every function call whenever some higher-level function is perturbed to probe its sensitivity to changes in its parameters. Similarly, symbolic algebra programs developed by AI, such as REDUCE (Hearn 1985), may allow applying expert algebraic manipulation to analytic functions within a simulation.

The relationship between AI and simulation is bilateral: AI has produced many systems that use models as sources of internal expertise. One of the earliest examples of this was Gelernter's Geometry Machine (Gelernter 1959), which embedded a model of a geometry student's diagram (itself a model), and used a virtual "diagram computer" to test hypotheses against this internal diagram. This has become a classic AI paradigm that expresses AI's recognition of the importance of models to intelligent agents: in seeking to model such agents, AI is naturally driven to model their use of models! In the case of the Geometry Machine, whose stated motivation was to solve problems generally considered to require intelligence, the engineering approach converged with the modeling approach in choosing a solution based on a model of how we ourselves solve geometry problems: being inveterate modelers, we use a model (i.e., a diagram).

Another classic example of an embedded model in an AI system is SOPHIE (Brown, Burton, and DeKleer 1982), which taught electronic circuit diagnosis by means of an interactive dialog (in English). In order to allow students to ask hypothetical questions such as "What would happen if I measured the voltage across points A and B?", SOPHIE used a simulator of the electronic circuit being diagnosed. This simulator was treated as a source of expertise about electronic circuits. The AI program that conducted the dialog with the student did not encode answers to all possible questions the user might ask; instead, it answered those questions by consulting its internal model, i.e., running its embedded simulation.

There is considerable evidence that in order to exhibit more than superficial intelligence, AI systems must make use of "deep structures", or models of reality like those described above. Simple action-response rules can produce programs that perform impressively up to a point, but beyond that point there is no escaping the need to give programs real "understanding" of the world, at least within their domains. There are many possible

approaches to providing such understanding, but they all essentially involve giving a program a model of its world that it can use to answer a wide range of unanticipated questions about that world.

3. KNOWLEDGE-BASED SIMULATION AT THE RAND CORPORATION

This section elaborates some of the areas discussed above, drawing on recent work in Knowledge-Based Simulation at The RAND Corporation. This work is representative of current research efforts that are attempting to blend AI and simulation.

Artificial intelligence and simulation have been major areas of research at RAND for many years (Klahr and Waterman 1986). The work of Newell, Shaw and Simon at RAND in the 1950s (Newell, Shaw, and Simon 1957) was one of AI's earliest successes and defined many areas that continue to be focal points for AI research. More recently RAND's research in expert systems produced the languages RITA (Anderson and Gillogly 1976; Anderson, et al. 1977) and ROSIE (Sowizral and Kipps 1985; Kipps, Florman, and Sowizral 1987) as well as several expert system applications, including LDS (Waterman and Peterson 1981), TATR (Callero, Waterman, and Kipps 1984) and SAL (Paul, Waterman, and Peterson 1986). Similarly, RAND's long history of simulation research produced the Simscript language (Kiviat, Villanueva, and Markowitz 1968) as well as both theoretical and experimental results in game theory and monte carlo simulation. RAND began applying AI to simulation in the late 1970s and early 1980s. The development of the object-oriented ROSS simulation language demonstrated the potential benefit of AI for simulation technology. The Knowledge-Based Simulation effort continued this tradition. The following describes this work as a way of highlighting some of the overlap (and potential overlap) between AI and simulation.

The goal of RAND's Knowledge-Based Simulation effort is to make simulations both more powerful and more comprehensible by (1) allowing modelers to build, validate, evolve and maintain more powerful and realistic simulations that model a wider range of relevant phenomena, and (2) allowing users to interact with these simulations in ways that provide deeper understanding of the phenomena being modeled. Making simulations more powerful requires extending the kinds of modeling they can perform and the kinds of questions they can answer (as discussed above). Making simulations more comprehensible requires developing techniques for *intelligent exploration and explanation*, which requires allowing users to modify both the model and the course of events in a simulation, and making the simulation explain its behavior in useful ways.

This research has involved a number of distinct tasks, the first of which is reasoning about simulation behavior. This ultimately includes being able to ask goal-directed questions, questions about whether or how an initial state can produce a desired result, questions about the possible values of variables in a simulation, questions about the interactions of objects or factors, questions about the goals of an object, and questions about why an object performed an action. The inability of current discrete-state simulations to answer such questions is the result of limitations in their representational and inferential capabilities stemming from the fact that knowledge is represented implicitly in procedural code and is therefore not amenable to inference. Support for reasoning requires representing the behavior of objects in ways that allow the use of automated reasoning techniques (like forward and backward chaining) and integrating these with other forms of inference, such as those based on the use of object taxonomies.

In addition to the explicit use of reasoning, it is important to allow implicit reasoning based on multiple relations. Complex simulations require the ability to represent multi-dimensional relationships among objects, such as "A is *a-kind-of* B", "A *is a-part-of* B", "A is *in-control-of* B", "A is *in-communication-with* B", or "A *is near* B". It is vital for the simulation user to be able to define relations freely, examine the state of the simulation in terms of these relations, and modify them dynamically. Most object-oriented systems support only minor variations of the "class-subclass" (also called "IS-A" or "taxonomy") relation along with a corresponding "inheritance" mechanism to maintain taxonomic relationships (i.e., specialized inferential support for the class/subclass relation). It is important to provide a true multiple relation environment in which different kinds of relations are supported by appropriate specialized inference mechanisms and to provide a general facility to

allow the simulation developer to define new relations with appropriate inferential support.

In order to be comprehensible to users, simulations must provide intelligent exploration and explanation. This should include graphic querying of the simulation state, being able to roll the simulation back to a previous state, change a parameter, and rerun the simulation, saving multiple simulation states for later analysis and comparison, being able to build or modify simulation scenarios graphically, and being able to build or modify simulation objects graphically (e.g., defining and exercising new behaviors graphically).

As mentioned above, sensitivity analysis is one of the great abandoned areas of simulation. Yet without it there is no guarantee that the results of a simulation might not be drastically different if some small change were made to some initial parameter. Sensitivity analysis is also important for indicating which parameter values are the most important to verify for a simulation to be valid and believable. The straightforward approach to sensitivity analysis requires running a simulation many times, perturbing individual parameters to see how the results differ. This is prohibitively expensive in most cases, as a consequence of which it is rarely done. Current RAND research seeks to provide a means of computationally feasible sensitivity analysis in a simulation environment, utilizing a new approach that propagates and combines the sensitivities of composite functions through a computation. This approach computes a representation of the sensitivity of each called function the first time it is executed, and propagates this sensitivity information rather than recomputing it each time it is needed.

Another major shortcoming of current simulation models is their inability to vary the level at which they are aggregated (also referred to as their "resolution"). It is generally necessary to choose a desired level of aggregation in advance and design a simulation around that level. Changing this level typically requires considerable reprogramming of the simulation; changing it under user control or dynamically is generally unthinkable. The fact that the level of aggregation of a model gets frozen in early in its design is a major impediment to the reusability of models and the utility of simulation in general. Users should be able to vary the level of aggregation of a simulation and to indicate which aspects of the model are of particular interest, running those aspects of the simulation disaggregated while running peripheral aspects at higher levels of aggregation. Users should also be able to run highly aggregated simulations to identify interesting cases and then examine those cases in more detail by rerunning them disaggregated. The goal is to develop a methodology for building simulations whose level of aggregation can be varied either statically or dynamically. This requires mechanisms for representing vertical slices of objects in an aggregation hierarchy and allowing interactions between objects at different levels of aggregation. It is also necessary to address problems of inconsistency that can arise between different levels: that is, running a simulation at an aggregated level should produce results that are consistent with the results of running the same simulation at a disaggregated level.

Modeling real-world environments that include human decision makers requires building simulations that embed models of intelligent agents possessing varying degrees of awareness, authority, initiative, intelligence, etc. This also requires hierarchical planning so that at each level, plans will be translated into objectives for agents at the next lower level.

Finally, there are a number of pseudo-objects or phenomena that are not modeled well by current object-oriented simulations. For example, terrain, roads, rivers, and weather defy easy representation by conventional object-oriented means. These pseudo-objects seem to require representations and manipulations that are different from those used for more compact objects, either because they traverse or interpenetrate other objects (without actually being part of them), or because they are best described by continuous models (such as partial differential equations). A number of AI techniques are being explored to represent such pseudo-objects and their interactions.

4. CONCLUSIONS

AI is a vast field that represents the leading edge of computer science research along many fronts. It can best be understood in terms of its two distinct motivations: the *modeling* motivation,

which seeks to model the cognitive processes of intelligence and the *engineering* motivation, which simply attempts to solve useful problems that cannot be solved by conventional means. AI has made many contributions to computer science and software engineering and has both produced and made use of many modeling concepts. The wedding of AI and simulation is still in progress; its consummation promises to be of great value to both fields of endeavor.

ACKNOWLEDGMENTS

The research described above at The RAND Corporation was supported by Defense Advanced Research Projects Agency Contract MDA903-85-C00030. Views and conclusions contained in this section are those of the author and should not be interpreted as representing the official opinion or policy of The RAND Corporation, DARPA, the U.S. government, or any other person or agency connected with them.

REFERENCES

Anderson, R. H., and Gillogly, J. J. (1976). *RAND Intelligent Terminal Agent (RITA): Design Philosophy*, The RAND Corporation, R-1809-ARPA.

Anderson, R. H., Gallegos, M., Gillogly, J. J., Greenberg, R. B., and Villanueva, R. V. (1977). *RITA Reference Manual*, The RAND Corporation, R-1808-ARPA.

Brown, J. S., Burton, R. R., and DeKleer, J. (1982). Pedagogical, Natural Language and Knowledge Engineering Techniques in SOPHIE I,II, and III. In: *Intelligent Tutoring Systems* (D. Sleeman, J. S. Brown, eds.). Academic Press, New York, 227-282.

Callero, M., Waterman, D. A., and Kipps, J. R. (1984). *TATR: A Prototype Expert System for Tactical Air Targeting*, The RAND Corporation, R-3096-ARPA.

Dahl, O-J. and Nygaard, K. (1966). Simula—An Algol-Based Simulation Language, *ACM Communications*, **9**, 671-678.

Gelernter, H. (1959). Realization of a geometry theorem-proving machine. *Proceedings of an International Conference on Information Processing*, Paris: UNESCO House, 273-282.

Goldberg, A. and Kay, A. (1976). *Smalltalk-72 Instruction Manual.* Report SSL 76-6, Xerox PARC, Palo Alto, California.

Hearn, A. C. (1985). *REDUCE User's Manual, Version 3.2*, The RAND Corporation, CP78.

Kipps, J. R., Florman, B. A., and Sowizral, H. A. (1987). *The New ROSIE Reference Manual and User's Guide*, The RAND Corporation, R-3448-DARPA/RC.

Kiviat, P., Villanueva, R., and Markowitz, H. (1968). *The SIMSCRIPT II Programming Language*, Prentice-Hall, Englewood Cliffs, NJ.

Klahr, P. (1985). Expressibility in ROSS: An Object-oriented Simulation System. In: *AI APPLIED TO SIMULATION: Proceedings of the European Conference at the University of Ghent*, 136–139.

Klahr, P. and Waterman, D. A. (1986). Artificial Intelligence: A Rand Perspective. In: *Expert Systems Techniques, Tools and Applications*, Addison-Wesley, 3–23.

McArthur, D., Klahr, P., and Narain, S. (1984). *ROSS: An Object-Oriented Language for Constructing Simulations*, The RAND Corporation, R-3160-AF.

Newell, A., Shaw, J. C., and Simon, H. (1957). Empirical Explorations with the Logic Theory Machine. In: *Proceedings of the Western Joint Computer Conference*, Institute of Radio Engineers, New York.

Paul, J., Waterman, D. A., and Peterson, M. A. (1986). SAL: An Expert System for Evaluating Asbestos Claims. In: *Proceedings of the First Australian Artificial Intelligence Congress*, Computerworld, Ltd., Melbourne.

Rothenberg, J. (1986). Object-oriented Simulation: Where Do We Go from Here? *Proceedings of the 1986 Winter Simulation Conference*, Washington, DC, 464–469.

Rothenberg, J., Narain, S., Steeb, R., Hefley, C., and Shapiro, N. Z. (1989). *Knowledge-Based Simulation: An Interim Report*, The RAND Corporation, N-2897-DARPA.

Sowizral, H. A. and Kipps, J. R. (1985). *ROSIE: A Programming Environment for Expert Systems*, The RAND Corporation, R-3246-ARPA.

Waterman, D. A. and Peterson, M. A. (1981). *Models of Legal Decisionmaking*, The RAND Corporation, R-2717-ICJ.

AUTHOR'S BIOGRAPHY

Mr. Rothenberg is a Senior Computer Scientist at The RAND Corporation who has worked extensively in knowledge-based simulation. He performed his graduate work in AI at the University of Wisconsin in the area of semantic nets for robotic applications. His recent work has been in the development of new formalisms for integrating object-oriented and event-oriented views of discrete-state simulation.

Jeff Rothenberg
The RAND Corporation
1700 Main Street
Santa Monica, CA 90406
(213) 393-0411