Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2007), *21*, 5–6. Printed in the USA.
Copyright © 2007 Cambridge University Press 0890-0604/07 $16.00
DOI: 10.1017/S0890060407070035

GUEST EDITORIAL

The AI in *AI EDAM*

WILLIAM P. BIRMINGHAM, Editor Emeritus, 1997–2001
Computer Science Department, Grove City College, Grove City, Pennsylvania, USA

1. INTRODUCTION

A natural question is why AI in design? Although the design applications written about in the journal vary widely, the common thread is that researchers use AI techniques to implement their ideas. The use of AI techniques for design applications, at least when *AI EDAM* was started, was partially a reaction against the predominant design methods based on some form of optimization. Knowledge-based techniques, particularly rule-based systems of various sorts, were very popular. One of the draws of these methods, I believe, was their ability to represent knowledge that is hard or awkward to represent in traditional optimization frameworks. This mirrors my experience: at the time, I was working in configuration with components that had a large number compatibility and resource constraints. Although many constraints could be represented in mixed integer linear programming systems, it was not easy to conceptualize, write, and most importantly, maintain the constraints in those systems.

This is more an explanation from the side of convenience: it was easier to build systems. Yet, there is something more: AI is fundamentally bound to computation—the science of computation—just like other fields of computer science. I acknowledge that there are philosophical issues that transcend computation, but, and particularly from a design perspective, AI is about computation. Because AI is bound to computation, the programs we write operate under the limits described by computer science theory.

Given this context of computation, I am pleased that today I rarely hear the comment "we don't know how to solve the problem, so we used AI technique X," where X is the technique of the day, be it rules, fuzzy logic, genetic algorithms, or agents. Unfortunately, this phrase was common in the AI and design community in the past. There is no magic in the world, and we cannot expect AI to magically solve problems we do not know how to solve.

Reprint requests to: William P. Birmingham, Computer Science Department, Grove City College, 100 Campus Drive, Hoyt 105, Grove City, PA 16127, USA. E-mail: wpbirmingham@gcc.edu

There is yet more to the "Why AI?" question: design can help us to understand AI and computation. In the 1980s, there was a great deal of interesting design work presented at conferences like AAAI and IJCAI, and there were AI researchers spending their time looking at design problems. This work was attractive to both AI researchers and design researchers.

From systems like R1 (McDermott, 1982) and VT (Marcus et al., 1985), design researchers learned about how constraints could be represented and manipulated. In addition, this research paved the way for the commercial configuration and part-selection systems that today have wide application.

AI researchers also learned a few things about computation by using design problems. The size of design problems, stemming partly from the large number of constraints and parts that must be considered, posed substantial software-engineering issues and motivated work in machine learning. More fundamentally, this exposed the enormous combinatorial problems inherent in many design problems. This, in turn, led to research into better algorithms for searching large spaces of design alternatives (e.g., backtracking and preprocessing).

The size of systems necessary to solve even small design problems could not be ignored by AI researchers. Thus, AI research moved into building and maintaining large knowledge bases. Design applications again played a major role. Machine learning (Mitchell et al., 1985) used electronic design, and knowledge acquisition used elevator design (Marcus et al., 1985). These applications helped researchers uncover the computational properties of their approaches.

Another example of the interplay between AI and design is multiagent systems. The idea of agency has a long history in AI, both as a model of intelligent behavior and as a way of organizing computation (e.g., a form of distributed computing with varying degrees of decentralization). During the 1990s, the tremendous growth of the Internet and Web spurred great interest among computer scientists in distributed computing. AI researchers, in particular, were very interested in developing scaleable models that captured the autonomy of computation on the Web while allowing resources to be exchanged to aid in computation. Economic, voting theoretic, distributed constraint solving, and distributed decision making models, to name a few, gave rise to powerful computational systems when agents have separate knowledge (which they are unable or unwilling to share) and preference structures (again, which are not shareable).

At this time, design researchers were interested mostly in problems that were solved by a single designer, usually in the context of larger design activity. Designers rarely work in isolation. They work with other designers, other organizations, and almost certainly with clients. Thus, the "single designer" model implicit in much of the design research at the time was limiting.

The advent of powerful multiagent systems marked a major change in the direction of design research. Wellman (1995), for example, demonstrated how decentralized mechanisms could solve design problems. More importantly, these systems captured the kinds of interaction that are typical of design processes. Groups of designers and organizations collaborate on large projects, bargain over budgets, and share design information, but typically not their knowledge. In addition, these systems had the opportunity to scale in a sustainable way: rather than creating enormous knowledge bases, agents representing this knowledge can be used as appropriate. This avoided the nasty scaling problems that dogged earlier design systems.

During this time, new ways of conceiving and extending old design problems were formulated. D'Ambrosio et al. (1996) found ways of distributing the configuration problem based on real organizational models from the automobile industry. Darr and Birmingham (2000) inspired by Ward et al. (1990) developed representations that naturally supported concurrent engineering using ideas of distributed agency. These systems, like those of Wellman and others, had well-developed and understood computation and mathematical properties.

Design spurred AI research by presenting real problems and pushing the limits of certain systems. For example, design problems motivated work in distributed constraint satisfaction, where there is a need to search spaces that are not shared among agents.

There are other areas of cooperation, such as constraint solving and grammars. I have highlighted a few of the ones in which I worked most closely.

2. CONCLUDING THOUGHTS

This process of give and take between design and AI has served both disciplines well. AI methods, particularly in the areas of machine learning and probabilistic reasoning, continue to have great promise for design research.

Our AI colleagues have done an excellent job in firming up the theoretical underpinnings of their work. I think it is important for design researchers to do the same. Design research does not typically involve the development of formal theory. We build software systems for either a new design method, a new application, or sometimes both. The evaluation of these systems typically consists of running some test cases; this type of evaluation is more anecdotal than empirical. Yet, for those areas where it is appropriate, formal analysis is essential if the *field* is to move forward. Without this, we are left with little to mark our progress.

Recall my earlier statement that AI is about computation: basic algorithmic analysis is still necessary to understand the quality of the systems we build. Without this analysis, it is hard to figure out what makes the difference: was it an important insight into design or a clever algorithm? Algorithmic analysis can yield important insights: we may find that simple computational methods go far for some hard problems or that combinatorial effects of a design problem indicate that the only practical approach is heuristic.

I look forward to the next 20 years of *AI EDAM*. I hope to see that the field has made substantial progress in understanding the computation necessary to solve design problems. I cannot wait to obtain a copy of a book of algorithms for design problems, similar to those we use in algorithms classes (e.g., Cormen et al., 2003).

ACKNOWLEDGMENTS

The community owes a debit of gratitude to Clive Dym for starting the journal and seeing it through its early years. Clive did a tremendous job. In addition, I want to acknowledge Dave Brown's efforts: the journal is in great shape. Finally, the CUP staff gave me great support during my tenure as Editor, especially Morrell Gillette: thank you!

REFERENCES

Cormen, T.H., Leiserson, C.E., Rivest, R.L., & Stein, C. (2003). *Introduction to Algorithms*, 2nd ed. New York: McGraw–Hill/MIT Press.

D'Ambrosio, J.G., Darr, T., & Birmingham, W.P. (1996). Hierarchical concurrent engineering. *Concurrent Engineering: Research and Applications 4(1)*, 47–57.

Darr, T.P., & Birmingham, W.P. (2000). Part-selection triptych: a problem definition, problem properties, problem-solving method. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 14(1)*, 39–51.

Marcus, S., McDermott, J., & Wang, T. (1985). Knowledge acquisition for constructive systems. *Proc. IJCAI-85*. San Mateo, CA: Morgan Kaufmann.

McDermott, J. (1982). R1: a rule-based configurer of computer systems. *Artificial Intelligence 19(2)*, 39–88.

Mitchell, T.M., Mahadevan, S., & Steinberg, L. (1985). Leap: a learning apprentice for VLSI design. *Proc. IJCAI-85*. San Mateo, CA: Morgan Kaufmann.

Ward, A.C., Lozano-Perez, T., & Seering, W. (1990). Extending the constraint propagation of intervals. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 4(1)*, 47–54.

Wellman, M.P. (1995). A computational market model for distributed configuration design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 9(2)*, 125–133.