ORSA Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Commentary—The Promise of OR is Helping to Solve Problems

Harvey J. Greenberg, Frederic H. Murphy,

ORSA Journal on Computing
Vol. 6, No. 3, Summer 1994

0899-1499/94/0603-0270 $01.25

COMMENTARY

The Promise of OR is Helping to Solve Problems

HARVEY J. GREENBERG / *Mathematics Department, University of Colorado at Denver, Denver, CO 80217-3364; Email: hgreenberg@cudnvr.denver.colorado.edu*

FREDERIC H. MURPHY / *Department of Management Sciences/Operations Management, Temple University, Philadelphia, PA 19122; Email: v5256e@vm.temple.edu*

When you were learning optimization, how many times did you draw pictures to help you understand what was happening? When you now obtain results from an optimization model, how many times do you draw sketches and form mental pictures of what the solution means? When the solution does not look right, how often do you use this process of visualization to help you debug the model or this instance of it?

Visualization is crucial to comprehension, and computer technology has made it possible to see what you want. What we want, and in what form, is what system design is about. Someone had to actually do this, and mere programming without careful design would not suffice. Someone did: Christopher Jones.

In his feature article, Chris is right on target by bringing together a massive amount of material on the subject of visualization and optimization, some of it his own. Reaching back to World War II operations research, Chris begins by reminding us that the original promise of OR was to solve problems. A "problem" is something like "How do I stock my warehouse?" rather than "How do I solve this mathematical program?" In this sense, solving a problem includes communicating results from the model that was used. In this view, better algorithms are only a part of what is needed, as they are a means, not an end. While no one would doubt the importance of good algorithms in optimization, most people who have used OR in the trenches would agree that the total time spent on problem solving is far greater than the computer time, and that the total computer time is far greater than the optimization time. The potential for greater success in problem solving, therefore, lies in helping people use new technology for deeper insights, not necessarily just helping the optimization component of the process.

To design a visualization system, we must return to basics and understand models conceptually. Several people have done this over the decades since the inception of OR. Traditional uses of diagramming on a blackboard are transferred to electronic media in various ways, such as with colors, shapes, and icons that enable deeper insights. More recently, influence diagrams have been developed for insights into how variables and relations influence outcomes, and these have been implemented in some software systems. These are only some of the examples of how recent advances in computer hardware and software have created new opportunities for visualization. The past decade has brought out visual interactive modeling by several authors, some in OR and some in other disciplines. This is what makes Chris' article so timely: the quest for deeper insights into optimization models and solutions through the use of visualization technology has advanced so much in recent years by professional communities that do not appear to communicate with each other that we need a clear understanding of how the approaches relate to each other and what needs to be done.

Visualization not only presents thoughts but also their connections with ideas. When designing algorithms, our concerns are the capabilities and limitations of computer hardware and software. With modeling we are concerned with cognition and learning. Although we compute, we are not computers. Computing is a learned process for us, not the fundamental behavior it is for machines. Consequently, the (sometimes great) divide between modelers and algorithm developers should not come as a surprise.

The simulation community moved in this direction about a decade ago, and current software systems have animation and other visualizations to aid comprehension and model management. This has more recently begun to happen in optimization, as revealed in the feature article. Perhaps one use of visualization was understated: to manage models. Large-scale mathematical programs often contain many thousands of equations and tens of thousands of variables, even millions. Sheer size, however, is not the measure of complexity. For example, a transportation problem is not considered complex even if it has 50,000 nodes (one equation per node) and 1,000,000 arcs (one variable per arc plus slacks and surpluses). Complexity is more elusive, and a model manager must consider not only the integrity of the data, but also the related validity of the model's structure. In this context, visualization tools have been available for decades, with improvements evidenced by what we can buy today. It is the foundation for visualization that is more recent, and new results are enabling much more effective model management aids (for example, see [6]).

For our knowledge of modeling to develop in useful directions, we have to learn more about how humans see, learn, and comprehend. This is a different paradigm from discrete mathematics or real analysis. We have to look to psychologists who have studied how people process information. For example, one of the chief presentation concerns is avoiding clutter. Clutter is a meaningless concept to a computer; however, it relates to the way we *chunk* information, where the meaning of a chunk differs by individuals. To a young child the sentence, "See Spot run." might contain three chunks: one chunk = one word. To a younger child, it might be 10 (or 12) chunks: one chunk = one letter (or space). To many adults this sentence is one chunk.

We can remember 7 ± 2 chunks at one time for processing,[10] and it is no coincidence that large linear programs have about five nonzeroes per activity. We function by creating concepts, so that we can have a hierarchy where, for a situation, we store about seven concepts, then form a structure for holding the myriad of details that we need to know. The computer process of zooming corresponds to retrieval of detail, and this is part of Chris' vision.

Visualization must consider content as well as form. By this we mean the cognitive framework of the analyst must be taken into account, not just the use of computer graphics. We previously commented on this at length.[8] In that paper, we considered the difference between the traditional language, OMNI, and the algebraic language, GAMS.[7] It is no accident that OMNI is organized around processes because its initial clientele were chemical engineers, who think in terms of processes, such as refining. By contrast, GAMS was developed at the World Bank, and its initial clientele were economists, who think naturally in terms of equations. The two cognitive frameworks are different, and visualization of a model's structure needs to reflect such differences in conceptualizations.

As suggested by Chris' 270 references, the field of visualization has been very active, and one must go to the numerous papers and books to get the details. This feature article succinctly provides a map to aid the process of learning more about the pieces and how they fit into a big picture. There are some developments in other disciplines, however, that should be mentioned.

Besides the vast literature in computer science, there are relevant research activities in artificial intelligence, cognitive psychology, statistics, and simulation, to name just a few. Bobrow[2] presents a collection of papers that has influenced a line of research in the artificial intelligence community. In the related field of cognitive psychology, recent attention has focused on the effect differences in cognitive structures have on solving mathematically equivalent problems.[9] Cleveland and McGill[5] present a collection of papers by the statistics community on how to present visualization of highly multi-dimensional data. Nance and Overstreet[11] show how certain digraphs aid diagnosis of a simulation model and its results. Although these papers are from professional communities outside of OR and are not motivated by optimization applications, the methods and principles employed in these papers do apply to visualization in optimization (see [8] for elaboration).

What is exciting are the opportunities to carry out more research and apply visualization to important OR problems. One area not covered entirely by the feature article pertains to visualization of time. One use of new visualization technology is animation, which Chris himself is developing for sensitivity analysis.

In addition to the usual sensitivity analysis, animation enables us to go beyond a point solution and literally see the pathway to an optimum. Such visualization has been dubbed "algorithm animation,"[4] but it can be used to gain other insights by operating in the problem domain. Imagine, for example, a map on your screen with icons that represent regional production or process submodels. The icons are color-coded to represent levels of production or processing, respectively. Links appear with thickness proportional to the level of flow.

Suppose the underlying model is a linear program, and we use the simplex method to solve it. It is often useful to think of the iterations as a simulation of the market with instantaneous changes in activity levels as responses to changes in prices (caused by changing the basis). By watching colors and thicknesses change, we can visualize the market dynamics en route to its equilibrium. When it stops, we have a greater sense of *why* the solution is optimal, instead of being shown just the final point. The same animation can apply to nonlinear and integer optimization problems. In combinatorial portions of a mathematical program, such as capacity expansion, we can visualize plants appearing and disappearing as the algorithm proceeds towards an optimum.

Chris suggests that the principle behind hypertext is another exciting frontier. So far, hypertext implementations (for example, see [1, 3]) have been used as a way to organize information that makes navigation easier. Its potential for aiding visualization is not as well understood, but the principle behind hypertext can be a key to the kind of integration needed to support a multi-view environment for modeling and analysis. One kind of integration, for which the hypertext principle applies, is cross indexing the structure of a model's ingredients: objects, relations among objects, and attributes of the objects and relations. This enables a variety of visualizations, such as how one model differs from another.

Another frontier is virtual reality. There are people in other disciplines that are exploring ways to use this new technology to assist modeling, which extends to analysis support. Exactly how this might apply to the kind of optimization models we build in operations research has not yet emerged, but we think this will happen before the end of the century.

With no pun intended, Chris Jones' article is visionary. It draws not only from a vast literature, but also from the author's own experiences that range from conceptual (notably, graph grammars) to implementation (notably, MIMI graphics). The article is inspirational, as the advancement of visualization will undoubtedly contribute to the promise of OR.

References

1. H.K. Bhargava, M. Bieber and S.O. Kimbrough, 1988. Oona, Max and the WYWWYWI Principle: Generalized Hypertext and Model Management, *Proceedings of the Ninth International Conference on Information Systems*, 179–191.
2. D.G. Bobrow (ed.), 1984. *Qualitative Reasoning About Physical Systems*, MIT Press, Cambridge, MA.
3. G.H. Bradley, 1989. Mathematical Programming Modeling Project—Overview, in *Impacts of Recent Computer Advances on Operations Research*, R. Sharda, B.L. Golden, E. Wasil, O. Balci and W. Stewart (eds.), North-Holland, Amsterdam, 447–462.
4. M.H. Brown, 1988. *Algorithm Animation*, MIT Press, Cambridge, MA.

5. W.S. CLEVELAND and M.E. MCGILL (eds.), 1988. *Dynamic Graphics for Statistics,* Wadsworth & Brooks/Cole, Pacific Grove, CA.
6. H.J. GREENBERG, 1993. *A Computer-Assisted Analysis System for Mathematical Programming Models and Solutions: A User's Guide for ANALYZE,* Kluwer, Boston, MA.
7. H.J. GREENBERG and F.H. MURPHY, 1992. A Comparison of Mathematical Programming Modeling Systems, *Annals of Operations Research 38,* 177–238.
8. H.J. GREENBERG and F.H. MURPHY, 1994. Views of Mathematical Programming Models and Their Instances, *Decision Support Systems,* to appear.
9. A. LESGOLD, 1988. Problem Solving, Chapter 7 in *The Psychology of Human Thought,* R.J. Sternberg and E.E. Smith (eds.), Cambridge University Press, New York, 188–213.
10. G.A. MILLER, 1956. The Magic Number Seven Plus or Minus Two: Some Limits on Our Capacity for Processing Information, *Psychological Review 63,* 81–97.
11. R.E. NANCE and C.M. OVERSTREET, 1988. Diagnostic Assistance Using Digraph Representations of Discrete Event Simulation Model Specifications, *Transactions of The Society for Computer Simulation 4:1,* 33–57.