Teaching MIP Modeling and Solving

BY
TOBIAS ACHTERBERG, MARTIN GRÖTSCHEL AND THORSTEN KOCH

Dear reader, do you know what the maximum sizes of linear programs are that can be solved at present in acceptable running time? What about instances of integer or mixed integer programming? What about special combinatorial optimization problems such as the min-cost flow, the max-cut or the traveling salesman problem? If a colleague, customer or student asks, what would you answer?

It goes without saying that an O.R./math/CS/engineering student, studying whatever branch of optimization, needs a thorough education in the related theory, a good understanding of the major algorithm and some experience in the implementation of such methods. Most curricula take care of that appropriately. But what about real problem solving? Do we teach the "real thing" here?

Many of us still employ standard computational textbook exercises, some use spreadsheet examples, more advanced classes utilize student versions of commercial optimization software. Is this really what (good) O.R. students should learn? Do we provide them with the right picture of *modeling and solving problems from industry*? Do they know what can be achieved at present computationally?

Today, competition, innovation, misconceptions, data modifications and planning or execution mistakes force companies to reconsider plans made in very short time frames. The complexity of the business environment rules out "manual trial and error." This (nevertheless still existing) approach is mostly substituted by various (fast and frugal) heuristics. Why not try exact methods, e.g., when the models turn out to be mixed integer programs (MIPs)? Typical answers are: too messy, too cumbersome, takes too long, won't work.

These answers have been true up to nearly a decade ago. But since then the MIP landscape has changed completely. In quite a number of relevant large-scale instances, MIP codes can provide provably optimal or almost-optimal solutions. This capability leads to new desires. Models, e.g., in supply chain management, are made bigger and more accurate, models that have been treated hierarchically or separately in transportation/logistics are merged into one, business planning horizons are increased, etc. Modeling alternatives are now tested in depth.

Empower Students

All these aspects combined may lead to models that are of extreme sizes – again out of reach for today's codes and machines. But now modeling skills based on a deeper understanding of mixed integer programming theory and computational experience may help. And this may be the (high tech) competitive advantage a company is looking for. How can this be taught and learned?

Our answer is: Students must have access to the full power of the current solution technology and they must be able to "play" with the data and codes. Student versions of commercial software seldom allow the treatment of large-scale instances. These codes cannot be modified, and new ideas can rarely be tested. That is why we believe that source code for those interested in this kind of endeavor should be made available, and in addition, students need data of large-scale practical instances where modeling pitfalls can be experienced.

Problem solving in practice requires the execution of three basic steps: modeling, solver run and solution analysis. These three steps have to be iterated (often many times) until all *important* constraints are adequately represented, some irrelevant side conditions are removed, and the solution appears satisfactory. In complex cases, the solver software may have to be modified as well. How existing software can be adapted to run faster on particular models is another topic that needs to be taught. That is why there is a basic need to have both large-scale, real-world instances and flexible software that can handle such instances available in the classroom and on the students' laptops.

Industry Projects

At the DFG Research Center Matheon, the Zuse Institute and TU Berlin, we have been involved for many years in quite a number of industry projects in areas such as public transport, logistics, manufacturing, telecommunication, chip design and energy optimization. Many master's and Ph.D. students participated in these projects and have developed high performance codes in the areas of linear programming (LP) and mixed integer programming (MIP). Now we have made an attempt to integrate such projects into classroom education and provide a complete tool consisting of the algebraic modeling language Zimpl (http://zimpl.zib.de), the LP solver SoPlex (http://soplex.zib.de) and the MIP framework SCIP (http://scip.zib.de).

This combination of software allows users to easily build and solve linear mixed-integer programming models. Models written in the Zimpl language can be directly read by the MIP-solver SCIP. The software is highly portable, free for academic use, and the complete source code, as well as ready-to-run binaries for several platforms, are available for download.

As recent benchmarks show (http://plato.asu.edu/bench.html), the performance is at least on par with the best other freely available tools and, depending on the instance, sometimes even comparable to the top commercial codes. Two other initiatives that provide free tools with source code should be mentioned: The GNU Linear Programming Toolkit GLPK (www.gnu.org/software/glpk) and the Computational Infrastructure for Operations Research COIN-OR (www.coin-or.org).

We tested the combination of SCIP, Zimpl and SoPlex as the main software environment for the "Optimization at Work"

block course held in Berlin in October 2005. The course consisted of more than 80 hours of lectures and exercises, with more than 100 participants from 10 countries.

The participants were mostly graduate students studying mathematics. There were also several undergraduate students, and Master's students in engineering and computer science.

Students had to bring their own laptops. This was very successful for two reasons: First, everybody could work within a familiar environment since the software presented is available for Linux, Windows and Mac-OS. Second, after the course the students took home not only their newly acquired knowledge, but also their working environment, keeping the ability to actually model and solve problems.

New Application Area

During the block course, a new application area was usually presented in the morning lecture of each day. In the afternoon, the students received data and a description of the problem setting. Then they had to work out and solve the mathematical models on their own. Note that many of the students had neither seen Zimpl nor worked with a MIP-solver before.

The students were given problems covering a wide range of applications, including computing the tour of a welding robot, modeling Sudoku puzzles, solving Steiner tree problems, optimizing pizza production, computing routing weights for IP networks, modeling telecom network design problems, bus routing, driver scheduling, line planning and repairman tour scheduling problems. The students experienced that choosing the right model is often more important (and more effective) than having the best solver implementation. Especially with real-world problems, having the ability to experiment swiftly with different formulations is essential. Conveying this idea is a major issue, as mathematically equivalent formulations might behave very differently in practice.

Having students address hard problems from practice by building mathematical models and trying to solve them is a great experience. The students develop practical skills and advance their knowledge both in theory and practice. In the fall of 2006 we repeated this two-week block course in Germany and again with 50 students in Beijing, China. We received enthusiastic student evaluations. We will now start to work on making the course material generally available. **IORMS**

Tobias Achterberg *(tachterberg@ilog.de) is a Ph.D. student at TU Berlin and software developer at ILOG.* ***Martin Grötschel*** *(groetschel@zib.de) is a mathematics professor at TU Berlin, the vice president of Konrad-Zuse-Zentrum and chair of the DFG Research Center Matheon in Berlin.* ***Thorsten Koch*** *(koch@zib.de) is a researcher at the Konrad-Zuse-Zentrum Berlin.*