OPTIMA 101

Mathematical Optimization Society Newsletter

MOS Chair's Column

September 5, 2016. It is with great excitement that I write my first OPTIMA column! After being a Vice Chair for a year I switched roles with Bill Cook last July. This is also a very good moment to thank Bill for all his work as a chair. I am really happy that MOS has the system of the previous chair staying on as a Vice Chair for another two years so that I can profit from all of Bill's experience and his good advice. I am also looking forward to working together with the Council and the officers.

Last summer there were several nice events linked to the society in one way or the other. There were IPCO (Liege), ICCOPT (Tokyo), and ICSP (Búzios) conferences organized on three different continents. Also, my "alma mater" CORE at Université Catholique de Louvain celebrated its 50th anniversary. During the meeting one of our previous chairs, George Nemhauser, and one of our distinguished members, Michel Goemans, were awarded a Dr. hc. from the university.

One of the many things that make mathematical optimization so interesting is its interaction with other fields of mathematics. During the summer, the use of polynomial optimization in proving a new vastly improved upper bound on the size of a cap set in a certain vector space, attracted a lot of attention. The result by Ellenberg and Gijswijt was discussed on several math blogs, among others the one by Terence Tao!

I hope for a lot of interaction with, and suggestions from, the membership by mail or in person!

Karen Aardal
Delft Institute of Applied Mathematics
k.i.aardal@tudelft.nl

Note from the Editors

Dear MOS members,

Whoever has attended a conference on discrete optimization during the last 25 years most likely has realized the role CPLEX and Gurobi have played for research concerning practical aspects of the discipline. But it's not only about research. Despite originating from academic desires, the little programming project Bob Bixby started in the eighties very soon turned into a great business endeavour. In the interview you find in this issue of our newsletter he shares with us a lot of interesting and entertaining insights into the success story that evolved subsequently. In two discussion columns Martin Grötschel and Bill Cook point out how important Bob Bixby's work has been for the development and the spreading of linear and integer programming.

Many thanks to Bill Cook for supporting Optima as the MOS chair during the last three years, and a warm welcome to our new chair Karen Aardal!

Sam Burer, Co-Editor
Volker Kaibel, Editor
Jeff Linderoth, Co-Editor

You have to figure out who your customer is going to be – An interview with Bob Bixby

The Birth of an LP Solver

Your initial research was in matroid theory. Can you talk a little bit about how you started getting interested in building a linear programming solver?

I have told this story many times, but it's still fun to tell. I was teaching at Northwestern in the early 80's, and these IBM PCs had appeared. I was playing around with that stuff. I wanted to use linear programming in my classes. There were a lot of things that somehow got in the way. There were a few LP codes around then, like Roy Marsten's XMP and Linus Schrage's LINDO. Anyway it was LINDO I tried to get. I didn't want to pay for the academic license, so I wrote my own little code for this class.

In C already?

It was not. No. In Fortran. This was actually when I was starting to get in to programming a little bit. Back then it was Bill Cunningham and I. I forget exactly when it was published, but this was around the time when Bill Cunningham and I wrote a paper showing that you can basically use some matroid-motivated algorithms to test whether you could do elementary row and column operations on a matrix to turn it into a network problem. Tom Baker, who was working at Exxon at the time, heard me talk about this at a meeting at SUNY Binghamton. We got to know each other and I visited and consulted at Exxon a little bit on network optimization.

Then Tom started Chesapeake Decisions Science, one of the early supply chain companies. It might have been the first, I'm not an ex-

Contents of Issue 101 / September 2016

pert on the history. He took me up to his attic, his hatchery if you will, where he was getting ready to start this business. He showed me what was the early versions of this MIMI language, which was sort of the kernel software that Chesapeake used. Then he started his company, and he got in touch with me. I don't remember the exact year but this was all in the early 80s. He needed a network optimization code to put into MIMI. He had himself gotten a book and had implemented a network simplex. He called me up and said basically, "If you can implement a network algorithm, I'll compare it to mine. If yours is faster, I'll take it and pay you something for it." I implemented a version of a network simplex algorithm.

Was yours faster?

Tom did a test, and his implementation must have been really bad because mine was nothing special but it was like sixty times faster so they embedded it. It's embedded in Aspen to this day. Well not the exact original version. Tom was a great guy. Everything was agreed to on the phone or with a handshake. He paid me whenever they sold MIMI and this network code was in there. I don't remember what I made a year, maybe $20,000.

He never asked you to improve or enhance the code?

He never did. At some point for some reason, I realized that I could make it better and actually paid David Applegate to just fix up the data structures and the trees to do smarter things.

Anyway, so then sometime after this because he was happy with this experience, Tom calls me up and he says "I need an LP code, I'm using (Roy Marsten's code) XMP but it's too big." So remember we are running on these machines, they are 32-bits—wasn't much memory. And his old MIMI system including XMP was just too big. And he asked if I had an LP code. I said, "Yes I have an LP code." It really was this really simple-minded thing. Anyway, it got converted to C, and then what ensued for me was, in retrospect, an incredibly interesting period. It lasted about two years.

Is this just about when you were going to Rice [University]? In 84/85?

Yes, I think I was probably already at Rice. But I was certainly influenced by my time at Northwestern, because at that time I knew very little about computational linear programming. I'd only taught linear programming.

But not sophisticated numerical topics like basis factorization techniques?

Basis updates, you name it, I knew nothing about it. But Bob Fourer did. I had actually hired him at Northwestern. Bob is a very smart guy, knowledgeable. He had worked for a government entity of some kind. He did staircase linear programming and had a lot of experience with MPSX. He knew what was out there, and we discussed things. He sensitized me to some of the issues. But I never actually went and read anything. So then I gave this LP code to Tom Baker. What ensued was almost exactly a two-year period. It was constant, he would just send me a problem. He knew how to push my buttons; he would send me a problem.

Some instance that was slow or breaking?

Breaking. Mostly breaking. I'd think, "Ah it's a piece of junk." I would be at home. This is my hobby. I would mess around and figure it out. It was just numerical stability problems one after another. I would fix one and wait for the next one.

Just to get a picture of what else was going on at that time, were you still doing research with matroids?

Purely matroids. Linear programming was not research for me, just playing around. But even to this day I love running benchmarks and solving problems. That aspect of my personality sort of emerged naturally. I would start working and I would just get captured. I did not know any numerical analysis. It was all new to me actually, the whole thing, a big epsilon delta proof, absolutely was. But it was all balancing epsilons and deltas. It was like proving something was going to converge to get these tolerances to balance out right.

So you were focused solely on numerical issues?

I was really almost completely focused on numerical issues. Sometimes I was getting occasional ideas. For example, how to do partial pricing in a different way.

Would you have stopped working on this if there were not these instabilities? Say Tom just keeps sending you checks and nothing is breaking?

Sure. But we know that's impossible. Even if it had been a really good code, it was going to break. After about two years of the code constantly breaking, I was thinking to myself, "This is really junk." Finally, one day one of Tom's people calls me and says "By the way, Amoco wants to buy your code." This can't be true. They've got MPSX. Why would they want to get my code? So then I actually for the very first time thought that maybe my code wasn't so bad. I was going to make a comparison. I knew little of the landscape of LP at that time, and I thought that XMP was the standard. I had no idea that in fact MPSX was way faster.

Did you do a benchmark against XMP?

So after these two years, I did a test. This was quite amusing. I was going to compare my code to XMP. All that was available then was this NETLIB list of twenty incredibly tiny problems by today's standards but those were the problems that were available. I really did not want to fail. I did not want to do this test and discover what I had really was junk. So I carefully ran through the list of models, and I picked up one of these models known as `standata`. It was a very sparse problem, and I knew my linear algebra wasn't so hot, but that this would put me in the best position. So I ran it against XMP on this one model and I think I was on the order of seven times slower. I was crushed, absolutely crushed. And then I sort of woke up two days later thinking, I got to fix this. I worked like crazy for two weeks. And I eventually got it so it was only a factor of two. I don't remember what I did. Eventually, I finally woke up from being so irritated and crushed and looked. I think `standata` had like a thousand rows. XMP was solving it in 70 iterations! Hello? XMP was getting lucky on the starting basis. So then I just ran all the rest of the problems. `standata` was the only one it was slower on—I am not exaggerating. In fact, within a week or so on almost all the models my code was four times faster.

Are there things in that version of CPLEX that you thought were especially novel?

I had my own novel version of partial pricing, but initially I was very proud of the numerical stability. You discover really quickly that most negative reduced cost, well that it is a bad idea. It is unstable for one thing. It's too predictable. You want things that aren't so predictable. The thing I didn't know about these problems Tom Baker was sending me was that they were snapshots from a sequential linear programming procedure. Notoriously unstable, most unstable problems I've ever seen. I was solving problems that had condition numbers like 10^{20}. In general, if you have a condition number like that, the answer doesn't mean anything, and you should just give up anyway. But if you do an SLP you can't give up. You want to get to the end. Because at the end it does make sense. So it was really a perfect storm. It was perfect, the code was incredibly stable after these two years.

Starting CPLEX

In what year are we now? Is this when you thought about starting a company?

I think this is around 1986. After the licensing to Amoco (via Chesapeake) went through, I started to try to sell it. It took me on the order of six months to discover I had no idea how to start this business. So then I actually recruited Janet Lowe. I had been teaching in the business school, and she was the best. I picked the two best students that I had in the 3-4 years I had been teaching this modeling class in the business school. She was actually my first choice. I asked her if she was interested in this company I had in mind, and she was. So that's when we started. In order to get started, I thought that I had to do something that I believe very few academics would have done. I came to the conclusion that there was no way I could really get good people to be involved in this enterprise and be committed to it if I was just paying them a salary. I gave them 49 percent of the business. They had done nothing.

You gave 49% of the business to Janet Lowe?

Yeah, with her husband Todd Lowe. I had done all this work for 4 or 5 years, but I still decided to give them a bunch of equity in this enterprise. That turned out to be a smart thing to do.

Janet is a good finance person and a very good marketing person. Todd is a fantastic salesman and he is technically extremely strong. That's how the business got started; we were incorporated in 1987. The first release was in 1988, and at that time I was responsible for all of the development. Around this time, I started to get more serious about the company. After 1989 or 1990, it was pretty clear that the venture was working.

Did you only have an LP Solver at that time?

No. I don't know exactly when the MIP came in, but Todd gets credit for that part. He was the one doing the sales. He just immediately figured out, if you want to succeed in this business you needed a MIP.

CPLEX Development

What was your role in designing the IP Solver?

One thing I helped design was the callable library. One explicit paper that influenced me was a paper by Grötschel in the early 90's. He showed that if you paid a lot of attention and developed cutting planes, you can solve difficult integer programming problems. (I think it was the linear ordering problem.) There was also a TSP paper where he was complaining about how hard it was to use MPSX. That's when I got the idea of building a framework. All these people were doing research in integer programming, and they needed a black box solver that fits this environment. This idea that the LP was in the background. All you had to do is change your bound, and CPLEX would manage the rest. CPLEX would figure up the next basis and solve the next problem. The callable library, which was exactly the right thing for what people wanted to do in integer programming actually turned out to be exactly the right thing for business. That took over.

When did it come out, the callable library?

It had to be right around 90/91.

So you were competing against IBM's OSL at that point? They had "user exits" for amending algorithm behavior as well.

We were competing against OSL at that point, and OSL was certainly better. We probably caught OSL in terms of LP around 1992 or something. In 1994, the dual simplex method really matured. We really figured it out, had dual steepest edge, and we were doing a lot of smart things.

Wasn't dual steepest edge in OSL at that point as well?

Back then, dual didn't really exist. You think it existed, but what existed was only a weak sister. Dual was only implemented as something to use within MIP. Nobody thought of it as an actual algorithm to solve problems—you know, as a first class citizen. So that was a new idea. I remember being at another Oberwolfach meeting, it must have been around 93/94. Somebody was up there talking about stuff that they were doing with airline problems. I don't remember the exact topic, but essentially I raised my hand said, "Don't you guys know about dual simplex with steepest edge. You should be using it." This completely changed the practice. Of course this is exactly the right thing to do for those in the know.

Did the early versions of CPLEX have cutting planes?

There where knapsack cuts. But in the early versions, you could see that solvers such as MINTO were solving problem we couldn't solve.

Were you already doing presolve and all the techniques like from the famous Crowder, Johnson, Padberg paper?

I don't think we had a particularly good presolve. The MIP part of the code was a kind of creeping along until Irv Lustig came on at some point. He didn't know about MIP explicitly, but he did know presolve. The MIP code took off really when Ed Rothberg and Zonghao Gu came on board. That was really the CPLEX 6.5 story. Ed's first job after Stanford was Intel, and he implemented a parallel MIP code for them. Then Ed went to work for SGI. We sort of built a partnership with SGI. The object of course from SGI's point of view was to sell SGI machines. We could link into CPLEX Ed's parallel barrier, which was the first parallel barrier, with all his knowledge of Cholesky. We had a barn-burner barrier code. And I had actually worked on crossover, making a couple of little observations that made it way faster.

Back then, we had been trying to break into the airline business, because they were all using IBM hardware and OSL. We could demonstrate to them that we were faster in some cases, but the problems were still solving in an acceptable amount of time for them with OSL, so they weren't interested in switching.

But what turned the corner for us was that I think it was American who was considering a merger with another airline. As part of the due diligence for the merger, they wanted to look for opportunities in terms of more efficiently running fleets. They needed to do this reasonably fast. Because they needed to go back and forth talking with all the people considering the deal. For these problems, our parallel barrier code running on SGI was around 40 times faster. It made something possible that wasn't possible. So they started to use our stuff. They really had no choice.

What else was happening at CPLEX around this time?

It was an exciting period with this competition going on. One good story is that back in the early 90's CPLEX didn't really have anything to fight degeneracy.

Nothing!?

It was really nothing. If you weren't around then, the philosophy, the belief in the subject, was sure some problems were massively degenerate, but it was not getting in the way of solving things. It was a combination of two things that led to changes in the CPLEX code. First, we got this instance LAU2 (United Airlines spelled backwards) from John Gregory at Cray. It was an early fleet model they wanted to solve. I ran CPLEX on it, which would have been version 1.something or maybe 2. It ran for 7 hours on a Cray YMP and was still stuck in phase one. It was making no progress. The second event was that a graduate student of mine, Sanjay Saigal, was actually working on max-cut-problems. These problems were horribly degenerate. I had to invent something in the code to handle this degeneracy or Sanjay could not do anything. So that's when perturbation got introduced.

I don't know if you know the story. I think my most important contribution was this idea of what is called shifting, which came out of an idea from Paula Harris for the ratio test, and it is closely related to perturbation.

And all the codes use it now. It is basically the following idea: Observation number 1: you have to have a feasibility tolerance. You can't put a hard zero on it because problems are massively degenerate. This means when you solve for the basic variables there are going to be a bunch of variables that want to be zero, but they aren't. Some will be slightly positive, some of them slightly negative. If you insist on zero, it concludes everything is infeasible. So you need a tolerance. It is similar to an idea of Paula Harris for exploiting the tolerances to improve the stability of the ratio test. You can expand the set of possible leaving variables. It was Bob Fourer who made me aware of this.

One of the unfortunate parts of this was that you get variables that actually were negative. And all of a sudden then you would have a basic variable that was actually negative. And instead of it going from greater than or equal to 0 to 0 when it went out of the basis, it would actually go from a negative value to zero. In other words, you thought you were moving in one direction, but in effect the pivot ended up going in the other direction. So you get non-monotonicity, and I struggled with this for like two weeks. How am I going to fix this? I tried all kinds of instrumentation, until I got a very simple idea which was—change the bounds. So if it was negative and I didn't want to make a negative step, I just changed the bound on the variable to make it feasible. This is bound shifting.

So when you get done, you solved a different problem because you have changed the bounds on some variables by some small amount. But you are certainly dual feasible with the objective function. In particular, since the simplex method is a combinatorial gadget, unlike interior point algorithms, you just set the bounds back to what they're supposed to be and solve for the basic variables. 95% of the time I was feasible too. You are done.

So geometrically you avoid degenerate vertices by walking along infeasible faces.

You're jumping ahead a little bit. That's correct actually. But the reasoning at that time is just to be more numerically stable, which then lead to having to do something to deal with these negative variables, which then lead to this bound shifting idea, which introduced a natural perturbation. Automatically made the simplex algorithm less sensitive to degeneracy at the same time.

Is there a paper about this?

There's no paper on this particular thing.

On what you think was one of your greatest contributions? Kind of an unfortunate thing. Why is there no paper?

I talked about all the stuff. I just never got around to it.

So this is all basically about LP's still? What were you doing for MIP?

As far as the MIP stuff goes, basically I bask in the reflected glory for what was done by other people. I didn't do much on the MIP stuff. I played a small role in it in the end. A lot of the MIP stuff was influenced by the TSP work which I was very much involved in. For example, strong branching came out of that, which got developed. Somehow in a discussion that Bill Cook and I were having, it came up and I put it in CPLEX. Also, I spent a lot of time tweaking and fiddling with the CPLEX library to make it work extremely well with the TSP code. I did this sort of knowing in the background this work was going to help integrating anything that had to do with integer programming. So I was certainly contributing to MIP in that sense. However, the real contributions got made by Ed Rothberg and Gu. And now by Tobias Achterberg.

When did the big improvements in MIP come?

CPLEX 6.5 was a 10 times improvement. Gu was the one who really came in and implemented all of the cut stuff. When it was discovered that these Gomory cuts, mixed integer cuts work so well. Those are so easy to implement. That's the silly thing about it, they're the easiest ones. There's numerical issues you have to worry about it, and you need to take it seriously. But they're the easiest ones to implement. Gu was the one who's doing that stuff. Just a ton of stuff. Most of presolve. Ed put on the node presolve and started putting in the probing stuff. Also started putting in the bounds strengthening stuff and so forth. Then just increasingly of course this idea and that idea. For example, started putting in heuristics for the first time.

Starting Fresh

So what prompted the ILOG acquisition? It was around 97?

It's exactly 97 that it happened. The idea actually did not come from me at all. Todd and Janet hatched the idea. A part of the motivation was this was the time for IPOs. Selling was really a way to do an IPO without having all the costs of getting investment bankers and doing a road show. I was doing development and was still an academic at this point in time. The other motivation was that Todd, who was managing the company, was not interested in managing a larger enterprise that was potentially international. You know the web was not then what it is now. So if you really wanted to sell to people around the world, you would have to travel around the world. Selling to somebody else for that opportunity to get the upside without having to manage this larger thing. That was a part of the motivation and then we were looking around for the opportunities that were available.

This was right around the time when ILOG had done an IPO. They were a French company, but they came to the U.S. and did an IPO on the NASDAQ. They came to the U.S. and went to companies saying "We're the optimization company." Most companies told them, "No, you're not. CPLEX is the optimization company!"

So ILOG approached you?

Yes, they approached us. That's right. They bought CPLEX to be a subroutine of the constraint programming set. It turns out things got flipped at some point.

Why did you decide to leave CPLEX and start Gurobi?

I can tell you why I wanted to start Gurobi. Now I can't speak 100% for Ed and Gu, so I shouldn't. But I can tell you in my case. The straw that broke the camel's back for me was that mixed integer programming CPLEX was never a strategic product for ILOG. When ILOG was thinking strategically about optimization, they were never able to in the context of MIP. They always start in terms of constraint programming. All the ideas they got were constraint programming things.

So you left before the IBM acquisition of ILOG?

Oh yes. We all had left, but not so much before.

How long after you left ILOG did you decide you're bored and wanted to do something?

It was very soon after. Gu and Ed literally within 3 or 4 weeks started working together. They were planning to do something. I left, got in touch with them, and asked if I could help.

When you got back in, were you active in the initial Gurobi development? Did you get your hands back on the simplex method again?

No. Absolutely zero development. I did quite a bit of testing. I pestered the development team all the time to say, "Why isn't this performing and that performing?" I saw this bad behavior and that

One must imagine Bob Bixby working on improvements of his LP-code with a similar dedication as we can observe at this picture which shows him carrying out some highly precise wood works in his garage workshop. The piece he is constructing is a copy of Hamilton's Icosian game (Bill Cook obtained the precise measurements of the original game from a museum in England), in which one player chooses a path with four edges in the graph of the Dodecahedron which the other player then has to complete to a Hamiltonian cycle. *(Photo: Bill Cook)*

bad behavior. "Why is it doing this and why is it doing that?" What I really was interested in doing was running the business.

What do you think. How important was it to give it for free to academic people immediately?

A company like Gurobi, there would be fundamentally two reasons. But we didn't think about it very hard, we just did it. The two reasons would be to first of all establish the name. CPLEX is such a strong brand name. You've got to get it out there; you've got to get it out there in the academic community. And even though we've gotten it out there and so forth, this is still 4 or 5 years later, and we're just starting to get to the point that it isn't just assumed that mixed integer programming equals CPLEX.

The other reason is that anecdotally 60 % of the licenses were going to non-OR people. So that's a reason in and of itself—more people, biologists or whatever, using this technology and it spreads the gospel. We all benefit from that. Those are the two reasons.

Speaking of commercial, what would you say are the branches of business where MIP, Gurobi, CPLEX are used the most?

I actually did a little survey of the customers, companies, we sold to in a one-year period. It was a couple of years ago. Keeping in mind, we don't always know what people are using things for. A lot of times they come to us and it's not even necessarily that they don't want to tell us. They come up, they got an application, they need a solver, they want to do a benchmark. We enable that. They do it.

Anyway, I made a list, and I came to the conclusion that there were about 40 different application areas covered. Number one was supply chain. Number two was electrical power, based upon my counting. Of course there is energy in there and a number of things over energy, but not electrical power. Finance was up there. I think also work force management. I was the person deciding the class and which bucket it goes in, things overlapped. Probably transportation would have been higher if I hadn't separated airlines from railroads for example. Those were separate categories.

When you were with ILOG it was kind of well known that you had a huge library of test instances and that this was a big advantage. When you went off to Gurobi, how long did it take you to develop a reasonable library?

Well, actually, times have changed. We had a library of, I think within two days, we had 600 models collected from various sources on the web. Already fairly reasonable. NEOS contributed enormously—a lot of good models. I think that a lot of good ones are sitting there. I think it needs to be combed through again. We have like 10 000. I think the real test set that they use is more like 3500 models.

So if you attend computational IP talks at conferences do you think they should do the testing in a different way?

I don't see that there is much computational IP that gets talked about.

In some sense, companies like CPLEX, XPRESS, and Gurobi put academics doing computational IP kind of out of business. Many people are doing MINLP now.

I think there is still plenty of room for MIP actually. The truth is that there has to be relatively little that's come out of the research community in the last 10 years or so. Certainly RINS was a big thing. That's had a huge influence. Symmetry testing was a big thing, but not a huge thing. That got put in Gurobi, based on orbital branchings in some form.

It seems harder and harder to generate and demonstrate good/impactful ideas without access to the internals of commercial IP solvers.

It's perfectly clear, it's very hard to test whether something is a good idea or not. I mean the most you can hope for these days is that it somehow looks like reasonable idea when you do a little bit of testing. Then you know people working for the commercial solvers will look at it and try to do something.

Managing cutting planes can be very difficult, and we would actually be happy if there were some nice thing we could do that would allow people to contribute and everybody would benefit from that. We have no interest in making this hard to do. You know there's so much to do—how do you aggregate the cuts? Which ones do you keep and throw away? Mixed integer rounding, Gomory, and flow cover cuts. They all interact with each other.

What is your current role at Gurobi?

I'm Chairman of the Board, and I'm CSO (Chief Strategy Officer). If you work for a company you need a title. Chairman is not a title. I'm half time now, whatever that means exactly. I manage the sales team and run the code all the time and complain like I always have. **laughter**

Do you still do things at Rice?

I have an emeritus title, but I don't go to the department anymore. I did teach for a couple of years in a business school. They were perfectly happy to have me teach for them, but I discovered I didn't really enjoy it so much. I mean the course I taught was fairly popular and they were happy with it, but I wasn't fundamentally enjoying it. I discovered during that period of time if I was teaching, I really liked to teach math. Run the business and be a businessman, but if I'm teaching, I enjoy teaching math. Alexander Martin has arranged for me to teach a class in Erlangen every year for the past 4-5 years.

Matroids?

No Matroids. It's computational linear mixed integer programming. But I prove theorems. I prove things and I enjoy that. I go there for about a month, it's concentrated. So I keep that, I didn't want to give that up. I still enjoy going to meetings and staying connected with what people are doing, but I don't do any research anymore. Well except, we are writing a paper about presolve in Gurobi, actually.

Yeah and you've got to write a paper about bound shifting!

One final question: Do you have any advice for maybe young people at academia who want to turn their research outcome into a business. Do you have to give up on matroids? *laughter*

There really aren't that many success-stories in the OR community of people starting businesses, not many. Advice #1 is if you are thinking about selling something, you have to figure out who your customer is going to be. Put yourself in the position of the person who is going to buy it. It's really easy to think, I've got this wonderful algorithm, and it's going to save this company so much money. Okay, so who are you going to talk to at this company? Most likely you are *not* going to be talking to some executive. They won't understand it. More likely, you're going to be talking to somebody at the developer level. So this person is likely to be pretty low in the hierarchy and they are going to have to explain why they should spend thousands of dollars on your product.

It also helps to have good marketing people. It helps the business enormously, but if you think the marketing people are going to give you super-duper ideas for the next thing to do because they go out and do a market study, you are kidding yourself, just forget it.

Another thing I see people do wrong all time is underestimate the value of having good sales people. Don't think that because you think you have a great algorithm or whatever, that the world is going to come running to you. You have to turn it into a business and you have to get good people that are invested in that business.

Thanks so much for your time, Bob!

The questions have been posed by Volker Kaibel (kaibel@ovgu.de), Jon Lee (jonxlee@umich.edu), and Jeff Linderoth (linderoth@wisc.edu) in October 2015 during the workshop *Mixed-integer Nonlinear Optimization: A Hatchery for Modern Mathematics* in Oberwolfach, Germany.
We are very grateful to Nicholas Woyak (The University of Iowa) and Susanne Heß (Otto-Von-Guericke Universität Magdeburg) for transcribing the audio recording of the interview.

Martin Grötschel

Comments on Bob Bixby's interview: Mathematics and the Real World

Mathematics is the only scientific discipline that produces „eternal truth". Mathematical results and theories may be beautiful, deep, or complex, and we may admire them as great achievements of the human mind because of that – independent of their relevance for the real world. Some mathematical achievements are significant for understanding and solving practical problems in industry, society, or other academic areas. In fact, there are many examples of the latter kind. What is often overlooked (also by mathematicians) is that mathematics is rarely applicable directly. One step in the application process is "good modelling" which needs interaction between mathematicians and practitioners. In most cases, a mathematical theorem or algorithm is practically only useful if it is made available via executable software.

Many of my mathematical colleagues, in particular those who have never tried to program a mathematical idea, consider implementation a relatively trivial matter and of minor importance – despite the fact that the high visibility of mathematics in the sciences and the respect for mathematical achievements in industry is today largely based on what the modern mathematical codes achieve in a large number of application areas. The statement "Done by computer" usually means that the implementation of a mathematical algorithm produced the result.

Bob Bixby is one of the (few) champions who have made optimization a "household tool", employed by a large variety of users who usually have no clue which amount of work is behind a fast and reliable code for solving linear or mixed-integer programs. In his interview, Bob, very humbly, plays down the role he performed in the last thirty years in making LP and MIP codes the most important workhorses in optimization – easily utilizable and extremely fail-safe.

I have taught optimization for almost 40 years by now, in particular classes on linear and integer programming (even Bob's daughter Ann attended one of my classes). I have bothered generations of students with my requirement that they would not pass the LP class unless having delivered a working code for linear programming. The Simplex algorithm (or any other method for solving linear programs) can be written down on a blackboard in a few lines, but making it work in practice is a real achievement. Students usually do not believe that, so they have to learn it the hard way.

I do remember lots of complaints of students who correctly implemented the formulas that I wrote on the blackboard, but whose code stopped/crashed, and that they had no clue why. I was most astonished that this particularly often happened with students majoring in computer science who had never heard of numerical instability. In fact, still in my last LP class three semesters ago, no code written by students was able to solve all the NETLIB problems of the 1980s (actually most codes failed on half of the problems) which are "peanuts" for every current professional LP code. Moreover, the running times were orders of magnitude slower than professional algorithms, such as CPLEX or Gurobi – and that is what I wanted to show my students. They did finally learn to be somewhat humble and to understand that the implementation of mathematical algorithms is not small potatoes.

One problem with the numerical behavior of linear programming or other optimization codes is that there is not so much theory available that is really applicable. Implementation is more like a handicraft with lots of little tricks and ideas based on experiments and on fiddling with parameters. That is what Bob indicates in his interview. You need to look very carefully at many details, make experiments, and infer reasonable conclusions from these. When I visited the President of the Berliner Handwerkskammer (Berlin Chamber of Crafts) some years ago and outlined the potential use of mathematics for his "business", he gave me his definition of "Handwerk (craft)": "A craft is something that you cannot learn from books, you learn it by experiments and by imitating experienced people."

Bob very humbly describes in his interview how he developed his craft. He implemented linear and integer programming codes by looking at failures, learning on the run, adapting, making little experiments and improvements, and by accepting advice from others. Implementing great software, given this example, is in fact a craft according to the definition of the Handwerkskammerpräsident. The naive users of CPLEX, Gurobi, and similar optimization codes may believe that the enormously increased power of the optimization software that is available nowadays derives from progress in theory. That is true too, but they should be made aware that craft plays an essential role when science impacts the real world – even in mathematics.

I know that some colleagues would be seriously offended if they were called craftsmen. In Bob's case I believe the opposite is true when it comes to his LP/MIP work. Those who have visited his Houston home (like me) know that he is a fanatic woodworker; here he may even pay more attention to detail when it comes to producing a chair, a bed, or a desk. As the "Minimalist Woodworker" Web page says: "You need to spend time practicing the craft whether you are making napkin rings or a piece of furniture." and continues "Hobbyist woodworkers are a lucky group. [...] We can try new things, use different woods and play with the unconventional just to have a bit of fun." I am sure that Bob has this fun when woodworking and implementing and testing mathematical codes.

Bob Bixby is one of the true leaders of our field, and our community owes him a lot. Without his efforts we would not be able today to solve these incredibly large real-world models coming from transportation, supply-chain management, biomathematics, energy, etc. which now make optimization one of the most successful branches of mathematics.

Martin Grötschel, Berlin-Brandenburgische Akademie der Wissenschaften, Jägerstraße 22/23, 10117 Berlin. groetschel@bbaw.de

Bill Cook

Comments on Bob Bixby's interview: Beauty in the Details

When discussing TEX in his book *Birth of a Theorem*, Cedric Villani writes "Knuth has probably done more than any other living person to change the daily working lives of mathematicians." I'd say Bob Bixby comes in as a close second for optimizers, certainly for those of us working in linear and integer programming.

Not everyone uses Gurobi or CPLEX on a daily basis, including, I assume, Bob himself. But every good optimization library used by researchers has been influenced in design and in standards of performance and accuracy by Bob's software. What a joy it is to lay down an integer model, say with the help of AMPL, GAMS, Python, or, in my case, good old C, and fully expect optimal solutions flying back to our computer screens.

What comes across cleanly in the interview is that the revolution in optimization software that has taken place in the past two decades was not founded on an accident or stroke of fortune. But rather on methodical attention to detail. In New York City they have the saying "If you see something, say something." In computational optimization, Bob's motto is that if you see something, then fix it! When numerical results are off, don't ignore the bad values or try to work around them, but identify the problem and see what you can do about it.

I have to point out that in this arena, Bob is aided by the fact that he only sleeps a couple of hours a night. Much easier to tackle a nasty problem when the rest of the world is snug in their beds. And also aided by the fact that details are his forte. I remember spending an afternoon in his Houston garage workshop, watching Bob and Dave Applegate working at leveling some large mechanical device for shaping wood. A millimeter here and a millimeter there, hour after hour. In the end, beauty in the details.

William Cook, Combinatorics and Optimization, University of Waterloo, 200 University Avenue West, Waterloo, Ontario, Canada N2L 3G1
bico@uwaterloo.ca

Call for papers
Mathematical Programming Series B: Special Issue on Topics in Stochastic Programming

Stochastic optimization has seen recent advances with far-reaching impact involving risk measures, connections between robust optimization and stochastic programming, and applications in areas rang-

ing from energy and natural resources to economics and finance to statistical and machine learning. Mathematical Programming, Series B invites submissions of manuscripts to a special issue on stochastic programming with a focus on distributionally robust optimization; scenario generation and reduction; and, stability of stochastic programs, stress testing, and further output analysis. Of particular interest are manuscripts on these ideas using multi-stage stochastic programming, employing risk measures, or involving important applications. The special issue will be dedicated to Jitka Dupacova, one of the founders of stochastic programming whose deep contributions continue to influence the state of the field today.

All submissions will be reviewed according to the standards of Mathematical Programming, Series A. Please submit all manuscripts using the Mathematical Programming style files with a maximum of 25 pages. See ftp.springer.de/pub/tex/latex/svjour3/global.zip for the LaTeX macro package.

The deadline for submission of full papers is January 15, 2017 with first-round reviews expected to be completed by July 15, 2017.

Authors are kindly asked to submit their manuscripts via www.editorialmanager.com/mapr/ and select Jong-Shi Pang as the handling editor for consideration in this special issue.

Please direct questions about the special issue to the guest editors:

Tito Homem-de-Mello, School of Business, Adolfo Ibanez University, Santiago, Chile. tito.hmello@uai.cl

Milos Kopa, Department of Probability and Mathematical Statistics, Faculty of Mathematics and Physics, Charles University in Prague, Prague, Czech Republic. kopa@karlin.mff.cuni.cz

David Morton, Department of Industrial Engineering & Management Sciences, Northwestern University, Evanston, Illinois, USA. david.morton@northestern.edu

Call for papers

Mathematical Programming Series B: Variational Analysis in Modern Statistics

Modern statistics is faced with a series of challenges as it addresses an expanding number of applications in machine learning, artificial intelligence, forecasting, cyber security, smart systems, social networks, and in developing the internet of things. Increasingly complex models, such as those in deep learning, nonparametric estimation, and large-scale statistics, rely on sophisticated mathematical foundations, especially variational analysis. For this special issue, we invite contributions in the interface between statistics and variational analysis including insightful surveys of open problems in statistics as well as technical papers on timely subjects such as fitting criteria in high-dimensions, large-scale optimization algorithms, trade-off between statistical and computational errors, constrained inference, and infinite-dimensional variational analysis in nonparametric statistics.

The special issue will be dedicated to Roger J-B Wets in honor of his 80th birthday in 2017. His fundamental work in variational analysis has often been motivated by statistics and optimization under uncertainty.

All submissions will be reviewed according to the standards of Mathematical Programming, Series A. Please submit all manuscripts using the Mathematical Programming style files with a maximum of 25 pages. See ftp.springer.de/pub/tex/latex/svjour3/global.zip for the LaTeX macro package.

The deadline for submission of full papers is February 1, 2017 with first-round reviews expected to be completed by August 1, 2017.

Authors are kindly asked to submit their manuscripts via www.editorialmanager.com/mapr/ and select Jong-Shi Pang as the handling editor for consideration in this special issue.

Please direct questions about the special issue to guest editor:

Johannes Royset, Department of Operations Research, Naval Postgraduate School, Monterey, California, USA. joroyset@nps.e.du

Application for Membership

I wish to enroll as a member of the Society. My subscription is for my personal use and not for the benefit of any library or institution.

☐ I will pay my membership dues on receipt of your invoice.
☐ I wish to pay by credit card (Master/Euro or Visa).

Credit card no. _____ *Expiration date* _____

Family name _____

Mailing address _____

_____

_____

Telephone no. _____ *Telefax no.* _____

E-mail _____

Signature _____

Mail to:
Mathematical Optimization Society
3600 Market St, 6th Floor
Philadelphia, PA 19104-2688
USA

Cheques or money orders should be made payable to The Mathematical Optimization Society, Inc. Dues for 2016, including subscription to the journal *Mathematical Programming*, are US $ 90. Retired are $ 45. Student applications: Dues are $ 22.50. Have a faculty member verify your student status and send application with dues to above address.

Faculty verifying status _____

Institution _____

IMPRINT
Editor: Volker Kaibel, Institut für Mathematische Optimierung, Otto-von-Guericke Universität Magdeburg, Universitätsplatz 2, 39108 Magdeburg, Germany. kaibel@ovgu.de ■ **Co-Editors:** Samuel Burer, Department of Management Sciences, The University of Iowa, Iowa City, IA 52242-1994, USA. samuel-burer@uiowa.edu ■ Jeff Linderoth, Department of Industrial and Systems Engineering, University of Wisconsin-Madison, Madison, WI 53706-1572, USA. linderoth@wisc.edu ■ **Founding Editor:** Donald W. Hearn ■ Published by the Mathematical Optimization Society ■ Design and typesetting by Christoph Eyrich, Berlin, Germany. optima@0x45.de ■ Printed by Oktoberdruck AG, Berlin, Germany.