Michael Todd: Optimization, an Interior Point of View >>>

Michael Todd

Interview of Michael Todd by Y.K. Leong (matlyk@nus.edu.sg)

Michael Todd is well-known for his fundamental contributions to continuous optimization, both in the theoretical domain and in the development of widely-used software for semidefinite programming.

His research work has left a deep impact on the analysis and development of algorithms in linear, semidefinite and convex programming; in particular, on interior-point methods, homotopy methods, probabilistic analysis of pivoting methods and extensions of complementary pivoting ideas to oriented matroids.

He did his B.A. at Cambridge University and Ph.D. at Yale University. Except for a two-year stint at the University of Ottawa, his scientific career began and developed into prominence within Cornell University, where he is now the Leon C. Welch Professor in the School of Operations Research and Industrial Engineering.

He has been invited to give talks at major scientific meetings and universities throughout the world. He held special appointments at leading universities and centers of research in economics and operations research, such as the Fields Institute (Toronto), Carnegie-Mellon University, the Cowles Foundation for Research in Economics (Yale), the OR Center (MIT), the University of Washington, BellCore (US), Cambridge University and the Center for Operations Research and Econometrics (CORE, Leuven, Belgium). He has served, and continues to do so, on the editorial boards of leading journals on optimization, operations research and computational mathematics. Among the honors and awards given in recognition of his important research are Guggenheim and Sloan Fellowships, the George B.

Continued on page 17

Continued from page 16

Dantzig Prize of the Mathematical Programming Society and SIAM, the John von Neumann Theory Prize of the Institute for Operations Research and the Management Sciences (INFORMS) and INFORMS Fellow.

Todd has close research links with NUS faculty in the Department of Mathematics and was Chair of the Organizing Committee of the Institute's program on "Semidefinite programming and its applications" held in 21 December 2005–31 January 2006. During his visit for this program, Y.K. Leong interviewed him on behalf of *Imprints* on 9 January 2006. The following is an edited version of the interview in which he gives us a stimulating glimpse of the theoretical insights behind one of the most important applications of the mathematical sciences to operations research, engineering, economics and industry.

Imprints: You did a B.A. in mathematics at Cambridge and went to Yale to do a Ph.D. in administrative sciences. Was the thesis topic a mathematical one?

Michael Todd: Yes. I took a course from Herbert Scarf in mathematical economics at Yale. He described his recent work in computing approximate fixed points, and I got very fascinated by his work and, in general, by complementary pivot algorithms which use purely combinatorial arguments to solve optimization problems. I wanted to understand the combinatorial background to these methods. That was the basis of my thesis. It was indeed a mathematical one. "Administrative sciences" is a strange name. There aren't too many departments of administrative science, and they chose it so that it didn't sound too much industrial, too much business school. Basically, it's about the science and mathematics of decision-making.

I: Why didn't you go to the mathematics department instead?

T: I had been supported in Cambridge by Shell. They had a fellowship for me and they suggested that I go abroad for a couple of years to a business school. With a fellowship between my college in Cambridge and Yale, I went there mainly to see America for a couple of years and then I decided to stay because it was fascinating. Choosing the department was sort of difficult, and it was really an accident.

I: Were you interested in pure or applied mathematics right at the beginning?

T: At Cambridge, my work was basically in pure mathematics, but towards the end of it — and especially when I was at Yale — I decided that the applications were interesting. I got fascinated by the applications, in particular, by algorithmic questions.

I: Is semidefinite programming a generalization of linear and convex programming?

T: Semidefinite programming is a generalization of linear programming. In linear programming the variable is a vector whose components all have to be non-negative. In semidefinite programming, you have a symmetric matrix and all its eigenvalues have to be non-negative, so it has to be positive semidefinite. So it is more general than linear programming but it is a subclass of problems in convex programming.

I: Could you give us some examples of problems that involve semidefinite programming?

T: One of the nicest things about semidefinite programming is the wide range of areas in which it has been applied. I think that the first interest probably came from people in control theory who wanted to study ways of controlling dynamical systems optimally and making sure that they were stable. That led to inequalities that required certain matrices to be positive semidefinite. There are also applications in a completely different area related to combinatorial optimization problems connected with graph partitioning. Another source of semidefinite programming is robust optimization, which has been a hot topic recently. All of these different areas lead to an interest in efficient algorithms for semidefinite programming.

I: Is there an optimally efficient algorithm for solving linear programming problems?

T: That's the holy grail of linear programming research. It's a very intriguing situation. Now we have two different classes of algorithms — simplex algorithms and interior-point methods, and there is wide disparity between them on some classes of problems — sometimes one is much faster than the other. They are very different theoretically. The simplex method in the worst case is exponential but seems to perform very well in practice. Interior-point methods have a polynomial time bound and they perform much better than that bound in practice. For large-scale problems, it is not clear which one is the more efficient. There may be some new methods that will do even better. We're still waiting to hear about that.

I: Are these two methods connected?

T: Not very closely. They are based on very different geometric views of linear programming. The set of feasible

Continued on page 18

Continued from page 17

solutions in linear programming is a polyhedron and the optimal solution always lies at a vertex. So it's natural to consider algorithms that just go from vertex to vertex and that's what the simplex method is based on — an algorithm that traces the skeleton of this polyhedron. Interior-point methods move through the interior and make smooth approximations. So they ignore much of the combinatorial structure and look at the analytic structure.

I: So one is discrete and the other is continuous.

T: Exactly. Interior-point methods never get an exact solution unless you do a special rounding procedure, but they get very, very close, and incredibly fast: you have to solve a very small number of systems of equations which are more complex than the equations in the simplex method.

I: How do you know which method to use?

T: It could be based on the software that you have. Most efficient commercial software allows you the option to use either one. I think people look at their class of problems and decide which one works better for their problems.

I: Which one is more popular?

T: I think for historical reasons the simplex method is more popular, but if you want something jazzy, the interior-point method is certainly a wonderfully efficient method for solving these problems.

I: Are there any probabilistic methods?

T: There are but we should distinguish two viewpoints. First of all, some algorithms make random choices and there are some very interesting theoretical ideas that have been used in low-dimensional problems that have much better computational complexity on certain classes of problems than the more usual ones. But there are also probabilistic analyses of the deterministic algorithms that people typically use on large-scale problems. Simplex and interior-point methods work in practice much, much better than their worst-case bounds. We would really like to understand that. One way to do it is to assume that the problem is random and to understand the average behavior of the algorithm on random problems. Some very interesting results have been obtained along those lines.

I: I noticed that there is a mention of homotopy in one of your papers. Is there something topological about it?

T: I think it is more a question of how the methods are based on different geometric views, and earlier I described a little bit how the simplex method is based on the combinatorial geometry and the interior-point method on the convex geometry. My earlier work was related to algorithms for computing approximate fixed points: homotopy ideas come up, but also the combinatorial topology and geometry of triangulations. Those algorithms were very interesting but not too much can be said about their computational complexity. They tend to be useful for small dimensions, up to maybe 50, on very nasty nonlinear problems, whereas linear programming and semidefinite programming problems are often much, much larger and more highly structured.

I: Is the software for implementing the algorithms freely available?

T: That really depends on whether you are talking about linear programming or semidefinite programming. Linear programming is very widely applicable and has huge commercial implications. So the very best codes cost you some money, but there are some very good codes that you can obtain freely. There are a couple of websites where you can get some good codes for linear programming. But for semidefinite programming, the market is probably more in the scientific and engineering community; so you can't charge them a lot of money. Most of the algorithms are freely available, and several of those are available on the web. A good starting point is the NEOS Solver for Optimization site.

I: Have you written some of those yourself?

T: Yes, actually with one of my National University of Singapore colleagues and another colleague: Kim-Chuan Toh, who's in the Mathematics Department here, and Reha Tütüncü of Carnegie-Mellon University. We have a package for semidefinite programming, and it can also be used for linear programming.

I: Could you give us an idea of the complexity involved in semidefinite programming?

T: I'll give you some sort of an idea. First of all, these interior-point methods have been extended from linear programming to semidefinite programming. They typically take a very small number of iterations, perhaps 10 to 50, but each iteration involves a lot of work. Even if you have a problem with sparse data, in the semidefinite case you have to solve a generally dense large linear system of equations and that can be very costly. So these methods are typically very computationally burdensome, and the number of linear constraints can only get up to a thousand or two. These algorithms can give very accurate solutions. Other classes of algorithms, based more on first-order methods, can solve much larger problems with tens of thousands of constraints. They get much less accurate answers and don't have such

Continued on page 19

Continued from page 18

good complexity bounds, but can be quite fast in practice. I'd say a thousand to ten thousand is the order of the matrices involved and the number of constraints that you can handle with these methods.

I: Can all linear programming problems be solved in principle by quantum computers or a theoretically most powerful computer?

T: I don't know a huge amount about quantum computers. From what I understand, I think it is possible to solve linear programming problems in one step. There's only a finite number of possible options, the vertices of the polyhedron, and the quantum computer is allowed to examine them all simultaneously and pick out the best. Similarly for biological computers based on DNA and so forth. I don't know how practical these methods are. For semidefinite programming, I don't see that you can get an immediate solution, but it will be interesting to find out.

I: What happens if one day we really get quantum computers? Will linear programming problems be trivialized?

T: Yes, but maybe also all NP-complete problems too. It's not clear that these methods can really push all problems that are currently considered interesting to become totally trivial. I don't know whether such computers will really ever become that practical.

I: Do you believe in quantum computers?

T: I think it is a nice theoretical concept to consider, but I'm not expert enough on computers to comment on that.

I: Do you consider yourself to be an applied mathematician or a pure mathematician?

T: I'd say applied mathematician — that's what I say to people I meet on the plane. But just as with pure mathematics this generally gets the same response, "That was my worst subject. I don't understand it at all," which is very unfortunate. Sometimes I try to explain some of the nice things that mathematics can do.

I: Do you think algorithmically or geometrically?

T: I think geometrically a lot of the time. There are so many different ways of looking at optimization problems, from optimality conditions, to the theory of the algorithms and the modeling. I try to keep computational concerns in the back of my mind, but I'm still very interested in the theory as well. The geometric viewpoint on optimization problems really attracts me.

I: But at the end of the day, you still have to do the computations.

T: Yes, you do, and it's nice to be within, say, six degrees of separation, or fewer, from people who are actually practically solving applied problems. Even if you are not producing the software, you are motivated by improving the algorithms so that people can actually solve larger problems faster.

I: Except for two years in Canada, you have been at Cornell right from the beginning of your career. Have you ever thought of moving to other universities?

T: There have been a few times when I thought about it. But overall, Cornell has been a very attractive environment for me. The School of Operations Research and Industrial Engineering has some wonderful colleagues, both in optimization and more generally in operations research. The university as a whole, and mathematics and engineering, have wonderful people, and the quality of the graduate students has been terrific. I really enjoy working with the students in operations research and applied mathematics. It's also a wonderful place to live and very naturally beautiful.

I: Do you talk to people in economics?

T: Economics, once in a while, probably less than people in engineering, computer science, mathematics, but still occasionally, yes. My interest in economics was more during the 70s, a long, long time ago. I have sort of lost touch with the latest things that have been done now.

I: What advice would you give to a graduate student who is interested in applied mathematics?

T: You really need to find a problem where you feel so excited about it that you have a fire in your belly to keep working on it. You should look at all options, keep your options as open as possible, find an advisor to help you see the right approach at the right time and to let you do what inspired you to work in the area, and hope you find the way ahead of you.

I: Have you gone back to Britain?

T: I've gone back socially, for family reasons or whatever, every year or so. I spent one sabbatical back there and I've been back for several conferences. I find in the area I'm working in there are interesting people in many places in the world: in England, but also in Belgium, France, Germany, Japan and Singapore besides the US that I work with as well.

Continued on page 20

Continued from page 19

I: The name of the school you are in — "School of Operations Research and Industrial Engineering" — seems to give people the impression that it has very little to do with mathematics.

T: It's more of a question of how it evolved. We have people who are much more involved with practical work and consulting, but I think many of us regard ourselves as a mathematical sciences department within engineering. We have people working in applied probability, statistics, and optimization, from quite a theoretical viewpoint to a more practical viewpoint. It's nice to have that full spectrum, but many of the faculty were very well-trained mathematically. A lot of us have appointments also in the Center for Applied Mathematics, and some people have appointments in mathematics as well.

I: How is your relation with the engineers?

T: Pretty good. Some fields of engineering are closer than others. We are not too much involved in the experimental side, but for example our relations with electrical engineering and computer science are very good.

I: Do you try to educate the engineers mathematically?

T: I try. I very often have students from other parts of engineering taking my classes. Along with the modeling and computation involved, I try to make them understand that the abstract viewpoint can be valuable. I hope they appreciate the beauty of mathematics. I think that in a strong engineering college, the students are pretty much aware of the advantages of having good mathematical training, particularly the graduate students.