informs

ICS NEWS

INFORMS Computing Society Newsletter

Inside This Issue

The Editor thanks Matthew MacLeod, who has been Assistant Editor of *ICS News* during 2007–8. He hails from the Centre for Operational Research and Analysis (CORA), Defence R&D Canada.

2008 ICS Prize Goes to Robert W. Day and S. Raghavan

Jim Orlin (right) presenting the Prize to Day (left) and Raghavan (center)

The 2008 ICS Prize was awarded to Robert W. Day and S. Raghavan for their paper, "Fair Payments for Efficient Allocations in Public Sector Combinatorial Auctions" [*Management Science* 53:9 (2007), 1389–1406].

This paper presents a new, practical approach for combinatorial auctions, auctions in which bidders specify bids on bundles of items rather than on individual items. Combinatorial auctions have been applied when the value of a bundle of goods to a bidder is not merely the sum of the values of the individual items, such as in airplane slot allocations and spectrum allocations.

continued on page 30 ▷

2008 ICS Student Paper Award Goes to Guanghui Lan

Alper Atamturk (right) presenting the Award to Lan (left).

The 2008 INFORMS Computing Society Student Paper Prize was awarded to Guanghui (George) Lan for his paper "Efficient methods for stochastic composite optimization" (available at http://www2.isye.gatech.edu/~glan/papers/OPT_SA4.pdf). Guanghui Lan is a fourth-year doctoral student at Georgia Tech and is advised by Arkadi Nemirovski, Renato Monteiro and Alexander Shapiro. The paper considers a stochastic convex optimization model in which the objective function is the sum of smooth and non-smooth components. The objective function is stochastic in that its subgradient is available only through an unbiased estimator. A new robust stochastic approximation algorithm is shown to achieve the theoretically-optimal rate of convergence. The committee valued the paper's rate-of-convergence results, which involve both the expected error of the algorithm's iterates and associated large-deviation results.

continued on page 30 ▷

About Us

The ICS mission is to articulate and lead the development of interfaces between operations research and computer science. It began in 1976 as the *Special Interest Group on Computer Science* of the Operations Research Society, then elevated to the *Computer Science Technical Section* [ICS *History Archive* ▹]. We became the INFORMS Computing Society in 1998 and continue to fulfill our mission with fun-loving camaraderie.

We are INFORMS' leading edge for computation and technology.

Officers

Chair:
Robin Lougee-Heimer
IBM T. J. Watson Research Center
robinlh@us.ibm.com

Vice-Chair / Chair Elect:
Robert J. Vanderbei
Princeton University
rvdb@princeton.edu

Secretary / Treasurer:
Kipp Martin
University of Chicago
kipp.martin@chicagogsb.edu

Past Chair / Historian:
John W. Chinneck
Carleton University
chinneck@sce.carleton.ca

Board of Directors

Robert F. Dell
Naval Postgraduate School
dell@nps.edu

Pascal Van Hentenryck
Brown University
pvh@cs.brown.edu

Steve Dirkse
GAMS Development Corp.
sdirkse@gams.com

Matt Saltzman
Clemson University
mjs@clemson.edu

Jonathan Eckstein
Rutgers University
jeckstei@rci.rutgers.edu

Jonathan Owen
GM R&D Center
jonathan.owen@gm.com

Editors

Journal on Computing:
John Chinneck
Carleton University
editor_joc@mail.informs.org

ICS News:
Harvey J. Greenberg
Denver, CO
ICSnewsEditor@mail.informs.org

Matthew MacLeod
Defence R&D Canada — CORA
mmacleod@ieee.org

Mathematical Programming Glossary
Allen Holder
Rose-Hulman Institute of Technology
icsMPGlossary@mail.informs.org

Representatives

COIN-OR:
Robin Lougee-Heimer
robinlh@us.ibm.com

INFORMS Subdivisions Council:
John Chinneck
chinneck@sce.carleton.ca

Practitioner and Practice Activities Committee:
Ioannis (Yianni) Gamvros
igamvros@ilog.com

Communication

Webmasters:
Pascal Van Hentenryck, Brown U.
pvh@cs.brown.edu
Laurent Michel, University of Conn.
ldm@engr.uconn.edu

ICS Email List Moderator:
Matt Saltzman, Clemson University
mjs@clemson.edu

Blog:
Bill Hart, Sandia National Labs
wehart@sandia.gov

Photographer:
Harlan P. Crowder, Retired
hc_subscribe@comcast.net

Projects/Committees

Leading Edge Tutorials:
Rob Dell, Naval Postgraduate School
dell@nps.edu

Education:
Jill Hardin (Chair), Virginia Commonwealth U.
jrhardin@vcu.edu
Chris Beck, University of Toronto
Kevin Furman, ExxonMobil
Art Hanna, St. Mary's University
Allen Holder, Rose-Hulman Institute of Technology
David Rader, Rose-Hulman Institute of Technology
Cesar Rego, University of Mississippi
Advisors: Harvey Greenberg and Ariela Sofer

Membership:
Richard S. Barr (Chair), SMU
barr@seas.smu.edu
Harvey Greenberg, U. Colorado Denver
Karla Hoffman, George Mason University
Irv Lustig, ILOG
Ted Ralphs, Lehigh University

2009 Biennial Meeting:
ics09@mail.informs.org
John Chinneck, Carleton University
Bjarni Kristjansson, Maximal Corp.
Matt Saltzman, Clemson University
Chris Starr, College of Charleston

2008 ICS Prize:
icsPrize@mail.informs.org
Jim Orlin (Chair), MIT
Mike Trick, Carnegie Mellon University
Pascal Van Hentenryck, Brown University

2008 ICS Student Paper Award:
icsStudentAward@mail.informs.org
David Morton (Chair), University of Texas
Alper Atamturk, University of Calif-Berkeley
Nick Sahinidis, Carnegie Mellon University

2008 Harvey J. Greenberg Service Award:
hjgserviceaward@mail.informs.org
Fred Murphy (Chair), Temple University
Robin Lougee-Heimer, IBM
David Woodruff, University of Calif-Davis

Get involved — Volunteer your help.

Message from the Chair

Robin Lougee-Heimer, IBM
robinlh@us.ibm.com

This is Harvey Greenberg's fourth and final ICS newsletter as Editor, and I just want to say in big, bold, font, **THANK YOU, HARVEY!!!** Harvey took over as Editor in 2007, and with help from Matt MacLeod (CORA), he has taken our newsletter to new heights. Not only is it wonderfully comprehensive (which is no small feat with all the projects, prizes, conferences, member news, and general happenings we have going on) but each edition has debuted new features (check out this issue's "Researcher's Corner"▹), all delivered in a professional-quality layout that makes our newsletter a useful marketing tool for advertising who we are and what we do.

As Chair, I have been fortunate in working with Harvey's infectious energy, and passion and luckily for us he brought those same traits to bear in identifying an outstanding replacement. Taking over the helm from Harvey is Jeff Linderoth (University Wisconsin-Madison and past ICS Secretary/Treasurer). I am looking forward to more great newsletters under Jeff's leadership.

If there is not enough content in the Fall 2008 Newsletter for you, there is more going on around the Society that you will be hearing about soon. Hot topics percolating up from the recent INFORMS meeting in DC include a proposal to seriously consider establishing "special interest groups" (SIGs), the upcoming final report from the ICS Education Committee, the idea of trying to increase our student ranks by establishing a formal mentoring program, and more do-able takes on the "webinar" idea proposed at the INFORMS Seattle meeting last year, just to name a few.

It is tempting to comment on all the happenings, but I will let you read the news for yourself. My executive summary is: thanks to your involvement, ICS just keeps getting better. Volunteering with the Computing Society is a great way to learn more, meet new people with common interests, and have some fun along the way. If you would like to get tapped in, send me an email at robinlh@us.ibm.com, or better yet, come to the 11^{th} Computing Society Conference in Charleston, SC on January 11–13, 2009.

Looking forward to seeing you at ICS'09.

ICS News Incoming Editor

Jeffrey T. Linderoth has been appointed the new *ICS News* Editor. Jeff is Associate Professor of Industrial and Systems Engineering and Associate Professor of Computer Sciences at the University of Wisconsin-Madison. As you can see from his background and current activities at his web site [http://www.engr.wisc.edu/ie/faculty/linderoth_jeffrey.html], Jeff brings a great deal of expertise and enthusiasm to this job.

He will be assisted by Matt MacLeod, who has been serving as Assistant Editor, 2007–8 (see cover ▹).

COIN-OR

Robin Lougee-Heimer, IBM
robinlh@us.ibm.com

The Computational Infrastructure for Operations Research (COIN-OR, http://www.coin-or.org) is the premier website devoted to open-source software for the operations research community. Prize winning work, new project announcements, advances in existing projects, and event highlights lead the news headlines at COIN-OR.

- EPA and Sandia Team Wins the 2008 COIN-OR Cup
- New Project: Couenne for Nonconvex MINLP
- New Java Interface Supporting COIN-OR Solvers Released on SourceForge
- Binaries and RPMs Available for Select Projects
- Existing Projects Continue to Evolve
- First COIN-OR Vendor Workshop at INFORMS
- Workshop on Open-Source Software at CPAIOR
- 2008 Annual Report Available

EPA and Sandia Team Wins the 2008 COIN-OR Cup. The 2008 COIN-OR INFORMS Cup was presented by Cup Organizer, Brady Hunsaker (Google) to the U.S. Environmental Protection Agency (EPA) and Sandia National Laboratories Team of Terra Haxton, Robert Janke, Regan Murray (EPA), and Bill Hart, Jonathan Berry, Erik Boman, Robert Carr, Cindy Phillips, Lee Ann Riesen, Jean-Paul Watson (Sandia) at a celebration sponsored by IBM.

The EPA, in collaboration with the University of Cincinnati, Sandia National Laboratories, and Argonne National Laboratory, has developed the Sensor Placement Optimization Tool (TEVA-SPOT) to design contamination warning systems (CWSs) that can quickly detect contamination incidents and reduce the overall impact of terrorist attacks. The EPA partnered with member utilities of the American Water Works Association to apply TEVA-SPOT to nine utilities. These water utilities have begun installing CWSs based on these designs, and these CWSs are predicted to reduce the median impact of contamination incidents by 48 percent, and the corresponding economic-impact reduction is over $19 billion. The EPA's TEVA Research Project was a finalist in the 2008 INFORMS Edelman competition.

"COIN-OR was a key element of many of the discrete optimizers that are included in TEVA-SPOT, and the INFORMS Edelman judges highlighted the impact of COIN-OR in this project."

TEVA-SPOT can apply the PICO integer programming solver to find provably optimal sensor placements for a wide range of objectives. PICO uses COIN-OR projects, OSI, CLP, and CGL to perform general-purpose, parallel optimization of integer programs. The COIN-OR Vol software, for unconstrained facility location, was also adapted for a Lagrangian method that can compute lower bounds on the optimal sensor placement. This was combined with a fractional rounding heuristic to provide a fast, low-memory heuristic for sensor placement. This heuristic is particularly critical because water utilities need to develop CWS designs with many contamination scenarios on limited-memory workstations.

(continued on page 29 ▷)

Mathematical Programming Glossary

Allen Holder, Rose-Hulman Institute of Technology
icsMPGlossary@mail.informs.org

The ICS *Mathematical Programming Glossary* continues to fulfill its educational and research intent. Jörg Rothe, at the Institute für Informatik, Heinrich-Heine-Universität Düsseldorf, has joined the Board and has agreed to assist us in broadening The *Glossary*'s coverage of complexity theory. He has already authored a supplement on fixed-parameter tractability▷. Jörg is also working jointly with a Ph.D. student to re-write the supplement on computational complexity and to author new definitions for a Tour▷ on complexity. We welcome Jörg and appreciate his assistance. Here is our complete Editorial Board, whom I thank for their advice over the two years that I've been Editor:

Harvey Greenberg, who recently retired, is using his new found time to re-write and broaden *Myths and Counterexamples*. We all know that good examples are paramount to learning a discipline, and this resource has become an impressive collection of illustrative examples that highlight common and often subtle flaws with the surrounding folklore. Harvey is maybe uniquely qualified for this task due to his breadth and depth of experience, and his efforts are greatly appreciated. The first version of his revision is now available at http://http://glossary.computing.society.informs.org/.

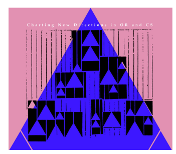
Many improvements in formatting are underway. The process of re-writing the html code to achieve a uniform standard continues. We are also moving toward a standard format for supplements. The *Glossary* is receiving just over 10,000 hits per day, and I regularly receive emails about clarifications and/or suggestions.

Please visit http://glossary.computing.society.informs.org/ to use the ICS *Mathematical Programming Glossary* and learn how you can contribute.

Leading Edge Tutorials

Rob Dell, Naval Postgraduate School
dell@nps.edu

ICS sponsored another leading edge tutorial at the Washington DC INFORMS meeting where Professor Warren Powell presented *Approximate Dynamic Programming* to an audience of approximately 70. Look for his tutorial to join the other leading edge tutorials available for download at http://computing.society.informs.org/lEdge.php.



INFORMS Journal on Computing

John Chinneck, Carleton University
editor_joc@mail.informs.org

Things are busy at the *INFORMS Journal on Computing*! In response to increased workloads, we have recently welcomed four new Associate Editors:

- Hande Yaman of Bilkent University will primarily assist John Hooker in the Constraint Programming and Optimization area.
- Theodoros Evgeniou of INSEAD and Gediminas Adomavicius of the University of Minnesota will primarily assist Alexander Tuzhilin in the Knowledge and Data Management area.
- Russell Schwartz of Carnegie Mellon University will work primarily with Harvey Greenberg in the rapidly expanding Computational Biology and Medical Applications area.

In other personnel changes, the Journal thanks departing Associate Editors Akhil Kumar of Pennsylvania State University and Rex Kincaid of the College of William and Mary as they rotate off after many years of fine service to the Journal. We are also losing Michel Gendreau, Area Editor for Heuristic Search and Learning after many years of valued leadership. But we are losing Michel for the best of reasons: he is the new Editor-in-Chief of the *INFORMS Journal Transportation Science*. David Woodruff (University of California at Davis) replaces Michel as of October 1, 2008. We are fortunate to recruit David: he is a leader in this field of research, and has lengthy previous experience as a *JOC* Associate Editor in this same field. We wish Michel success in his new role, and we bid David a warm welcome.

The one-year anniversary of my appointment as Editor-in-Chief of the Journal passed in July so there is some opportunity for reflection on this significant milestone. My first and most powerful observation is that the editors volunteering their time at the *JOC* are a very sharp, knowledgeable and erudite group.

Their understanding of topics at the interface of operations research and computer science is very deep, and they are well-connected and respected by the research community that they serve. The well-deserved reputation of the Journal rests on this fact. Second, one of the main accomplishments of my first year has been the transition of the Journal to an online manuscript submission and handling system. This has not been without hiccups, but with the able assistance of *JOC* Managing Editor Kelly Kophazi (the resident expert on the system), the system is now working smoothly and we are beginning to see some of the benefits, such as centralized records of all transactions concerning an individual paper. This will help us in reducing the time between initial submission and final decision.

In the near term, I am looking forward to the completion of the processing on the special issue on High-Throughput Optimization, and to the publication of a number of interesting papers, including a Feature Article that is in the late stages of review. Finally, let me remind you once again that the *INFORMS Journal on Computing* is the premier outlet for research in the interface of operations research and computer science. Make sure to send us your highest quality research. For further information and author guidelines visit our website at http://www.informs.org/site/IJOC/.

INFORMS DC Meeting — Record-breaking ICS Activity. . . AGAIN!

Robin Lougee-Heimer, IBM
robinlh@us.ibm.com

The Computing Society was at its best during the INFORMS 2008 DC meeting with technical sessions galore, a new "Computing Reception," tutorials, prize winners, progress on all ICS project fronts, and a blog ▹ to capture it all.

Technical Program

The old record of 81 ICS-sponsored sessions established in Seattle (2007) was blown away by a phenomenal 105-session line up in Seattle — a 30% increase! The program consisted of (i) our traditional cluster, (ii) special topics clusters, (iii) clusters co-sponsored with other subdivisions and Invited cluster organizers, and (iv) jointly-sponsored sessions in non-joint clusters. The 105 sessions appeared with ICS attribution in the program, but are not in one list on the conference website.

The ICS special topic clusters were: Bioinformatics and Systems Biology (with the Health Applications Section) [5 sessions], Constraint Programming and Operations Research [5 sessions], Open Source Software (with the Optimization Society) [12 sessions], Mixed Integer Nonlinear Programming (with Optimization Society) [4 sessions], and Metaheuristics [3 sessions]. The joint clusters were Optimization and Software (with the Optimization Society) [17 sessions], and Computational Biology (with Invited) [4 sessions]. The joint sessions in non-joint clusters were collaborations with Data Mining [6 sessions], Energy, Natural Resources, and the Environment [1 session], Information Systems [2 sessions], Optimization/Discrete Optimization [1 session], Optimization/Stochastic Programming [4 sessions], Quality, Statistics and Reliability [7 sessions], Services Science [3 sessions], Telecommunications [5 sessions], and Transportation Science and Logistics [1 session]. The traditional cluster [25 sessions] included the ICS 2008 Prize session, ICS Leading Edge Tutorial session, Recommendations from the Education Committee session (with the INFORM-ED Fora), and sessions for the Artificial Intelligence Section (which did not have a sponsored cluster).

Thanks to everyone who presented in and organized ICS-sponsored sessions, and a special thank you to the cluster organizers:

Kevin Furman, ExxonMobil

Joao Goncalves, IBM

John Hooker, Carnegie Mellon University

Mary Beth Kurz, Clemson University

Leo Lopes, University of Arizona

Robin Lougee-Heimer, IBM

Ted Ralphs, Lehigh University

Nick Sahinidis, Carnegie Mellon University

Meinolf Sellmann, Brown University, and

Mona Singh, Princeton University.

Computing Reception

In the spirit of continuous improvement, this year we tried something new. We separated the wine-and-cheese event from the Business meeting, increased the beverage order, extended the mixer's hours, lined up dozens of "ambassadors" to welcome newcomers, and invited the world to come get to know us at the "Computing Reception." Thanks to the graphic talent of Membership Committee Chair, Dick Barr (Southern Methodist University), a one-page flyer was generously distributed around the conference to promote the reception and other society events in addition to the usual one-page power point that we ask Sponsored Session Chairs to show. Besides the obvious marketing benefits, the new format meant that our business meeting was NOT held in parallel with all the other subdivision meetings on the same night, making our meeting more accessible to attendees with multiple subdivision memberships. Thanks to a generous offer by Bjarni Kristjansson (Maximal Software), color hard copies of the Spring newsletter and preprints of upcoming interviews, and other promotional materials were available at the Reception and elsewhere. Thanks to the photographic talents of Harlan Crowder, you can see some of this for yourself along with great shots from the Business Meeting at http://picasaweb.google.com/hpcrowder/Ics_washington_08.

Business Meeting

The Computing Reception segued into the Business Meeting conducted by Chair Robin Lougee-Heimer (IBM) and attended by 83 people (up from 65 last year in Seattle), according to the official roster. Reports from the officers, updates from our project leaders, and new business filled the agenda, while characters from *The Simpsons* filled the .ppt deck, keeping the fun-factor high throughout the perfectly-paced presentation.

The meeting kicked off with a surprise. Past Chair, John Chinneck was presented with a replica of the silver tankard he gave to the Society last year in instituting a ceremonious swig to inaugurate the incoming Chair. The gift was engraved with

"The Big Geek"
John W. Chinneck
ICS Chair, 06-07

in commemoration of the ICS Chair's mug inscription, "The Big Cheese."

Secretary/Treasurer Kipp Martin (University of Chicago) reported that our finances are in good shape with an estimated $27K, due primarily to the very successful ICS 2005 meeting in Annapolis organized by Bruce Golden, S. Raghavan, and Ed Wasil. In addition, this year the Student Paper Award was endowed with a generous gift of 75K Danish Kroner by the Mica Fonden of Denmark. A silver lining to the weak dollar is that the gift was valued at $15K USD (!) A special investment account was created for the endowment.

Membership Committee Chair, Dick Barr, reported 448 members as of September 2008. This is DOWN from last year and below the 500 members required for Society status. (Weird, huh? We have 30% MORE sessions, 40% MORE money. . . and 10% FEWER members. Maybe we should lose another 10% of members. <I'm kidding!!!!> Let's fix this. Sign up at http://computing.society.informs.org/join.php. Give your students a membership — it's only $1.) An on-site membership drive was held and a big thank you to the 11 new and continuing members who signed up.

Reports given by project leaders appear in this newsletter, as well an update on the *INFORMS Journal on Computing* from Editor John Chinneck.

New business included the changing of the guard. Thanks to our two Board Members and newsletter Editor whose terms are ending:

- Rob Dell (Naval Postgraduate School)
- Pascal Van Hentenryck (Brown University)
- Harvey Greenberg (retired)

and to our continuing officers: Chair, Robin Lougee-Heimer (IBM), Vice Chair, Bob Vanderbei (Princeton University), Secretary/Treasurer Kipp Martin (University of Chicago), Board Members Steve Dirkse (GAMS), Matt Saltzman (Clemson University), Jonathan Eckstein (Rutgers University), Jonathan Owen (GM).

You can read more about the DC meeting and add your own comments about the conference on our blog at http://computing.society.informs.org/serendipity, thanks to ICS Blog Master, Bill Hart (Sandia Labs). Check it out. Next up — the INFORMS 2009 Annual Meeting in San Diego, CA!

ICS 2009

John Chinneck
Bjarni Kristjansson
Matthew Saltzman

Our Biennial Meeting will be held January 11–13, 2009, at the Francis Marion Hotel in Charleston, South Carolina. The pieces of a very exciting ICS 2009 conference are rapidly falling into place. Here are some recent developments:

- We will be featuring Richard M. Karp ("Dick") and Miron Livny as plenary speakers. Karp published a landmark paper in complexity theory, "Reducibility Among Combinatorial Problems," in which he proved 21 problems to be NP-complete. He has received many major awards, such as the Turing and Lanchester prizes and the National Medal of Science, among others. Livny personifies the OR/CS interface with his work on high-throughput computation, including Condor and the Open Science Grid project. He has also won the SIGMOD Test of Time award for his seminal work on distributed databases. These are definitely speakers that you want to meet and hear!

- Microsoft has been developing a set of tools for optimization, constraint solving, and modeling. The project has been in stealth mode, but is scheduled for release soon. Members of the Microsoft Solver Foundation development team will be at the conference to present the details and to explain how this new product fits into the landscape of existing solvers and modeling systems.

- The conference volume is now in production. The schedule is tight, but we hope to have it on hand for distribution at the conference. The 400+ page book contains 24 interesting refereed articles on topics at the OR/CS interface: new developments in modeling, optimization techniques, and numerous innovative applications.

- The first ever presentation of the Harvey J. Greenberg Award for Service to ICS.

- A terrific brochure describing the conference is now available. Download a copy from http://ics09.meetings.informs.org/Brochure4.pdf and post it on your door and distribute it to your colleagues.

Abstracts for presentations are continuing to come in to the conference web site at http://www.ics2009.org. We anticipate a full schedule of interesting presentations on any and all topics at the interface of operations research and computer science. Presentations on the conference theme of "Operations Research and Cyber-Infrastructure" — the computing and communications infrastructure that supports large-scale operations research — are especially welcome.

Streams are currently being organized on the following themes (though more are very welcome):

- *Approximate Dynamic Programming*: Warren Powell (powell@princeton.edu)
- *Computational Biology and Medical Applications*: Harvey Greenberg (hjgreenberg@gmail.com) and Allen Holder (holder@rose-hulman.edu)
- *Constraint Programming*: Pascal Van Hentenryck (pvh@cs.brown.edu) and Laurent Michel (ldm@engr.uconn.edu)
- *Data Mining and Classification*: tba
- *Global Optimization*: Cole Smith (cole@ise.ufl.edu)
- *Heuristics and Metaheuristics*: David Woodruff (dlwoodruff@ucdavis.edu)
- *Information Technology*: tba
- *Integer Programming*: Ted Ralphs (tkralphs@lehigh.edu)
- *Networks*: tba
- *Open Source Software*: Bill Hart (wehart@sandia.gov)
- *Optimization*: tba
- *O.R. Cyberinfrastructure*: Bob Fourer (4er@iems.northwestern.edu)
- *Student Papers*: Robin Lougee-Heimer (robinlh@us.ibm.com)
- *Vehicle Routing*: tba
- *Vendors*: Bjarni Kristjansson (bjarni@maximalsoftware.com)

Check out the conference web site at www.ics2009.org from time to time: things are changing fast these days. We look forward to seeing you all in Charleston, rated number 3 city to visit in the USA by Condé Nast Traveler magazine!▷ If you haven't been there, January 11–13, 2009 is your opportunity to combine business with pleasure.

An Interview with Marcel F. Neuts

Marcel F. Neuts is Professor Emeritus of Systems and Industrial Engineering, University of Arizona. Among his many honors, he received the ICS Prize (1987) "for his seminal works in computational probability" (citing [9]), and was inducted into Omega Rho for having "made conspicuous contributions to Operations Research and Management Science. . . " In 1997, the *Marcel F. Neuts Prize* was established in honor of Marcel's retirement, for the best paper in *Stochastic Models*, the journal he founded. Harvey Greenberg interviewed Marcel June 28 by phone.

Q. *You entered Stanford in 1958 to study statistics. Tell me what it was like and why you worked with Sam Karlin.*

A. Very stimulating. Everyone was hard-working and deep into research. I was drawn to Prof. Karlin because of his reputation and my interest in game theory, which was one of his main interests at that time.

Q. *You arrived at Purdue in 1962, the same year they formed the world's first Department of Computer Science. Was their new focus on computing, as a part of applied mathematics, an influence on your subsequent work?*

A. No, I was not yet into computing. I thought computation was incidental once you had the mathematics.

Q. *In 1966 you taught a seminar at Purdue on algorithms for Markov renewal processes, and you later indicated [10] that you began to realize the computational challenges were much greater than you had thought. Can you recall what happened when you began to ask people about this at ORSA meetings?*

A. I naively thought everyone knew how to compute. The people with whom I spoke did not know and did not think it was worthwhile to develop computer skills. It was all about the mathematics. There was an attitude against algorithmic thinking that challenged me. When someone says that I cannot, or should not, do something, I am inclined to do it.

Q. *I believe you attended your first ORSA meeting in Fall 1966, where you presented "On the Numerical Solution of Queueing Problems?" Was it in 1967 that you introduced the first ORSA session on computational probability? How did that develop — that is, why did you organize the session? Did it achieve what you wanted?*

A. Yes, I believe that the session was the following meeting. To my astonishment, the session drew great interest. It was well attended and there was much discussion. I met people who shared my belief in the need for computational probability as a scholarly pursuit. Similar sessions became part of subsequent ORSA meetings.

Q. *In [10], you credit Descloux's review of your 1971 article [4] as having made an impression on you with respect to numerical computation — a kind of wake-up call. Can you elaborate?*

A. Actually, his advice came more in private conversation, as we were friends. The key challenge that he raised was whether solutions could be computed, even with the use of transforms. He knew, what I had not yet learned, that the computational challenge was not resolved merely by presenting some equations.

Q. *You further indicated that the works of Richard Bellman influenced you to consider computational issues. Can you elaborate on Bellman's influence?*

A. His book, *Some Vistas of Modern Mathematics* [University of Kentucky Press, 1968], had some wise comments. An example is 'Anyone who thinks computation is trivial never did it.' (Quote may not be exact.)

Q. *You also mentioned the influence of the work of Ulf Grenander, who coined the term "computational probability" as a project title, joint with IBM and Brown University. Did you have access to his sequence of reports (1968–9), or did you use his (co-authored) ensuing book [2]?*

A. I read both, and things were advancing so rapidly at that time that I learned from both.

Q. *Correct me if I'm wrong, but it seems that your first paper that was computational in its fundamental content was published in 1973 [5]. The starting point was the discretization of the M/G/1 queue that you developed with Stella Dafermos [1]. Did the numerical analysis occur after this theory, or did the concern for algorithmic feasibility motivate the discretization in the first place?*

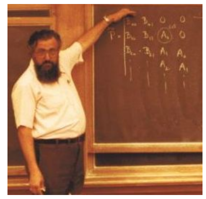
A. The latter. Stella (who was at Cornell at the time) and I developed the discretization with solvability in mind. This led to the four papers in *NRLQ*.

Q. *In 1976 you moved to the University of Delaware and continued your focus on computational probability. What attracted you to Delaware?*

A. I was offered a Chair, which gave me greater opportunity for research and collaborations with students and visitors. Also, my friend, Adi Ben-Israel, accepted a Chair the same year. V. Ramaswami, a talented mathematician and one of my star students, followed me to Delaware from Purdue to do his thesis research. Guy Latouche, then a young CS researcher from Belgium, had a visiting appointment during the next year. Within a year, David Lucantoni also joined the fold. At the University of Delaware, we formed a highly synergetic group. Together, we developed the first beginnings of what became known as the "matrix analytic methods" in probability. In June 2008, MAM6, the sixth international conference on these methods, was (enthusiastically) held in Beijing, China.

Q. *Since the publication of your seminal book [7], others have written books about computational probability. Do you draw a distinction between "**computational** probability" and "**algorithmic** probability?"*

A. No, I do not distinguish these terms. I have used 'algorithmic' to describe the field because I like the phrase, 'algorithmic thinking.' At least from Bellman's work, some applied mathematicians were thinking along these same lines.



Marcel teaches about *Matrix-Geometric Solutions* in his 1981 workshop at The Johns Hopkins University.

Q. *Is your 1995 book of problems [11] and algorithms the only one of its kind since your earlier compilation [6]? Have others used it to pick up the algorithm-teaching torch?*

A. There are exquisite algorithmic problems for teaching and research. Recent books on applied probability models include a chapter on algorithmic development. Still, and recognizing that I am an impatient fellow, I wish that algorithmic thinking would be more daring, more radical in its methods and in the scope of its genuine applications. As far as passing the "torch," I am going to Nigeria in November 2008 as member of a group of algorithmic specialists led by Professor Attahiru Sule Alfa of the University of the Manitoba, Canada. That initiative is part of the development of a pan-African Engineering school at Abuja, Nigeria. I shall bring copies of my book and I shall share my enthusiasm for algorithmic thinking with the students.

Q. *In 1985 you went to the University of Arizona. What attracted you to go there?*

A. Climate. I was also promised an opportunity to create a Laboratory for Algorithmic Research, which I co-founded with Sid Yakowitz in 1987.

Q. *Tell me more about the Lab.*

A. Its goals were to develop algorithmic thinking beyond probability and bring this way of thinking into all of applied mathematics. We attracted visitors, and I ran a weekly evening seminar. Although the Lab did not develop as much as I would have liked, we did some good. I served on an NSF panel to review needs, and I pointed out the need for algorithmic thinking, citing the Lab as an example unit.

Q. *By the mid-1980s, you had a strong handle on what computational probability is and why it is important. There seems to be two related, but distinct, goals: (1) concern for algorithmic feasibility [7], and (2) new problems and conjectures obtained from computer experimentation [8]. Is this an accurate characterization of your definition and scope of computational probability?*

A. Yes

Q. *You mentioned that, at least in the 1970s, you felt students had a really hard time inferring properties correctly from computational experiments. With hindsight, can you say more about how students progressively reacted to your algorithmic approach during your years at Purdue, Delaware, and Arizona?*

A. Let me start with some background. In Belgium, in the 1950s, homework problems or recitation sessions ceased after the first year of mathematical university studies. Courses were taught purely "ex cathedra" with no opportunity for interaction with professors or assistants. Fortunately, to safeguard my enthusiasm for mathematical problems, I discovered the books of the Dutch mathematician Fred Schuh. I bought several of his compilations of challenging analytic problems and had much fun trying to solve them. Later, Professor Samuel Karlin, the author of several excellent mathematics books, also emphasized that budding researchers should "cut their teeth" by working good problems. I learned from both and, in my own teaching, became their faithful disciple. That later inspired my book of problems. I realized that the growth that students can gain in algebra or calculus from doing easy to hard problems had no equivalent in computation. This epiphany caused me to think about exciting problems to stimulate algorithmic thinking. In the progression that I experienced, students went from them having trouble with my questions of inference from computational experiments to asking themselves, "What is the physical meaning of my numerical results?"

Q. *When you began your journey in the mid-1960s, you found ORSA to be* ***the*** *right community to raise questions and find colleagues. The INFORMS Applied Probability Society (APS) is the descendant of the ORSA Applied Probability Group (founded 1971) and the TIMS College of Applied Probability (founded 1975). Would you say APS is still a premier professional group for computational probability?*

A. The mission and the interests of APS are not strongly algorithmic. Much of what I see is mainstream stochastic modeling using classical mathematical methods, such as diffusion processes and limit theorems. Algorithmic work that makes me sit up is often done by computer scientists or general OR analysts. Much such research is done in Europe or Japan. Really significant algorithmic work is often proprietary and many persons in academe are unaware of its import.

Q. *Were you a founder of either the ORSA or TIMS group?*

A. Narayan Bhat is the founder of the ORSA Applied Probability Group. I enthusiastically supported his efforts, but I principally devoted my own time to gaining a modest place in the sun for the matrix-analytic methods. Scientific innovations should be questioned and critically examined. During the early years of the matrix-analytic methods, my associates and I all had to run the gauntlet of professional criticism. To a high degree, this is as it should be. But, to gain the place in the sun, you must be a little bit "obsessed." I tirelessly gave lecture tours in many countries and spread the algorithmic word to whoever appeared interested.

Q. *You have been a champion of education in computational probability. In 1986 you said* [8]*:*

> "With the exception of discrete event simulation, few algorithmic aspects of probability have yet entered the educational curriculum."

Do you still think this is an issue, or have others picked up the algorithmic-torch as an approach to education in applied probability?

A. There is much interest in algorithms related to the matrix-analytic methods, PH-distributions and MAPs. The ongoing research on these topics is in very able hands. It is also a varied and international effort by scholars in China, Italy, Australia, and The Netherlands. For a general perspective, one needs to look at the ongoing algorithmization of applied mathematics. That will surely continue, being driven by emerging applications.

Q. *I understand that you developed a "family" environment with your students, which was especially important for a student from abroad. This included hikes, where you discussed thesis topics, among other things. I also understand that you drove to ORSA meetings with your students, just so you could be with them and apply your Socratic method of inquiry. Did this camaraderie make your group more productive? Was your wife, Olga, a part of this family; did she interact with students?*

A. Olga and our children have fond memories of the colleagues and graduate students who visited our home over the years. We discussed many eclectic subjects. My family and I benefited from these interactions and, so I hope, did our friends. A yearly ritual was a chili con carne dinner which I prepared. We also provided a vegetarian option, although at least once I erred on the hotness scale, having used a new, fiery chili powder from Madras. Ours was a good, interesting life which we have lived to the fullest.

Q. *ICS recently established an Education Committee, and they are formulating curriculum guidelines for an OR/CS education. Do you have any advice or thoughts for them?*

A. Make algorithmic thinking fundamental. Beyond courses, have a Lab dedicated to that community. The physical part of that is a room with computers, blackboards, a library, and anything else that will stimulate interactions. The organizational part is a unit with some budget for inviting speakers and visitors. It should be an exciting atmosphere with hard-working students and faculty engaged in research and general questioning.

Q. *What advice do you have for other researchers who are considering a career in the OR/CS interface?*

A. Stimulate your imagination. Closely follow the amazing adventure of Science which humanity has been living for the past three hundred years. Enhance your professional competency. With hard work and a modicum of luck, you may contribute something of your own.

Q. *What do you like to do when you are not working?*

A. Go to theater, concerts, and lectures. Once a week, my friend and colleague, Ferenc Szidarovszky , invites friends to a dinner followed by listening to classical music. Both his musical erudition and his record and tapes collection are superb.

Q. *Is there anything else you wish to add?*

A. Since my retirement I miss having Ph.D. students. I would like to have one to work with me on very large Markov chains. Examples of this can be found in language — for example, we can model the problem of determining a person's place of learning language from the phonemes we hear.

I am reading about genetics, a fascinating scientific area. Since your question asks about my wishes, I just need another fifty years to spend in good health and with the ability to do science.

References

[1] S. C. Dafermos and M. F. Neuts, A Single Server Queue in Discrete Time, *Cahiers du Centre d'Etudes de Recherche Opérationnelle* 13:1 (1971), 23–40.

[2] W. Freiberger and U. Grenander, *A Short Course in Computational Probability and Statistics*, Springer-Verlag, 1971.

[3] M. F. Neuts, Are Many 1-1-Functions on the Positive Integers Onto?, *Mathematics Magazine* 41:3 (1968), 103–109. Note: this won The Lester R. Ford Award ⊳ of the Mathematical Association.

[4] M. F. Neuts, A Queue Subject to Extraneous Phase Changes, *Advances in Applied Probability* 3:1 (1971), 78–119. Reviewed by A. Descloux in *Mathematical Reviews*, 1972: MR0283902 (44 #1132).

[5] M. F. Neuts, The Single Server Queue in Discrete Time — Numerical Analysis I, *Naval Research Logistics Quarterly* 20 (1973), 297–304. Note: there are four parts, all in this issue.

[6] M. F. Neuts (Ed.), *Algorithmic Methods in Probability*, TIMS Studies in Management Sciences, North-Holland, 1977.

[7] M. F. Neuts, *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*, The Johns Hopkins University Press, Baltimore, MD, 1981.

[8] M. F. Neuts, An Algorithmic Probabilist's Apology, in J. Gani (ed.), *The Craft of Probabilistic Modeling: A Collection of Personal Accounts*, Springer-Verlag, 1986, 214–221.

[9] M. F. Neuts, Computer Experimentation in Applied Probability, *Journal of Applied Probability* 25:A Celebration of Applied Probability (1988), 31–43. Note: this was cited in the award of the ORSA Computer Science Technical Section Prize, 1987.

[10] M. F. Neuts, Probability Modeling in ORSA: A Personal Perspective, *Interfaces* 23:5 (1993), 13–19.

[11] M. F. Neuts, *Algorithmic Probability: A collection of problems*, Chapman & Hall, 1995.

"The aim of analysis and computation is deeper understanding of the behavior of a physical probability model. "

— Marcel F. Neuts [11]

Editor Note

Marcel has pioneered an important field in the OR/CS interface, generally called "computational probability" (though Marcel has often called it "algorithmic probability"). He had to overcome the dispiriting effects of early introductions to computations in the 1950s when tedious calculations had a very low intellectual return. It was in the mid-to-late 1960s that applying probability models, particularly Markov processes, drew Marcel to the importance of algorithmic thinking and implementation. The sharp rise in computer technology in the early 1970s and the associated advances in algorithms led to his decision "to make computational probability a primary research objective." What Marcel omits is that he entered this at some risk. Almost no one in applied probability was seriously researching algorithms, much less making it a primary focus of both research and teaching. He has given his own personal reflections on those decades of developments. [8, 10]

Dear Dr. ORCS

This is a Q&A column with questions people have about the OR/CS interface and answers provided by the ICS community.

Dear Dr. ORCS: Over my career I have, out of necessity, developed heuristics and approximations to optimization problems with the intent of speeding up the time to obtain a "close to optimal" solution and fostering more what-if exercises. Many years ago, in the world of much slower computers, reducing computation times by several orders of magnitude had noticeable, palpable effects that decision-makers and users could appreciate. For example, in the old days, reducing run times by a factor of 30 might reduce a run from 30 hours to 1 hour.

Nowadays, that reduction might be from 30 seconds to 1 second and conveying and appreciating that significant difference is more problematic. So I had a couple of questions on my mind that I just put out for discussion:

1. Is the speed of modern computers reducing the search for better actionable heuristics because of a conception that there is less of a need for faster algorithms?
2. Would I and other professionals have taken the time to develop heuristics twenty or thirty years ago if computers back then were as fast as they are today?
3. Is there a good way to explain to management that a significant time reduction is still of interest and good for promoting what-if analyses even if, in most of the cases, the enterprise runs the heuristic reduces run times from 1 minute to 1 second or 3 minutes to 10 seconds?

— Meyer Kotkin

Karla Hoffman: *First, I commend you on your efforts to provide decision makers with fast approximate solutions in a timely manner and fostering what-if exercises. Working to solve* *difficult problems by being willing to develop algorithms and software is exactly what has made the profession of OR so successful.*

So, to answer your questions about heuristic solutions, I believe that there will always be a role for such technologies in our tool box. As we improve our exact algorithms and prove that we can assist decision makers in a better understanding of their complex operational problems, we are provided with greater challenges. Inevitably, the manager would like to expand the scope and scale of the problem being solved. Thus, we will continue to be forced to rethink and improve our current techniques. And, at least for now, there are many problems for which only by understanding the underlying structure are we capable of obtaining usable solutions.

Equally important, when our algorithms become fast enough, they become subroutines in other algorithms. Thus, the heuristics that were originally designed to provide good solutions to hard problems are now being used within our general purpose optimization codes to provide good bounds, thereby enabling exact algorithms to succeed. I would point out, however, that our ability to solve these large, difficult problems quickly with off-the-shelf software, although assisted by the speed-up in computing times, has been accomplished mostly by our extraordinary improvements in the algorithms themselves.

There are also many examples where we are still incapable of solving, to proven optimality, a given instance of a difficult optimization problem. We know that certain problem structures are especially difficult for general purpose optimization software (for example, traveling salesman, job shop scheduling, stochastic integer optimization). Such problems are often amenable to heuristic approaches that provide very good solutions. Also, there are problems for which our general purpose codes will only be successful when the user "assists" these algorithms with user-built heuristics or cutting planes specific for the given problem structure. Thus, your skills are extraordinarily useful in making our general-purpose codes capable of tackling our hardest problem structures.

I would also like to address the question of why we might want techniques that solve in minutes, or even seconds, rather than hours:

1. *When one can solve problems this quickly, one is able to perform sensitivity analysis on the results, and to determine the robustness of the solution to some of the assumptions.*

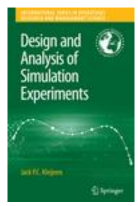
2. *Often, one needs to create a prototype model quickly and provide feedback to the user so that one can verify that the approach is correct.*

3. *And, of course, there are times when solution speed is critical (for example, think online responses such as MapQuest® or real-time scheduling).*

Thus far, I have discussed the use of heuristics to assist in solving difficult large problems. There are, of course, many problems that are more routine and are solved within seconds or minutes by off-the-shelf software. Even in these instances, one may wish to use a heuristic when one simply does not need an optimal solution to the problem due to the inaccuracy of the available data, or because the amount of money at stake does not justify the cost of procuring and using a state-of-the art software package.

So, my advice is to continue working with decision makers to solve their problems. Use all of the tools available, and where necessary, build new tools. Improvements in computing power coupled with the development of better algorithms and software has enabled our very impressive successes and, thereby, continues to provide us with new important challenges.

Book Review



Design and Analysis of Simulation Experiments, Jack P. C. Kleijnen, Springer ▹, 2008. (▹BibTeX entry)

Reviewed by Thomas Bartz-Beielstein, Faculty of Computer Science and Engineering Science, Cologne University of Applied Sciences.

Kleijnen's books, e.g. "Statistical Tools for Simulation Practitioners" from 1987 [1], are references for simulation practitioners working in industry, management, computer science, and many other disciplines. Unfortunately, these books have been out of print for many years. New statistical methods, namely *design and analysis of computer experiments* [7] (DACE) became popular, and new aspects, such as *multivariate* simulation output were considered important over the last 20 years.

In the meantime, Kleijnen continued his research and has published more than 200 articles. Now he has written a successor of his seminal books on simulation, so that a central source of information is available.

Overview

Chapter 1 introduces basic terminology used in the book and provides answers to questions like "What is simulation?" or "What is DASE?". DASE stands for *design and analysis of simulation experiments* — an acronym closely related to DACE.

Chapter 2 presents basics from regression analysis and designs for experiments. It starts with a simple metamodel $\mathbf{y} = \mathbf{X}\beta + \mathbf{e}$. Most of the following sections discuss properties of the least squares estimates $\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}$. This discussion includes consequences of violations of the normality assumption, properties of t statistics, and the treatment of replicates. Design considerations are discussed next. Design matrices are illustrated in a very comprehensive manner. Different designs types are presented in these sections.

The third chapter entitled "classic assumptions revisited" comprehends approx. 30 pages that present the author's experiences from simulation and optimization studies. This chapter is very instructive, because it discusses problems that occur in many simulation experiments:

- How realistic are the classic assumptions, and how can they be tested?
- Can the violation of these assumptions be repaired?
- If the violations cannot be repaired, what should be done?

Since *least squares* (LS) is a mathematical criterion, it does not require a normal distribution. However, additional statistical tests are based on certain statistical properties.

Bootstrapping, which requires a representative sample of the underlying distribution, is discussed. If simulation is very expensive and only a few runs are possible so that a representative sample cannot be obtained, parametric bootstrapping is recommended. Tests for constant variances are given. Variance heterogeneity is also mentioned for Kriging models (p. 147). Using a result from C. R. Rao, Kleijnen concludes (p. 77) that multiple simulation outputs can still be analyzed through *ordinary least squares* (OLS). This simplifies the situation for simulation practitioners, because they can apply classical statistical tests for the regression parameters per type.

Simulation optimization is introduced in chapter 4. It starts with the classic *response surface methodology* (RSM). Multiple outputs, constraints, and risk analysis are considered in this chapter. It also contains a description of *Latin hypercube sampling* (LHS).

Kriging basics are introduced in chapter 5. Differences between classical linear regression models and ordinary Kriging are outlined. LHS is presented as the default design technique for DACE. This is in accordance with the presentation given in [7]. Sacks et al. describe LHS as adequate for DACE, because LHS designs are computationally cheap, can cope with many variables, and provide a systematic way of discovering scientifically surprising behavior.

Screening, i.e., seeking for really important factors among the many factors, plays a significant role in simulation and optimization. Chapter 6 discusses screening methods with a focus on *sequential bifurcation* (SB).

The epilogue (chapter 7) gives a very short summary of the book.

Discussion

This book covers the relevant topics in simulation accompanied with references to authoritative publications. Since space for this review is limited, I will concentrate my discussion on the following aspects:

1. Classical linear regression vs. Kriging
2. Designs
3. Assumptions
4. Importance

Linear regression vs. Kriging. Kriging has gained a tremendous popularity in recent years. Kleijnen states (p. 140): "Although Kriging in random simulation is still rare, I strongly believe that the track record Kriging achieved in deterministic simulation holds promise for Kriging in random simulation!" Why should simulation practitioners still use classical regression? Answers to this central question are given in the book, but they are a little bit hidden in the text. Kleijnen claims that linear regression models can be used for local fitting, e.g., when searching for the optimum input combination, whereas Kriging is better suited for global fitting. Kriging and related nonlinear models may give better predictions than classical linear regression models, but "these alternatives are so complicated that they do not help the analysts better understand the underlying simulation model — except for sorting the simulation inputs in order of their importance." (p. 99)

On p. 147, one can read: ". . . we give examples of Kriging predictions [. . .] that are much better than the regression predictions. (Regression metamodels may be useful for other goals such as understanding, screening, and validation [. . .])".

However, Kriging might also be helpful in understanding and screening. I compare results from different models (classical regression, Kriging, classification and regression trees), whenever this is possible. Kleijnen applied Kriging to random simulation, I have also successfully applied Kriging to random optimization.

Design considerations. Kleijnen discusses optimal designs and mentions several optimality definitions. However, selection of optimal designs in practice is a "chicken or the egg causality dilemma": the optimal design depends on the regression model, but how can practitioners chose a suitable model *a priori*, i.e., before the simulation is performed and no experimental results are at hand? And, the situation is even worse: Santner et al. claim that LHS is popular in DACE, not because it is superior to other designs, but because it is easy to implement and understand [9]. Obviously, design considerations are not trivial. Unfortunately, the popular *one-factor-at-a-time* (OAT) design is not very effective and efficient. Kleijnen writes (p.7): "In practice, however, many analysts keep many inputs constant, and experiment with a few factors only. Another example of inferior practice is changing only one input at a time (while keeping all other inputs fixed at their so-called base values)." At the first sight, this statement is in contrast to Saltelli et al.'s rating of OAT designs [8]. They propose an elementary effects method which "is conceptually simple and easy to implement. It belongs to the class of OAT designs . . .". But Saltelli et al. consider different goals, namely *sensitivity analysis* (SA). On page 124, DASE and SA are compared: "SA may use DASE, because DASE gives better answers; i.e., the common-sense approach of changing one factor at a time gives estimators of factor effects that have higher standard errors, and does not enable estimation of interactions among factors . . .”

Kleijnen presents sequential designs as an alternative for LHS, because sequential statistical procedures are known to be more efficient and computer experiments proceed sequentially. However, he does not report problems related to sequential designs, e.g., that there may be a tendency for design sites to “pile up” [7].

Assumptions revisited. Assumptions from classic linear regression such as white noise and only univariate output “usually do not hold.” (p.73). This observation is typical for many real-world simulations. While discussing sequential bifurcation, Kleijnen notes that theoretically, the SB procedure does not satisfy the classical statistical assumptions. Nevertheless, numerical results look promising (p. 165). I recommend reading Gary Klein’s enlightening article [2] entitled “The Fiction of Optimization” that discusses these discrepancies between theoretical assumptions and the situation in field experiments. Klein claims that he has “not identified any decision researcher or analyst who believes that these [theoretical] assumptions will be met in any setting, with the possible exception of the laboratory or the casino.”

Importance. Importance can be relative — this is mentioned on p. 31: “I point out that a factor may be significant when tested through the t statistic [. . .], but may be unimportant.” (p.31 and also p.62). Kleijnen discusses, without explicitly mentioning, the large n problem which is well-known in philosophy of science [4]. Results that are statistically significant, e.g., results from t tests, are not automatically scientifically meaningful. Regarding *importance* also other problems might occur, e.g., factors that are statistically unimportant in the first phase of simulation might become important at a later stage.

A few additional notes

The classic 1996 publication [3] and Santner et al.’s 2003 book [9], which would be my first choices for Kriging designs discussed in section 5.4, are not mentioned in this section. For example, chapters 5 and 6 in [9] discuss space-filling designs and designs based on other criteria, e.g., maximum entropy.

Important is Kleijnen’s differentiation between strategic and tactical aspects (p. 9). Tactical issues such as “How long should a simulation run be continued?” or “How accurate is the resulting value of the estimator?” arise only in stochastic simulation. Strategic issues, e.g., “How to chose an adequate simulation model?” arise in deterministic and stochastic simulation. This book deals with strategic issues, tactical issues are discussed in the first part of the 1987 book [1].

It would be very interesting to read a summary of the most important open research questions from Kleijnen’s perspective. So, I was a little bit disappointed that the book was already finished after turning page 171.

Some rather technical remarks. Roman numerals are used for the resolution. This is standard notation in *design of experiments* (DOE), but not explained in this book. A sentence like “In a resolution R design no p-factor interactions is aliased with another effect containing less than $R-p$ factors” [5] might be helpful. The concept of aliasing is introduced later (on p. 44). Minor modifications in the structure of these sections might improve the comprehensibility of DOE concepts. Although Kleijnen discusses the most important aspects of factorial and fractional factorial designs, it might be useful having a book like [5] at hand to get deeper insight into this rather technical material.

Formatting could be improved (table captions should appear above the table, figures could be scaled better). But this is only a minor point.

There are more than 400 references listed in the book. It would be nice if an *apalike* bibliography style, i.e., a style which includes some hints about the authors, would have been used. Again, this criticism is only of minor importance.

Summary

This book was written for researchers, students, and mature practitioners who get valuable hints for their projects and requires basic knowledge of simulation and mathematical statistics. It summarizes results in a very compact manner and collects material that is scattered over numerous publications. It transforms ideas from statistics to simulation and optimization.

This book is a valuable source for instructors. It contains many examples with references to both toy and real-world problems. Some material in the book was used to teach a course “Simulation for Logistics” at the Technical University Eindhoven. Course materials are available on the author’s web pages. Exercises with solutions are given. Instructions for readers (recommended chapters) are given, but only very briefly.

The book complements Kleijnen’s seminal books on simulation (including new topics like Kriging and multivariate output) and summarizes research results from several hundred publications. It does not provide any rigorous proofs such as [9] or [6], but gives hands-on support. Although the *theoretical* concept of regression is rather simple, its *application* to real-world application requires an extensive knowledge and experience. This book imparts experience from one of the leading experts in this field. It is definitively an up-to-date reference for simulation and optimization practitioners. I do not know any other book which does this better.

References

[1] J. P. C. Kleijnen. *Statistical Tools for Simulation Practitioners*, Marcel Dekker, New York, NY, 1987.

[2] G. Klein, The fiction of optimization, In G. Gigerenzer and R. Selten, editors, *Bounded Rationality: The Adaptive Tool-*

box, pages 103–121, MIT Press, Cambridge, MA, 2002.

[3] J. R. Koehler and A. B. Owen, Computer experiments, In S. Ghosh and C. R. Rao, editors, *Handbook of Statistics*, volume 13, pages 261–308, Elsevier Science, New York, NY, 1996.

[4] D. G. Mayo and A. Spanos, Severe Testing as a Basic Concept in a Neyman-Pearson Philosophy of Induction, *British Journal for the Philosophy of Science*, 57(2):323–357, 2006.

[5] D. C. Montgomery, *Design and Analysis of Experiments*, John Wiley & Sons, New York, NY, 5th edition, 2001.

[6] F. Pukelsheim, *Optimal Design of Experiments*, John Wiley & Sons, New York, NY, 1993.

[7] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, Design and analysis of computer experiments, *Statistical Science*, 4(4):409–435, 1989.

[8] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola, *Global Sensitivity Analysis*, John Wiley & Sons, New York, NY, 2008.

[9] T. J. Santner, B. J. Williams, and W. I. Notz, *The Design and Analysis of Computer Experiments*, Springer, New York, NY, 2003.

Researcher's Corner

The following researchers recommend readings (other than their own) that they consider worthwhile for us to examine.

Jan Karel Lenstra: I was asked to recommend a few publications as worthwhile reading in the OR/CS interface. Here are five classics from the past. Don't blame me for going back half a century.

1. E. W. Dijkstra (1959), A note on two problems in connexion with graphs, *Numerische Mathematik* 1:1, 269–271.

   Two efficient algorithms, efficiently presented.

2. E. L. Lawler and D. E. Wood (1966), Branch-and-bound methods: A survey, *Operations Research* 14:4 , 699–719.

   A great survey, more than the sum of its parts.

3. R. M. Karp (1972). Reducibility among combinatorial problems, In R. E. Miller and J. W, Thatcher (eds.), *Complexity of Computer Computations*, Plenum Press, New York, 85–103.

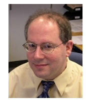
   Cook set the stage, Karp filled it and redefined combinatorial computing.

4. N. Christofides (1976), Worst-case analysis of a new heuristic for the travelling salesman problem, Management Science Research Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

   Trivial once you see it, and Nicos saw it. Still the best we can do.

5. M. Grötschel, L. Lovász, and A. Schrijver (1981). The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica* 1:2, 169–197.

   Changed the landscape of optimization.



James B. Orlin: Here are some of the publications that I find most useful when carrying out new research.

1. The NP-completeness columns of David Johnson (1981–2007). http://www.research.att.com/~dsj/columns/

   David Johnson has written 26 different NP-completeness columns since completing his book on NP-completeness with Mike Garey. The most recent column was published in 2007. They are extremely well written, and wonderful resources for the community. They cover a range of topics relating to NP-completeness, other complexity classes, the complexity of approximations, and more. They are also a very rich resource for references.

2. A. Schrijver (2004), *Combinatorial Optimization: Polyhedra and Efficiency*, Springer-Verlag, Berlin, FRG.

   Lex Schrijver has combined his three books on combinatorial optimization into a single 3-volume set which is both affordable and comprehensive. His style is perhaps more terse than many readers would like. But it has the advantage of making the nearly 1,400 pages of text cover even more material than one might expect. The book is at Amazon ▹. You can also download his free, book-like, course notes (223 pages), *Combinatorial Optimization*, at http://homepages.cwi.nl/~lex/files/dict.pdf.)

Editor's note: I asked for publications other than his own, but the following is enormously useful in this context:

3. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993.

Ward Whitt: Here are Basic References on Computational Probability:

1. P. Bratley, B. L. Fox, and L. Schrage (1987), *A Guide to Simulation*, second edition, Springer-Verlag, New York, NY.
2. P. Glasserman (2004), *Monte Carlo Methods in Financial Engineering*, Springer-Verlag, New York, NY.
3. W. K. Grassman, ed. (2000), *Computational Probability*, Kluwer Academic Press, Norwell, MA.
4. F. V. Jensen (1997), *Introduction to Bayesian Networks*, Springer-Verlag, New York, NY.
5. L. P. Kaelbling, M. L. Littman, and A. W. Moore (1996), Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research* 4, 237–285.
6. G. Latouche and V. Ramaswami (1999), *Introduction to Matrix Analytic Methods in Stochastic Modelling*, ASA-SIAM Series on Statistics and Applied Mathematics, Philadelphia, PA.
7. R. R. Motwani and P. Raghavan (1996), Randomized Algorithms, *ACM Computing Surveys* 28:1, 33–37.
8. M. L. Puterman (1994), *Markov Decision Processes*, John Wiley & Sons, Hoboken, NJ.
9. W. J. Stewart (1994), *Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, Princeton, NJ.

Harvey Greenberg: I find the following provide useful background, and they are available ***free*** of cost.

1. S. P. Bradley, A. C. Hax, and T. L. Magnanti (1977), *Applied Mathematical Programming*, Addison-Wesley. Available at http://web.mit.edu/15.053/www/.
2. E. Çinlar and R. J. Vanderbei (2000), *Mathematical Methods of Engineering Analysis*. Available at http://www.princeton.edu/~rvdb/506book/book.pdf.
3. R. B. Cooper (1981), *Introduction to Queueing Theory*, North Holland. Available at http://www.cse.fau.edu/~bob/publications/IntroToQueueingTheory_Cooper.pdf
4. L. Devroye (1986), *Non-Uniform Random Variate Generation*, Springer-Verlag. Available at http://cg.scs.carleton.ca/~luc/rnbookindex.html.
5. P. Kall and S. W. Wallace (1995), *Stochastic Programming*, John Wiley & Sons. Available at http://home.himolde.no/~wallace/.
6. S. P. Meyn and R. L. Tweedie (1993), *Markov Chains and Stochastic Stability*, Springer-Verlag, Available at http://probability.ca/MT/
7. H. S. Wilf (1986), *Algorithms and Complexity*, Prentice-Hall. Available at http://www.math.upenn.edu/~wilf/AlgComp.html.

Approximate Dynamic Programming: A Melting Pot of Methods

Warren B. Powell ▷
Princeton University (▷BibTEX entry)

Warren Powell is Professor of Operations Research and Financial Engineering at Princeton University, where he has taught since 1981. He is director of CASTLE Laboratory which specializes in the solution of large-scale stochastic optimization, with considerable experience in freight transportation. This work led to the development of methods to integrate mathematical programming and simulation within the framework of approximate dynamic programming, summarized in his book *Approximate Dynamic Programming: Solving the Curses of Dimensionality* [Wiley, 2007].

Stochastic optimization addresses the problem of making decisions over time as new information becomes available. This challenge arises in a number of disciplines, including operations research, economics, artificial intelligence and engineering. These fields each introduce different modeling issues and computational challenges. These issues, combined with the fundamental complexity of making decisions under uncertainty, has made stochastic optimization one of the richest and most challenging fields in applied mathematics.

Each of these disciplines has worked to solve the problem of making decisions over time, in the presence of different forms of uncertainty, within the context of their problem domain. It is perhaps not surprising, then, that there has been a certain amount of rediscovery of similar concepts under different names. Approximate dynamic programming, reinforcement learning, neuro-dynamic programming, optimal control and stochastic programming are all variations on a theme representing similar ideas discovered from the perspectives of different fields, often with different languages and notation.

As so often happens with languages, there is more to the differences than just words and notation. The more significant differences are the nature of the problems that each field faces. As a result, each field has discovered tricks and techniques that address the issues that arise within their problem domain. It is here that the fields have something to learn from each other.

In this article, I am going to show that the fundamentals of Markov decision processes, stochastic programming, control theory, and yes, even decision trees, can be combined within a general framework that integrates simulation and machine learning. The result is a scalable set of methods that are providing practical solutions to industrial-strength problems.

Notation

To describe our ideas, we need a notational system, which inevitably means making choices among the communities that are contributing to these problems. Choosing notation typically involves making a compromise between choosing variables that seem the most intuitive, while recognizing that notation is a language, and there is significant value in using notation that is familiar to the largest possible community. Somewhat more problematic is that notation tends to be associated with the characteristics of the problems that a community works on.

There is not enough space to address notational issues with any care (for a detailed discussion, see [5, Chapter 5]). I use S_t as the traditional variable for state, and x_t for the usual (in operations research) vector for decision variables. Decisions are made in discrete time, but activities (the arrival of information and the movement of resources) are made in continuous time. Time starts at $t = 0$, and decisions are made at $t = 0, 1, 2, \ldots$, which also corresponds to when we measure the state of the system (we only measure the state to make a decision). We let W_t be the information arriving during the time interval from $t-1$ to t, which allows us to claim that any variable indexed by t is known (deterministically) at time t. We note that this contrasts with the conventional style of the control theory community, where W_t would refer to the information arriving between t and $t + dt$ (in continuous time).

Central to dynamic systems is describing how it evolves from t to $t + 1$. I use the classical concept of a transition function, which is represented using

$$S_{t+1} = S^M(S_t, x_t, W_{t+1}).$$

The function $S^M(\cdot)$ goes under various names: "plant model," "plant equation," "system model" or just "model." It is more common to use $f(\cdot)$ for the transition function, but this uses up another valuable letter of the alphabet. The MDP community most commonly uses the one-step transition matrix $p(s'|s, x)$, which gives the probability of moving to state s' given that we are in state s and take action x. The OR community prefers to use systems of equations such as

$$A_t x_t - B_{t-1} x_{t-1} = b_t.$$

In many applications, these matrices are extremely large, and may be very difficult to write out explicitly. As a result, it is common in the control community to simply assume there is a function such as $S^M(\cdot)$, unless there is some specific need to exploit its properties.

Our challenge is to make decisions. If we were solving a traditional math programming-based model, we might formulate the problem as

$$\min_{x_0,\ldots,x_T} \sum_{t=0}^{T} \gamma^t c_t x_t$$

subject to various constraints on x_t including the linking constraints above. Here, γ is a discount factor. The difficulty is that this formulation cannot handle uncertainty in the form of an evolving information process, with decisions that are allowed to adapt to the process.

What we really want are decisions that adapt to the information as it arrives, but without peeking into the future. The stochastic programming community has learned how to formulate these problems using *nonanticipativity* constraints (see [2] for a complete presentation). But the resulting models are typically too large to be solved, and authors generally simplify the problem by limiting the number of outcomes in the future. So this leaves us with the question — how do we make decisions?

Making Decisions

The dynamic programming community addresses this problem by proposing to design a decision function $X^\pi(S_t)$ which returns a decision x_t which we assume is in a feasible region $\mathcal{X}_t$ (which might depend on S_t). There is typically a family of functions which we index using $\pi \in \Pi$, where π refers to a policy (to use the language of dynamic programming). Policies come in many flavors.

It is more conventional in this community to maximize a reward (we use a contribution) $C(S_t, x_t)$ that depends on the state and action (it might also depend on random information W_{t+1}). In this setting, our problem is to find the best policy π (or decision function X^π) that solves

$$\max_{\pi\in\Pi} \mathbb{E}\left(\textstyle\sum_{t=0}^{T} \gamma^t C(S_t, X^\pi(S_t))\right), \qquad \text{(WP.1)}$$

where $\mathbb{E}$ denotes the expected value, and we assume that we have some guarantee that a solution exists. If we use a properly designed decision function that depends on S_t, and if S_t is purely a function of $W_1, \ldots, W_t$, then the resulting decisions are automatically nonanticipative, which is why these terms typically do not arise in the dynamic programming or control theory communities.

There are numerous ways to create a decision function that represents a policy.

Myopic Policies. A myopic policy is any decision rule that does not attempt to project into the future. For example, we might simply solve problems that maximize the contribution in each period,

$$X^\pi(S_t) = \operatorname*{argmax}_{x_t\in\mathcal{X}_t} C(S_t, x_t).$$

As stated, this is just a single policy with a solution that might be good enough in some applications, but may be quite poor. There are numerous examples in engineering practice where people will play with the contribution function, adding bonuses and penalties to the contributions to try to get the model to produce good long-term results. This strategy might be written

$$X^\pi(S_t) = \operatorname*{argmax}_{x_t\in\mathcal{X}_t} C^\pi(S_t, x_t).$$

Here, $C^\pi(S_t, x_t)$ is parameterized by the various bonus and penalties, and now the challenge is to find the best values for these parameters.

Dynamic Programming. The most classical way of solving the objective function in equation (WP.1) is through dynamic programming, where we write

$$V_t(S_t) = \max_{x_t\in\mathcal{X}_t} \left\{C(S_t, x_t) + \gamma\, \mathbb{E}\left(V_{t+1}(S_{t+1})\,|\, S_t\right)\right\}. \qquad \text{(WP.2)}$$

This is generally known as Bellman's equation, the Hamilton-Jacobi equation, the HJB equation (to cover all our bases), or just the optimality equation.

"The textbook approach to solving* (WP.2) *is to solve this equation for each state S_t, and step backward through time. . . This is the 'curse of dimensionality' that is so widely cited as a reason why 'dynamic programming does not work'."

The textbook approach to solving (WP.2) (see [7] and its predecessors) is to solve this equation for each state S_t, and step backward through time (value iteration for infinite horizon problems does basically the same thing, although it is presented as an iteration counter rather than stepping backward through time). The problem here is that for many applications, S_t is a vector (for example, the number of different types of products in inventory), in which case the size of the state space grows exponentially in the number of dimensions. This is the "curse of dimensionality" that is so widely cited as a reason why "dynamic programming does not work."

Of course, this technique does work for some applications. But for the vast majority of real-world problems (not sure how to verify this claim), there are actually three curses of dimensionality: the state space, the outcome space (which determines the complexity of the expectation), and the action space (that is, the feasible region $\mathcal{X}_t$). The problem with so-called classical dynamic programming is that it assumes that you are using a lookup-table representation for the value function $V_t(S_t)$ (that is, there is a distinct value for each discrete state S_t). The method also assumes the expectation can be solved exactly, and it assumes that you can evaluate each action separately.

For many applications, the expectation cannot be computed exactly, and the vector x has to be chosen with one of a wide range of algorithms that have evolved from the math programming community. Needless to say, the usefulness of dynamic programming is looking quite limited.

Stochastic Programming. Stochastic programming evolved out of the mathematical programming community when interest grew (starting in the 1950's) to introduce uncertainty. A large body of research has evolved to solve the so-called two-stage problem that can be generally written as

$$\min_x \mathbb{E}(F(x, W)).$$

In the stochastic programming community, this is most commonly written

$$\min_{x_1} c_1 x_1 + \mathbb{E}(Q(x_1)) \tag{WP.3}$$

subject to various constraints. $Q(x_1)$ is referred to as the *recourse function* and is given by

$$Q(x_1, \omega) = \min_{x_2} c_2(\omega) x_2(\omega)$$

also subject to various constraints that depend on a random outcome ω. x_1 refers to the first-stage decisions while x_2 refers to the second stage. Note that $Q(x_1)$ is comparable to the value function of dynamic programming, except that x_1 is used as the state variable.

This general idea can be extended to multistage problems, but here is where the stochastic programming community splits. One branch uses the concept of scenario trees, while the other uses Benders decomposition. With scenario trees, the state of the system consists of the entire history, which explodes quickly in size, quickly producing intractably large problems.

With Benders' decomposition, equation (WP.3) is replaced with

$$\min_{x_1} c_1 x_1 + z \tag{WP.4}$$

subject to the same constraints on x_1 as before plus

$$z \leq \alpha_i + \beta_i^T x_1, \quad i \in \mathcal{I}^n, \tag{WP.5}$$

where α_i is a set of scalars and β_i is a set of vectors over the set $\mathcal{I}^n$ which is generated iteratively by the algorithm. Here, the recourse function is replace by a series of cuts that are generated by solving the dual of the problem for the next time period. A particularly powerful algorithm is the stochastic decomposition algorithm [3] (see also [2]), which offers a convergence proof (if the only source of randomness is in the right hand side constraint).

Approximate Dynamic Programming

There are three views of approximate dynamic programming: 1) a framework for solving complex dynamic programs, 2) a framework for making simulations more intelligent, and 3) a decomposition technique for very large-scale deterministic math programs. One of our messages is that ADP is a valid method for solving deterministic or stochastic, multistage mathematical programs.

There are many variations of approximate dynamic programming. This presentation only briefly illustrates the basic ideas.

General Strategy. At the risk of oversimplifying the diversity of algorithmic strategies that fall under the name approximate dynamic programming, the central idea of ADP is to replace the value function $V_{t+1}(S_{t+1})$ with some sort of statistical model that we call $\bar{V}_{t+1}(S_{t+1})$. Then, instead of solving (WP.2), we would make decisions by solving

$$X^\pi(S_t) = \underset{x_t \in \mathcal{X}_t}{\operatorname{argmax}} \left\{ C(S_t, x_t) + \gamma \, \mathbb{E} \left(\bar{V}_{t+1}(S_{t+1}) | S_t \right) \right\}. \tag{WP.6}$$

Then, instead of stepping backward through time (forcing us to compute the value of being in every state), we step forward through time, sampling a single sequence of states. We use information gathered from solving these decision problems to update the value function approximation.

"Formulating and estimating a value function approximation is at the heart of any ADP algorithm."

There are many ways to estimate the value function approximation. The simplest way to illustrate the idea is to use a lookup-table representation, where there is a value $\bar{V}_t(S_t)$ for each state S_t. Assume we are in state S_t^n at iteration n of our algorithm, and let $\bar{V}_t^{n-1}(S_t)$ be our estimate (from the previous iteration) of the value of being in each state. Finally let $\hat{v}_t^n$ be the value from solving the maximization problem in equation (WP.6) (this would be the value if we used max instead of argmax). We could update the value function approximation using

$$\bar{V}_t^n(S_t^n) = (1 - \alpha_{n-1}) \bar{V}_t^{n-1}(S_t^n) + \alpha_{n-1} \hat{v}_t^n, \tag{WP.7}$$

where α_{n-1} is a stepsize between 0 and 1. In practice, lookup-table representations are not practical, but they illustrate the basic idea.

We are not out of the woods. The difficulty is the other two curses of dimensionality, which means computing the expectation and then solving the resulting optimization problem (which itself may be fairly hard, especially if x_t is integer and we face a difficult integer programming problem). We address this problem by using a concept that goes under many names, but I prefer the name first proposed in [12] which is the post-decision state variable. This is the state of the system immediately after a decision has been made, but before any new information has arrived. In decision trees (where you make a decision at a decision node), this is called the outcome node. It has also been called the end-of-period state [4] and the after-state variable [10]. All that matters is that it is a deterministic function of S_t and x_t.

The post-decision state takes different forms depending on the nature of the application. Some examples are:

Blood management. Let R_{tb} be the amount of blood of type b on hand to be used during week t. Let x_{tb} be the amount of blood of type b used in week t, and let $\hat{R}_{t+1,b}$ be random donations of blood that will be available to be used during week $t+1$. R_t is the pre-decision state. $R^x_{tb} = R_{tb} - x_{tb}$ is the post-decision state, while $R_{t+1,b} = R^x_{tb} + \hat{R}_{t+1,b}$ is the next pre-decision state.

Managing expensive equipment. Let a_t be the vector of attributes of a business jet, which includes attributes such as location, repair status and time of availability. If we make the decision to move the aircraft, it might arrive late due to weather delays, and an equipment problem might arise. The post-decision state a^x_t might assume the aircraft will arrive on time with no equipment problems. The next pre-decision state, a_{t+1}, would reflect the weather delays and equipment problems.

Let $S^x_t = S^M(S_t, x_t)$ be the post-decision state (that is, the state after x_t has been determined). Instead of coming up with a value function approximation around S_t, we will instead come up with a value function approximation around S^x_t. If we were using a lookup-table representation, the update in equation (WP.7) becomes

$$\bar{V}^n_{t-1}(S^{x,n}_{t-1}) = (1-\alpha_{n-1})\bar{V}^{n-1}_{t-1}(S^{x,n}_{t-1}) + \alpha_{n-1}\hat{v}^n_t.$$

Note that we are using $\hat{v}^n_t$ to update the value function around the previous post-decision state $S^{x,n}_{t-1}$. The change is subtle but significant. Now, our policy looks like

$$X^\pi(S_t) = \arg\max_{x_t \in \mathcal{X}_t} \{C(S_t, x_t) + \gamma \bar{V}_t(S^x_t)\}. \tag{WP.8}$$

where $S^x_t = S^{M,x}(S^n_t, x_t)$ is the post-decision state if we are currently in state S^n_t and if we were to take action x_t.

Note that we no longer have an expectation in the decision problem in equation (WP.8). This is a major change that also helps us solve the third curse of dimensionality. We are primarily interested in problems where x_t is a vector. Depending on the characteristics of the problem, we might naturally want to solve our decision problem as a linear program, nonlinear or integer program, or using our favorite metaheuristic. If we use a metaheuristic, the value function approximation can take virtually any form (as long as it can be quickly computed). If the contribution function $C(S_t, x_t)$ is nonlinear, we could use a nonlinear value function approximation.

After we solve our problem at time t, we simulate our way to the next state by randomly sampling the exogenous information (which we represent using $W_t(\omega^n)$), and then compute

$$S^n_{t+1} = S^M(S^n_t, x^n_t, W_{t+1}(\omega^n)).$$

This is pure simulation — nothing fancy here, and it scales to really large problems.

Value Function Approximations. Formulating and estimating a value function approximation is at the heart of any ADP algorithm. A popular strategy in the ADP literature is to use linear regression, where the independent variables are referred to as *basis functions* $\phi_f(S_t)$, $f \in \mathcal{F}$, where each basis function $\phi_f(S_t)$ is also known as a feature. A basis function is any scalar function of the state variable. If we use this strategy, the value function approximation would look like

$$\begin{aligned} V_t(S_t) &\approx \bar{V}_t(S_t|\theta) \\ &= \sum_{f\in\mathcal{F}} \theta_f \phi_f(S_t). \end{aligned}$$

The appeal of basis functions is that the strategy is quite general. You take a physical problem, design a set of features (basis functions) that you think capture important properties of the problem, and then string them together linearly into an approximation. Our own experience, however, is that it is critical to take advantage of problem structure.

Our projects in CASTLE Lab all involve the management of physical resources. Let $a \in \mathcal{A}$ be an attribute vector describing a type of resource (e.g., the location and type of equipment) and let R_{ta} be the number of resources with attribute a at time t. Also let d be a type of decision (move, clean, repair, modify) that acts on a resource with type a, producing a resource with attribute $a' = a^M(a,d)$. Let $R_t = (R_{ta})_{a\in\mathcal{A}}$ be the state of all the resources, and let ρ_t be the state of other parameters (prices, weather, technology) which evolve randomly over time. Our system state variable is $S_t = (R_t, \rho_t)$. Let $\delta_{a'}(a,d) = 1$ if decision d acting on a resource with attribute a produces a resource with attribute a', and let x_{tad} be the number of resources with attribute a that decision $d \in \mathcal{D}$ acts on. The decision vector x_t must satisfy

$$\sum_{d\in\mathcal{D}} x_{tad} = R_{ta}. \tag{WP.9}$$

The post-decision resource vector is given by

$$R^x_{ta'} = \sum_{a\in\mathcal{A}}\sum_{d\in\mathcal{D}} \delta_{a'}(a,d) x_{tad}.$$

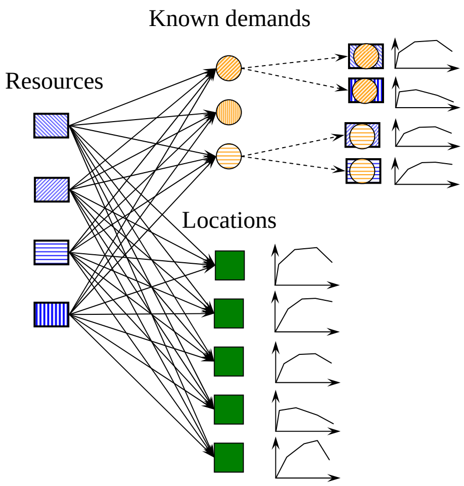


Figure WP.1: Illustration of a linear program for assigning resources to known tasks (with nonlinear value functions after task is completed) and to new locations (with nonlinear value functions capturing value of resources in the future)

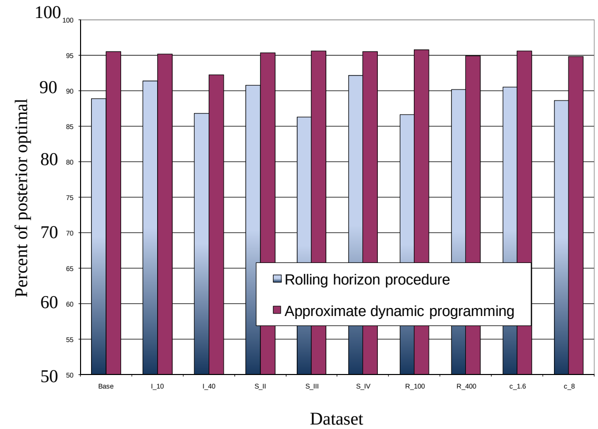


Figure WP.2: Comparison of ADP to a rolling horizon procedure for a stochastic, multicommodity flow network (from [11])

Now let $\hat{R}_{t+1,a}$ be exogenous changes to R^x_{ta} (arrivals, departures, delays, breakdowns). The next pre-decision state is given by

$$R_{t+1,a} = R^x_{ta} + \hat{R}_{t+1,a}.$$

For this problem class, there are some natural value function approximations. We can start by ignoring the exogenous parameter vector ρ_t. Possible value function approximations include linear in the resource vector:

$$\bar{V}_t(R_t) = \sum_{a \in \mathcal{A}} \bar{v}_{ta} R^x_{ta},$$

and separable

$$\bar{V}_t(R_t) = \sum_{a \in \mathcal{A}} \bar{V}_{ta}(R^x_{ta}),$$

where $\bar{V}_{ta}(R^x_{ta})$ may be piecewise linear, or any of a set of concave, nonlinear functions. Figure WP.1 illustrates the decision problem where we can assign a reusable resource to either a set of known tasks (which have the effect of modifying the resource) or we can modify the resource (move it, clean it, repair it, or set up the machine for a type of task). Separable, nonlinear functions (of the post-decision state) capture the value of resources in the future.

"…once the value functions are estimated, it is very easy to solve the 'here and now' problem…"

This strategy has been applied to several transportation applications where the resource is a vehicle (trailer, locomotive, freight car) which can serve a task (move a load of freight, pull a train) or be modified (move the equipment empty to another location, repair the locomotive). ADP compares favorably against a rolling-horizon model, where a decision at time t is based on what we know at time t and forecasts over a specific horizon. These results are then compared against the optimal posterior bound (where we optimize given we know the future), shown in Figure WP.2. Furthermore, once the value functions are estimated, it is very easy to solve the "here and now" problem (at time $t = 0$), which is useful for real-time applications.

Of course, these solutions are all approximate. A powerful strategy is to use Benders' decomposition (WP.4)–(WP.5) (see [3], which offers a convergence proof). In [5, Chapter 11], Benders' decomposition is presented as a type of value function approximation.

It is common to illustrate the process of estimating the value function by using the value of being in a state to update the value function approximation. For resource allocation problems, it is much more effective to use an estimate of the derivative of the maximization problem in equation (WP.6) with respect to the resource variable R_{ta}. Often this is available as a dual variable of equation (WP.9), although we sometimes have to resort to numerical derivatives.

It is important to realize that developing and estimating a value function approximation is a classical statistical modeling exercise (with some twists). It is not necessary to use all the elements of the post-decision state variable. The art of ADP is identifying the variables that are the most important. As with other fields of statistics, econometrics and machine learning, the art is designing the functional approximation, while the science is estimating the best possible function.

Some Applications. Our work in ADP has been motivated by problems in transportation and logistics, including major systems that are now running in production at several companies. Some of these projects include:

- We recently developed a large-scale fleet management system for optimizing the movements of drivers and loads at Schneider National. The model uses ADP to estimate the value of different types of drivers in the future. Drivers are modeled with a 15-dimensional attribute vector, although the linear value function approximation only uses four attributes (which still produces 600,000 parameters to be estimated). The system is in production at Schneider for performing a broad range of policy studies. [8]

- A few years ago we implemented a model for managing freight cars at Norfolk Southern. The model assign cars to known orders, and can reposition cars to different locations in anticipation of unknown demands. The system has been in production for a number of years. [6]

- We have recently completed a decade-long development project, resulting in the successful development and implementation of an optimization model using ADP for Norfolk Southern Railroad (paper in preparation).

- We recently completed a planning system for high-value spare parts for Embraer. The system uses ADP to determine how many spares of over 400 types of parts should be allocated over 19 service centers, subject to budget and service constraints [9].

In addition to these production projects, ADP has been used in numerous senior theses and projects at Princeton University. However, it has been the ability of ADP to handle the complexities of real-world problems that seems to set it apart from other methods.

Closing Remarks

So, ADP is truly a melting pot of methods. It involves statistics/machine learning (estimating the value function) and simulation, all within a dynamic programming framework that allows you to use your favorite optimization algorithm for solving decision problems at each point in time. This strategy has allowed us to solve some extremely large problems that arise in freight transportation, as well as applications in health and finance.

ADP is more than just a technique for solving stochastic problems. We often use it as a decomposition method for very large-scale deterministic problems, including some hard integer programming problems. We have found that commercial solvers such as CPLEX® may struggle with long horizons, but can often solve very large problems as long as the horizon is fairly short. Thus, problems that might otherwise be solved using a heuristic can be solved optimally at a point in time, with solutions that are quite good over time.

References

[1] D. P. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA, 1996.

[2] J. Birge and F. Louveaux, *Introduction to Stochastic Programming*, Springer-Verlag, New York, NY, 1997.

[3] J. Higle and S. Sen, *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming*, Kluwer Academic Publishers, Norwell, MA, 1996.

[4] K. Judd, *Numerical Methods in Economics*, MIT Press, Cambridge, MA, 1998.

[5] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, John Wiley & Sons, New York, NY, 2007.

[6] W. B. Powell and H. Topaloglu, Fleet Management, in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pages 185–216, MPS-SIAM Series on Optimization, Philadelphia, PA., 2005.

[7] M. L. Puterman, *Markov Decision Processes*, John Wiley & Sons, New York, NY, 1994.

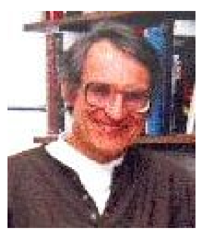
[8] H. P. Simao, J. Day, A. P. George, T. Gifford, J. Nienow, and W. B. Powell, An Approximate Dynamic Programming Algorithm for Large-Scale Fleet Management: A Case Application, *Transportation Science* (to appear).

[9] H. P. Simao and W. B. Powell, Approximate Dynamic Programming for Management of High Value Spare Parts, *Journal of Manufacturing Technology Management* 20:9 (to appear 2009).

[10] R. Sutton and A. Barto, *Reinforcement Learning*, MIT Press, Cambridge, MA, 1998.

[11] H. Topaloglu and W. B. Powell, Dynamic Programming Approximations for Stochastic, Time-Staged Integer Multicommodity Flow Problems, *INFORMS Journal on Computing* 18:1 (2006), 31–42.

[12] B. Van Roy, D. P. Bertsekas, Y. Lee, and J. N. Tsitsiklis, A Neuro-Dynamic Programming Approach to Retailer Inventory Management, in *Proceedings of the 36th IEEE Conference on Decision & Control*, Vol. 4, 1997, 4052–4057.



Computational Probability: Yesterday, Today, and Tomorrow

Winfried K. Grassmann ▹
University of Saskatchewan
(▹BibTEX entry)

Winfried Grassmann is Professor Emeritus in the Department of Computer Science at the University of Saskatchewan, working on many aspects of stochastic models, simulation, and performance evaluation. His 2008 articles are: “Three Canadian Contributions to Stochastic Modeling” [*INFOR*], “Warm-Up Periods in Simulation can be Detrimental” [*Probability in the Engineering and Informational Sciences*], and “Multiple Eigenvalues in Spectral Analysis for Solving QBD Processes” [*Methodology and Computing in Applied Probability*]. His most recent book is *Computational Probability* [Kluwer, 2000]. He is on the editorial boards of the *INFORMS Journal on Computing* and *INFOR* (*Information Systems and Operational Research*). Winfried is founder and president of the Saskatoon Section of the Canadian Operational Research Society.

In the 1960s, Ulf Grenander, a young professor of applied mathematics at Stockholm University, wanted to equip his students with the ability to deal with realistic problems rather than with the abstract problems described in the applied mathematics textbooks. To accomplish this, he went to the business community and searched for problems he could use in class to teach his students how to formulate and solve practical problems. However, the problems he found were too complex to be solved by analytical means, and numerical methods were needed. For solving such problems, the computer proved extremely helpful. Many of the problems he found involved stochastic processes and statistics. In this way, he became the creator of computational probability. Interestingly, he used Monte Carlo simulation as one of the major tools, and though there are valid reasons to consider simulation as part of computational probability, we will not discuss simulation in this review.

In 1966, Professor Grenander moved to Brown University, where he taught a class in Computational Probability. His students really liked his approach, and in particular, they liked experimenting and exploring alternative solutions. This was made possible because of on-line terminals which were just introduced in these years. Grenander and his group also used the computer to do computational mathematics, that is, they used the computer for exploring theoretical problems. His course gave rise to the first textbook in computational probability. It was co-authored by Freiberger, and its title is "A Short Course in Computational Probability and Statistics" [6]. This book appeared in 1971.

Another name closely related to computational probability is Marcel Neuts. He used Markov renewal processes to investigate queueing problems. These processes led to embedded Markov chains with an infinite number of states. To solve these systems, he originally used transform methods, but transforms are difficult to invert. In another research project, he used the computer to calculate transient probabilities for discrete-time queues. He was thrilled because the computer allowed him to obtain a wealth of information, information not obtainable by theoretical means. He therefore became an enthusiastic computer user, and he started to search for methods that would enable him to solve numerically the queueing problems he analysed earlier with transform methods. To this end, he formulated matrix equations which he could solve recursively. In this fashion, he started what later became known as the theory of matrix analytic methods. The two main paradigms in this area are GI/M/1 [16] and the M/G/1 [15]. Essentially, these methods can be used to find equilibrium probabilities of Markov chains with infinite-dimensional block-structured Hessenberg matrices with repeating rows. The M/G/1 queue results in an embedded Markov chain with an upper Hessenberg matrix, and the M/G/1 paradigm generalizes this notion by replacing the scalars of the Hessenberg form by matrices, which is to say that the M/G/1 paradigm deals with block-structured upper Hessenberg transition matrices. The GI/M/1 paradigm similarly generalizes the lower Hessenberg matrices arising in connection with the GI/M/1 queue. To find equilibrium solutions for such problems, Neuts created and theoretically justified a number of algorithms. Neuts also made important contributions to the GI/M/1 paradigm which he popularized. His work was influenced by Wallace [22], who found that the equilibrium vector in GI/M/1 paradigm problems is matrix geometric.

"[Professor Grenander's] course gave rise to the first textbook in computational probability."

Wallace is also know for his recursive queue analyser or RQA [23], a program he wrote in or before 1966. This program deals with what we call Markovian event systems. Like in discrete event systems, Markovian event systems have a number of state variables, and these variables can only be changed by events. However, in contrast to discrete event systems, events need not be scheduled. Instead, they just occur at a certain rate, a rate that depends only on the present state of the system. This makes these systems Markovian. RQA takes the event-based description of the system to generate the transition matrix of a continuous-time Markov chain. The program then determines the equilibrium probabilities by using the power method. A later addition to the program also allowed to find transient solutions by using Runge-Kutta.

In the late 1960s, influenced by Schassberger [20], I used the power method for finding equilibrium solutions for large Markov chains, a method that worked very well. I then discovered that Feller [5] suggested to randomize discrete-time Markov chains to find transient solutions of continuous-time Markov chains. Since randomizing a discrete-time Markov chain is essentially equivalent to randomizing the power method, I figured this method should work quite well for calculating the transient probabilities of continuous-time Markov chains. This turned out to be true. In 1971, I submitted a program implementing this method to the Share library of IBM [10]. A discussion of the method was only published in 1977 [7]. I soon discovered that for Markovian event systems, creating the transition matrix is quite cumbersome, and I therefore used a matrix generator as a front end. This generator had as input events, their rates and their effect on the state variables, and it was similar to the one used by Wallace [23] earlier. My work influenced Don Gross, who wrote, together with Doug Miller, a famous paper on randomization [12]. Because of this paper, the method spread, and it was used successfully by many researchers and practitioners. Many authors prefer the term "uniformization" to the term "randomization". Another word for this method is "Jensen's method", a term I prefer because it was Jensen who first suggested this method [14]. However, according to Gross and Miller [12], I was the first person who used this method as a numerical tool for finding transient probabilities for continuous-time Markov chains. Jensen's method is now the standard method to do this. We should note, however, that Jensen's method was used before extensively for theoretical derivations. In fact, it was used in 1967 by Schassberger [20, eq. 1.2.8], but he did not recognize its power for numerical purposes.

In principle, one could use standard numerical methods to analyse large Markov chains. However, the fact that probabilities are positive often allows one to modify algorithms such that all subtractions are avoided. This is advantageous because subtractions often magnify rounding errors committed at earlier stages of an algorithm. The culprit is subtractive cancellation, that is, the loss of significant digits arising from the subtraction of floating point numbers of almost equal magnitude. Avoiding subtractions can make a dramatic difference. To demonstrate this, consider Jensen's method again. Mathematically, Jensen's method can be interpreted as a Taylor expansion of the matrix exponential, except for a shift by an amount q, where q can be chosen such that subtractions are eliminated. Mathematically, shifting by q means that one writes $\exp(At)$ as:

$$e^{At} = e^{Pqt}e^{-qt} = e^{-qt}\sum_{n=0}^{\infty}\frac{(qt)^n}{n!}P^n,$$

where $P = A/q + I$. Now, Taylor expansions involving matrices are known to be numerically unstable, and one would thus conjecture that Jensen's method is unstable as well. However, if q is chosen such that P has only positive elements, no subtractions occur, and the algorithm can be proved to be extremely stable [8]. On the other hand, if q is chosen such that P contains negative elements, the algorithm becomes unstable. Since all numerical analysts learn that the Taylor expansion of the matrix exponential is numerically unstable, it comes to no surprise that when I first tried to publish Jensen's method, the paper was rejected, and it was suggested that the Runge-Kutta method would be more appropriate. However, according to Ingolfsson [13], Runge-Kutta is slower than Jensen's method.

Another case where the avoidance of subtractions can make a huge difference is in the solution of equations. Every numerical analyst will tell you that solving large systems of linear equations by Gaussian elimination will be numerically unstable. However, in the GTH algorithm, so named because it was first described by Grassmann, Taksar and Heyman [11], subtractions are avoided when solving the equilibrium equations of Markov chains. Because of the preconceived opinions of some numerical analysts, some referees were highly skeptical regarding this method. I can attest to this from my experience as an associate editor.

Deterministic methods, that is, methods that do not use Monte Carlo simulation, have one major disadvantage: if there are several state variables, one has to calculate their joint distribution. Unfortunately, the number of probabilities needed to determine this joint distribution increases exponentially with the number of state variables. This is known as the curse of dimensionality. In contrast, the computational complexity of discrete event simulation increases linearly with the number of state variables, and it follows that from a certain point onward, discrete event simulation leads to shorter CPU times for finding transient and steady-state solutions. Hence, in the area of Markovian event systems, there are definitely problems where simulation has a lower complexity than deterministic methods, both in time and space. This contradicts the frequently expressed opinion that simulation should be avoided when deterministic methods are available.

Though discrete event simulation can be faster than deterministic methods, we should not underestimate the power of deterministic methods in the context of Markovian event systems. In fact, problems with a million states or more are solved routinely by deterministic methods, and this is more than enough for many problems of practical importance.

"The question is whether or not there is a convenient representation of the system that is more compact than the transition matrix."

Of course, finding transient and equilibrium solutions for Markov chains with a million states is challenging, and much research is being devoted to find the best methods in this context. This task is somewhat simplified by the sparsity of the matrices involved. The reason is that the number of events increases at a much lower rate than the number of states. For instance, even if the number of states is in the millions, the number of events is typically below 100. Of course, the number of entries in a row is limited by the number of events that can occur in the state in question, and this implies that the transition matrices of Markovian event systems are sparse. Still, solving such huge systems is challenging, and much research has been devoted to finding faster solution methods in this context. This research was spearheaded by William (Billy) Stewart, who organized a number of conferences on this topic, and who wrote the a textbook on this subject [21].

The methods used for finding equilibrium solutions for huge Markov chains are mostly iterative because iterative methods preserve sparsity. Direct methods, such as the GTH method, will lead to fill, and this increases both the memory required and the time complexity. Direct methods are only used for matrices with a narrow band, say 100 or less. The iterative methods suggested include the power method, Gauss-Seidel iterations, projection techniques and decomposition methods. For transient solutions, the method of Jensen is widely used, which is no surprise because its close relation to the power method. For details of all these methods, see [21].

Note that the event-based representation of a Markovian event system is much more compact than the transition matrix of the corresponding Markov chain. The question is whether or not there is a convenient representation of the system that is more compact than the transition matrix. One compact representation uses Kroenecker products and stochastic automata (see [17, 21]), but these representations are not efficient when used in connection with most iterative methods.

Computational probability is used extensively, though not under this name, by researchers in computer performance analysis. We mention here, in particular, the work of Buzen [2], who gave the first algorithm for efficiently calculating steady-state solution for large closed queueing networks with exponential service times. We must also mention generalized stochastic Petri-nets or GSPNs [1,3] and stochastic activity nets or SANs [19]. These are graphical tools for formulating continuous-time Markov chains. In these graph, there are *places*, which correspond to the state variables in event-driven systems. Each place can contain a number of *tokens*, which in this sense indicate the value of the state variable representing the place. To move tokens between places, one has *transitions*, which in this sense correspond to the events. There are two types of transitions: immediate and timed transitions. In an immediate transitions, the tokens are moved as soon as certain conditions are met, in which case the transition is said to *fire*. In timed transitions, the time between enabling the transition and the state change is an exponentially distributed random variable. It is easily verified that GSPNs and SANs are Markovian, and transient and steady-state solutions can be found by the methods available for solving large Markov chains. We should mention, however, that in many cases, simulation rather than deterministic methods are used. The reason is that for large systems, simulation is faster.

There are many additional applications of computational probability which we cannot describe here because of space limitations. We should mention, however, that many reliability problems benefit from the theory of computational probability. Also, problems similar to the ones discussed in this paper arise in the context of Markov decision problems [18]. For these and other applications, see [9]. An interesting approach has been used by Drew et al. [4]. These authors use Maple® to manipulate distributions, and they call this computational probability. Some persons in the area may not agree with this, but it is an interesting approach.

You may not know it, but whenever you use Google, you benefit from the research in Markov chains. In fact, the so-called PageRank® algorithm, pioneered by Page and Brin, is based on a discrete-time Markov chain. They created a process that follows the activities of a user choosing links in web-pages at random, and this process is Markovian. These Markov chains have an incredibly large number of states, and finding the required equilibrium probabilities is no trivial matter. Indeed, it takes a few days to obtain the results, and both efficient calculations of the equilibrium probabilities, and efficient updating of these probabilities are an area of intense research.

"…whenever you use Google, you benefit from the research in Markov chains."

Nobody knows whether or not there will be another killer application like the PageRank algorithm. However, if the work of Page and Brin is any indication, this new application, if it exists, may very well involve sophisticated mathematical methods. In conclusion: we all should look for good and realistic applications, but we should not forget our mathematical tools. Who knows, maybe one of our readers will someday discover another killer application and make millions.

References

[1] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. John Wiley & Sons, New York, NY, 1995.

[2] J. P. Buzen. Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM*, COM-34(8):527–531, 1973.

[3] G. Ciardo. SMART, a stochastic model checking analyser for reliability and timing. http://www.cs.ucr.edu/~ciardo/SMART.

[4] J. H. Drew, D. L. Evans, A. G. Glen, and L. M. Leemis. *Computational Probability. Algorithms and Applications in the Mathematical Sciences*. International Series in Operations Research and Management Science. Springer-Verlag, New York, NY, 2008.

[5] W. Feller. *An Introduction to Probability Theory and its Applications*, volume 2. John Wiley & Sons, New York, NY, second edition, 1968.

[6] W. Freiberger and U. Grenander. *A Short Course in Computational Probability and Statistics*, volume 6 of *Applied Mathematical Sciences*. Springer Verlag, New York, NY, 1971.

[7] W. K. Grassmann. Transient solutions in Markovian queueing systems. *Computers & Operations Research*, 4(1-D):47–56, 1977.

[8] W. K. Grassmann. Rounding errors in certain algorithms involving Markov chains. *ACM Transaction on Mathematical Software*, 19(4):496–508, 1993.

[9] W. K. Grassmann, editor. *Computational Probability*. International Series in Operations Research and Management Science. Kluwer Academic Publishers, Norwell, MA, 2000.

[10] W. K. Grassmann and T. K. Ngai. Program description for transient solutions in continuous Markov chains. Share Program Library Agency, Program 360D-15.0.005, Research Triangle Park, NC, 1971.

[11] W. K. Grassmann, M. Taksar, and D. P. Heyman. Regenerative analysis and steady state distributions for Markov chains. *Operations Research*, 33:1107–1117, 1993.

[12] D. Gross and D. R. Miller. The randomization technique as a modeling tool and solution procedure for transient Markov processes. *Operations Research*, 32(2):343–361, 1984.

[13] A. E. Ingolfsson, E. Akhmetshina, S. Budge, Y. Li, and X. Wu. A survey and experimental comparison of service level approximation methods for non-stationary M/M/s queueing systems. *INFORMS Journal of Computing*, 19:201–214, 2007.

[14] A. Jensen. Markoff chains as an aid in the study of Markoff processes. *Skandinavisk aktuarietidskrift*, 36:87–91, 1953.

[15] M. F. Neuts. *Matrix-Geometric Solutions in Stochastic Models*. The Johns Hopkins University Press, Baltimore, MD, 1981.

[16] M. F. Neuts. *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Marcel Dekker, New York, NY, 1989.

[17] B. Plateau and W. J.Stewart. Stochastic automata networks. In W. K. Grassmann, editor, *Computational Probability*, International Series in Operations Research and Management Science, chapter 3, pages 113–150 Kluwer Academic Publishers, Norwell, MA, 2000.

[18] M. Putterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New York, NY, 1994.

[19] W. H. Sanders et al. Möbius, model-based environment for validation of system reliability, availability, security and performance. http://www.mobius.uiuc.edu/papers.html.

[20] R. Schassberger. Ein Warteschlangensystem mit zwei Warteschlangen. *Computing*, 3:110–124, 1968.

[21] W. Stewart. *An Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, Princeton, NJ, 1994.

[22] V. Wallace. *The Solution of quasi birth and death processes arising from multiple access computer systems*. PhD thesis, University of Michigan, Ann Arbor, MI, 1969.

[23] V. L. Wallace. RQA-1, The recursive queue analyser. Technical Report 2, Systems Engineering Laboratory, University of Michigan, Ann Arbor, MI, 1966.

Cyberinfrastructure and Optimization

Robert Fourer ▷
Northwestern University
(▷BibTEX entry)

Bob received his Ph.D. in Operations Research from Stanford University in 1979. He then joined the Department of Industrial Engineering and Management Sciences at Northwestern University, where he remains as Professor. Bob is especially well known for his development of ***A Mathematical Programming Language*** (AMPL) with David Gay and Brian Kernighan, for which they received the *ORSA CSTS Prize* for excellence in research at the interface of operations research and computer science. Bob also received a *Guggenheim Memorial Foundation Fellowship* (2002); the MPS *Beale-Orchard-Hays Prize* (with E.D. Dolan, J.J. Moré and T.S. Munson) (2003); the IIE *Medallion Award* (2004); and the *INFORMS Fellow Award* (2004).

In 2002 the U.S. National Science Foundation created a Blue-Ribbon Advisory Panel on *Cyberinfrastructure*, which submitted in January of 2003 a report entitled "Revolutionizing Science and Engineering Through Cyberinfrastructure" [2]. Subsequently, the NSF created an Office of Cyberinfrastructure (OCI) independent of its directorates in such traditional areas as biology, computer science, geosciences, physical science, and engineering. In the following three years the NSF sponsored workshops leading to nearly 30 reports (www.nsf.gov/od/oci/reports.jsp) on the role of cyberinfrastructure in specific areas of research.

OCI's statements of its mission provide a taste of what the term cyberinfrastructure is intended to encompass (taken from http://www.nsf.gov/od/oci/about.jsp):

> The Office of Cyberinfrastructure coordinates and supports the acquisition, development and provision of state-of-the-art cyberinfrastructure resources, tools and services essential to the conduct of 21st century science and engineering research and education.
>
> OCI supports cyberinfrastructure resources, tools and related services such as supercomputers, high-capacity mass-storage systems, system software suites and programming environments, scalable interactive visualization tools, productivity software libraries and tools, large-scale data repositories and digitized scientific data management systems, networks of various reach and granularity and an array of software tools and services that hide the complexities and heterogeneity of contemporary cyberinfrastructure while seeking to provide ubiquitous access and enhanced usability.
>
> OCI supports the preparation and training of current and future generations of researchers and educators to use cyberinfrastructure to further their research and education goals, while also supporting the scientific and engineering professionals who create and maintain these IT-based resources and systems and who provide essential customer services to the national science and engineering user community.

My purpose here is to describe some projects that fall into the intersection of cyberinfrastructure with the study and practice of large-scale optimization. Parts of this survey have been adapted from sections of [6–8].

Naturally many of these projects have to do with minimizing costs or maximizing profits in operations research applications. But activities as diverse as product design, manufacturing, and supply-chain management all seek to minimize costs, or a surrogate for costs. Many topics in the sciences, such as the folding of proteins, are studied as the minimization of energy or forces. Problems of these and many other kinds are addressed by a large optimization community with roots in computer science, operations research, management science, and numerous engineering and scientific disciplines.

"OR tools bridge the gaps between information, knowledge, and decision-making."

Cyberinfrastructure and Operations Research

Everyone is familiar with infrastructures: road systems, rail networks, power grids. An infrastructure does not produce goods or services itself; rather, it makes a wide range of productive activities possible. The interstate highway infrastructure does not itself carry out supply-chain management, for example, but it permits the development of supply-chain management systems that would not be possible otherwise. Indeed, it paves the way for phenomena that were not foreseen when it was built, such as crossdocks and suburban sprawl. The effectiveness of infrastructures depends critically on standards (track gauges and standard time for railroads, bridge heights for highways, voltages for power grids) and on accessibility to a broad base of users.

Among the major infrastructures of modern life, cyberinfrastructures constructed from computers, data networks, software, and communications standards are among the newest and most elaborate instances. The Internet and the Web are the best known examples. Like other infrastructures, they facilitate myriad applications — the Web's use for unexpected purposes is already legendary — and they depend critically on software standards such as IP, HTTP, and HTML.

Operations Research also has characteristics of an infrastructure, in the sense that it is a collection of theory, algorithms, and software that underpins and facilitates productive activity. In essence, OR tools bridge the gaps between information, knowledge, and decision-making, with major impacts in design, manufacturing, services, and supply-chain management. Like other infrastructures, OR tools serve many purposes that were not envisioned by their creators.

To achieve their potential, OR tools require high-quality data in the form of current, comprehensive information and knowledge. OR tools also require computational platforms that allow large, difficult mathematical problems to be formulated and solved, often in real time. Cyberinfrastructure can help with both these needs. In supply-chains, for example, cyberinfrastructures can collect data from sensors and suppliers and transmit orders, while OR tools organize the data and make decisions. In engineering design, cyberinfrastructures can provide access to diverse and distributed databases and to powerful computing platforms, while OR facilitates data mining, exploration of larger scenario spaces, and handling of uncertainty in the design process.

COIN-OR

The Computational Infrastructure for Operations Research (COIN-OR) project[1] is an initiative to spur the development of open-source software for the operations research community [14]. COIN-OR acts as a cyberinfrastructure in two ways. It makes available uniform tools for developing, managing, and documenting open-source projects. Also many of its projects are tools — or collections of tools — for use in OR applications.

Why open source? As the Open Source Initiative (http://www.opensource.org) explains, when people can read, redistribute, and modify the source code, software evolves. People improve it, adapt it, and fix bugs. The results of community-based efforts to develop software under open-source licenses have produced high-quality, high-performance code — including code on which much of the Internet is run.

Why open source for OR? COIN-OR envisions a scenario such as the following. You read about an optimization algorithm in the literature and you get an idea on how to improve it. Today, testing your new idea typically requires re-implementing (and re-debugging and re-testing) the original algorithm. Often, clever implementation details are not published. It can be difficult to replicate reported performance. Now imagine that the original algorithm is publicly available in a community repository. Weeks of re-implementing are no longer required. You simply check out a copy of it for yourself and modify it. The result is software reuse, and a tremendous productivity gain.

COIN-OR's most ambitious goal is to create for mathematical software what the open literature is for mathematical theory. It is building an open-source community for operations research software in order to speed development and deployment of models, algorithms, and cutting-edge computational research, as well as to provide a forum for peer review of software similar to that provided by archival journals for theoretical research.

[1] Editor note: See p. 3 for the COIN-OR report and links to its web site.

NEOS

Over the past decade a large group of collaborators have built NEOS, a *Network-Enabled Optimization System*, with the goal of making optimization an Internet resource [4]. NEOS has been developed in two parts:

- A NEOS Guide (http://www.mcs.anl.gov/otc/Guide) that collects tutorial material, case studies, test problems, and frequently asked questions for a range of optimization problem types.
- A NEOS Server (http://neos.mcs.anl.gov) that provides free Internet access to over 60 algorithmic packages ("solvers") that can be applied to optimization problems of diverse kinds.

NEOS has become the preeminent online resource in its field. The NEOS Server in particular has revolutionized optimization research, teaching, and applications, by providing immediate access to far more solvers than optimization users could hope to install locally. Numerous open-source solvers have been placed on NEOS, many of them based on recent research in such areas as global optimization, semidefinite programming, and nonlinear optimization with integer variables. But even many commercial solver developers have made their products available for free use on NEOS to encourage potential customers to try them out.

For the optimization community, the NEOS Server provides the characteristics generally associated with a cyberinfrastructure: facilitating applications rather than directly performing them; enabling more applications than were originally imagined; providing open access to Internet-based resources; and encouraging standards for information interchange. NEOS is envisioned as becoming not simply a stand-alone tool for the optimization community, moreover, but a resource that is interoperable with other analytic activities in business, science, and engineering. Towards this end, the NEOS Server software has recently been rewritten to use more standard conventions for data transfer (XML) and remote procedure invocations (XML/RPC) [5]. As a result the Server is fully callable from programs running anywhere on the Internet.

Business users of the NEOS Server do eventually install their preferred solvers locally, to gain greater reliability and security. The creation of a practical model for selling optimization through an arrangement like NEOS is a challenge that remains open.

Optimization Services

To create a sort of "next generation" of the NEOS Server, a project has been undertaken to design an innovative distributed optimization environment in which modeling languages, servers, registries, agents, interfaces, analyzers, solvers, and simulation engines can be implemented as services and utilities under a unified framework. This work, called Optimization Services or OS [6], defines standards for all activities necessary to support decentralized optimization on the Internet. A reference implementation is freely available as an open-source project under COIN-OR.

The OS framework is motivated by a conviction that, to be a practical tool, optimization increasingly needs to become integrated into modern corporate information technology (IT) infrastructures. The OR community has focused on standalone tools like modeling languages and solvers designed to work on a single machine, while the IT community is has been moving to tools like Extensible Markup Language (XML), Service Oriented Architecture (SOA), and Web Services that facilitate distributed computing. The OR community could much more readily achieve its objectives if optimization tools were built into technologies that the IT community is already using.

XML, SOA, and Web Services have facilitated the growing prevalence of *software as a service*: that is, software residing on a server that is accessed by numerous client machines over a network, as opposed to software residing in multiple copies on users' machines. Current examples of software as a service include customer relationship management (see http://www.salesforce.com), tax preparation, Gmail, and Google Calendar. Indeed, all of the major players in software have been promising software as a service; the trend is away from the fat client loaded with heavyweight applications, and towards distributed computing.

The goal of OS is thus to determine how optimization can be conceived as a modern software service. This is easier said than done, because optimization software embodies a number of difficulties that are inherent to its nature as a tool for numerical computing as well as symbolic modeling. For one thing, optimization lacks standards for communication in almost every respect:

- There are numerous optimization modeling languages, each with its own representations for models and data. In consequence there are only the most primitive standards for representing model instances, and even those are mainly confined to linear models.
- There are numerous solvers, each with its own application program interface (API), but there is no standard solver API.
- There are virtually no standards for representing solvers' algorithmic options and results.
- There is no standard protocol for registration and discovery of solver services over a network.
- Optimization projects use such a variety of operating systems, processor architectures, and compilers that developers of optimization applications have great difficulty supporting all platforms that are in demand.

Overall, optimization services exhibit a greater variety and complexity of information to be moved around and a much greater range of behavior to be dealt with than do typical business applications. To further complicate matters, solvers are categorized by mathematical problem types that do not readily correspond to the model types familiar to customers. Thus building an OS framework is more of a challenge than simply copying XML, SOA, and Web Services ideas from existing software over to optimization packages.

"…an infrastructure for large-scale optimization on advanced computing platforms will require a sort of supercomputing on demand…"

The results of the OS project comprise a *framework* for distributed optimization, a set of standards (or protocols) for

- representation of optimization instances, results, and solver options;
- communication between clients and solvers; and
- registration and discovery of optimization-related services using the concept of Web Services.

To provide these standards, OS incorporates general and robust formats for representing optimization model instances in text files or in memory, a common interface to these formats including `get( )`, `set( )`, and `calculate( )` methods, and standard registry and discovery protocols. OS also includes protocols that facilitate communication between modeling clients and solver servers on any combination of platforms.

This project's ultimate goal is to *make optimization as easy as hooking up to the network*. The vision is for all optimization system components to be implemented as services under the OS framework, and for customers to use these computational services much like utilities, with specialized knowledge of optimization algorithms, problem types, and solver options being potentially valuable but not required. The OS framework will in turn be built upon standards that are independent of programming language, operating system, and hardware, and that are open and readily available for use by the optimization community.

Intelligent optimization systems

Optimization services have largely been conceived as providing algorithmic solvers to people who want optimal (or at least very good) solutions to optimization problem instances. Underlying this view, however, has been a confidence that many owners of problems are knowledgeable as to which solvers are appropriate. Yet as previously noted, solvers are applicable to specific mathematical problems types such as linear, integer, smooth or nonsmooth nonlinear, logical, and many specializations. These do not readily correspond to the concerns of modelers who are thinking in terms of production, distribution, scheduling, design, and other model types applied in particular areas of science, engineering, and commerce.

It is thus worth considering what might be gained by taking a broader view. One can imagine an optimization cyberinfrastructure that incorporates software to automatically aid in the selection of solvers. Features of particular value include identifying convexity, both generally in objective functions and specifically in the case of constraints that can be viewed as quadratic cones; converting common nondifferentiable and discontinuous functions to forms that diverse solvers can handle; and making constraints involving natural combinatorial and logical operators accessible to both numerical and logic-based solvers. The DrAMPL project [9] has taken some steps along these lines, including the matching of deduced problem characteristics against a database of solver features.

Going further, one can envision a optimization services framework that incorporates intelligent aids for modeling, tuning of solver options, and analysis of results. Software embodying aids of these kinds was in existence as far back as the late 1970s, when ANALYZE [12] was developed at the U.S. Federal Energy Administration; Greenberg [11] provides an overview and bibliography of developments through the mid-1990s, and work in this area has continued, as evidenced for example by MProbe [3] and by the mechanisms for problem analysis and transformation found increasingly in implementations of optimization modeling languages. Many remain many ways in which the power of such systems could be further expanded, however, and it will be a significant challenge even to adapt existing systems to function as independent services that can be treated as part of the infrastructure of optimization.

Advanced computing

Software as a service implies the existence of hardware platforms to act as servers. Current optimization service frameworks, like NEOS and OS, rely on ordinary computers: PCs running Windows or Linux, and various Unix workstations. But there also exists the potential to enhance the practice of optimization by bringing advanced computing — a concept widely associated with cyberinfrastructure — to the optimization community.

By "advanced" I mean here any of several approaches that use multiple processors to accomplish what cannot be done effectively by individual computers, including

- *high-performance computing,* using large numbers of specialized processors and specialized interconnections;
- *distributed computing,* using standard computers working together through Internet connections; and
- *high-throughput computing,* using the computational resources of otherwise idle networked computers.

A great variety of optimization problems have features that permit advanced computing to be used to advantage. For example, the metaNEOS project of 1997–2001 applied advanced computing approaches in solving all of the following:

- the 10^{10}-variable deterministic equivalent of a 107-scenario stochastic program on a computational grid of about 800 workstations, in about 32 hours of wall-clock time [13];
- a previously intractable quadratic assignment problem using an average of 650 worker machines over a one-week period, providing the equivalent of almost 7 years of computation on a single workstation [1];
- a mixed-integer nonlinear programming problem with parallel efficiency of up to 80% on 600 million search-tree nodes [10].

Yet, essentially no applications in optimization have benefited from this work. Despite decades of development on advanced platforms of these kinds, experience in their use remains exceedingly rare among people trained in optimization. For most members of the optimization community, whose focus is modeling and solving rather than computing, it is a daunting (and disheartening) challenge to assemble and configure the hardware and software resources necessary to apply or even experiment with such advanced computational approaches.

"...supercomputing on demand ... is an area where the optimization and computing communities need to agree on some substantive and original cyberinfrastructure research."

The software services concept offers a clear possibility for a remedy to this situation. An advanced computing platform and the software tailored to it could be set up to act as an optimization server. Users anywhere on the Internet could send their problems to be solved, in much the same forms as are sent to ordinary solvers today — requiring at most a limited knowledge of advanced computing technology. The developers and maintainers of the optimization methods implemented on such servers would need to understand the technology in detail, but they would see their efforts benefit a great many more applications than at present.

High-performance computers are already accessed by their users via the Internet, to be sure. But for reasons of scarcity, security, or just plain custom, specialized multiprocessor computers and large multiprocessor networks have been available only by prearrangement of availability of the software and, in some cases, availability of hardware time. Optimization users expect to be able to request the use of algorithms when they're needed, and for unpredictable amounts of time; after all, that is what's available from NEOS. Such needs are inherent in the nature of large-scale optimization, which involves the use of algorithms that work well in practice but have no theoretical performance guarantees, and the repeated invocation of algorithms on varied problems generated under the control of independently implemented iterative schemes.

In sum, an infrastructure for large-scale optimization on advanced computing platforms will require a sort of *supercomputing on demand* that does not seem to have been so necessary for other applications. This is an area where the optimization and computing communities need to agree on some substantive and original cyberinfrastructure research.

Prospects for cyberinfrastructure in optimization

I introduced this article's topic through a description of the National Science Foundation's Office of Cyberinfrastructure, whose mandate is to fund basic research. Do innovations in cyberinfrastructure for optimization have a potential to be treated as research contributions? Grant panelists and journal referees have at times viewed the work described herein as straightforward applications of ideas already pioneered more broadly in the context of Information Technology. To further cyberinfrastructure as a research topic in optimization, proponents of this area of investigation may have to better educate the IT and OR communities in the aspects of optimization that truly pose challenges for cyberinfrastructure projects; some of these aspects have been noted in this article.

Perhaps the creation of cyberinfrastructures for optimization will evolve to be as much a commercial as a scientific activity, however. The last decade has seen an increasing number of companies that provide or embed optimization in their products and that could benefit from some of the ideas I have described. Bigger players such as SAS, Microsoft, and IBM are greatly expanding the role of optimization in their offerings and have the resources to establish the ideas and standards of the Optimization Services project among a broad range of clients. Indeed many of the concepts described in this article have lately been brought together under the umbrella of "cloud computing," which is a predominantly commercial phenomenon. The intersection of cyberinfrastructure and optimization would thus seem to have considerable potential for an exciting and influential future.

References

[1] K. Anstreicher, N. Brixius, J-P. Goux, and J. Linderoth. Solving large quadratic assignment problems on computational grids. *Mathematical Programming*, 91(3):563–588, 2002.

[2] D. E. Atkins, K. K. Droegemeier, S. I. Feldman, H. Garcia-Molina, M. L. Klein, D. G. Messerschmitt, P. Messina, J. P. Ostriker, and M. H. Wright. Revolutionizing science and engineering through cyberinfrastructure: Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure. Report cise051203, National Science Foundation, 2004. Available at www.nsf.gov/od/oci/reports/toc.jsp.

[3] J. W. Chinneck. Analyzing mathematical programs using MProbe. *Annals of Operations Research*, 104:33–48, 2001.

[4] E. D. Dolan, R. Fourer, J. J. Moré, and T. S. Munson. Optimization on the NEOS server. *SIAM News*, 35(6):4, 8–9, 2002.

[5] E. D. Dolan, R. Fourer, J-P. Goux, T. S. Munson, and J. Sarich. Kestrel: An interface from optimization modeling systems to the NEOS server. *INFORMS Journal on Computing*, 20(4):525–538, 2008.

[6] R. Fourer, J. Ma, and K. Martin. Optimization services: A framework for distributed optimization. Technical report, Optimization Services Project, 2008. Available at www.4er.org/CI/Optimization-Services.pdf.

[7] R. Fourer, J. J. Moré, T. Munson, and S. Leyffer. Extending a cyberinfrastructure to bring high-performance computing and advanced web services to the optimization community. Proposal to the National Science Foundation, 2006. Available at www.4er.org/CI/NSF-Proposal-2006.pdf.

[8] R. Fourer, J. J. Moré, K. Ramani, and S. J. Wright. An operations cyberinfrastructure: Using cyberinfrastructure and operations research to improve productivity in the enterprise. Report on a workshop sponsored by the National Science Foundation, 2004. Available at www.optimization-online.org/OCI/OCI.pdf.

[9] R. Fourer and D. Orban. DrAmpl: A meta solver for optimization. Northwestern University and École Polytechnique de Montréal, 2008. Available at www.4er.org/CI/DrAmpl.pdf.

[10] J-P. Goux and S. Leyffer. Solving large MINLPs on computational grids. *Optimization and Engineering*, 3(3):327–346, 2002.

[11] H. J. Greenberg. A bibliography for the development of an intelligent mathematical programming system. *Annals of Operations Research*, 65:55–90, 1996.

[12] H. J. Greenberg. *A Computer-Assisted Analysis System for Mathematical Programming Models and Solutions: A User's Guide for ANALYZE*, volume 1 of *Operations Research/Computer Science Interfaces Series*. Kluwer Academic Publishers, 1993.

[13] J. T. Linderoth and S. J. Wright. Implementing a decomposition algorithm for stochastic programming on a computational grid. *Computational Optimization and Applications*, 24(2–3): 207–250, 2003.

[14] R. Lougee-Heimer. COIN-OR in 2008. *OR/MS Today*, 35(5): 46, 2008. Available at www.lionhrtpub.com/orms/orms-10-08/frcoin-or.html.

COIN-OR

(continued from page 3 ◁)

The availability of COIN-OR's robust, open-source optimization software was critical to the success of this project. COIN-OR enabled the rapid development of rigorous optimization techniques that identify sensor placements with quantifiable performance guarantees. A specific goal of EPA's TEVA Research Project is to develop open-source software that promotes the security of water distribution systems. The EPA plans to release the TEVA-SPOT software, using COIN-OR, to encourage the use of CWS design tools in the water industry.

New Project: Couenne for Nonconvex MINLP. Pietro Belotti (Lehigh University), Andreas Waechter (IBM), Pierre Bonami (Universite de la Mediterranee), Jon Lee (IBM), and Francois Margot (Carnegie Mellon University) have announced the release of the Couenne project on COIN-OR. Couenne is a Branch-and-Bound algorithm for nonconvex Mixed-Integer Nonlinear Programming (MINLP) problems; its purpose is to find global optima of non-convex MINLPs, and it implements reformulation and convexification techniques to obtain valid lower bounds, as well as bound reduction algorithms and several branching techniques, including reliability branching. It relies, among others, on the COIN-OR projects Bonmin, Ipopt, Cbc, Clp, Cgl, and Osi. Future developments include an API for building MINLP models and the convexification of more complex operators, such as quadratic expressions.

New Java Interface Supporting COIN-OR Solvers Released on SourceForge. Thomas Schickinger announced that swIMP v0.9.1 has been released on Sourceforge and can be downloaded at http://swimp.sourceforge.net/. The swIMP project provides a Java wrapper for the Open Solver Interface (OSI) in COIN-OR. So far, swIMP has been tested with Clp, Cbc, GLPK, SYMPHONY, Vol and MOSEK. The new minor release v0.9.1 has been built and tested against the following recent versions of the COIN-OR solvers and GLPK (which is not available on COIN-OR): (i) CoinAll 1.0.0, which contains Clp 1.0.6, Cbc 2.0.0 and SYMPHONY 5.17, and (ii) GLPK 4.29.

Binaries and RPMs Available for Select Projects. Binaries are available for COIN-OR projects that have unit tests, documentation, and build on all supported platforms. The binaries are available individually and bundled together in the CoinAll package at https://projects.coin-or.org/CoinBinary. These binaries include standalone executables as well as the headers and libraries needed to build custom applications. Binaries are provided for Linux (both 32 and 64 bit, compiled with gcc and icc), Solaris, OSX, and Windows (both as native windows compile and under cygwin) platforms.

To further improve the ease of installation, COIN-OR now provides these same projects in RPM, one of the standard Linux packaging formats. The RPMs are available at http://www.coin-or.org/download/rpm for the RedHat Fedora 7, 8, 9, and 10, and Suse 11 distributions of Linux. The RPMs are available as part of the final testing phase. Production-level quality RPMs are planned for release by the ICS Conference in January 2009. Volunteers are encouraged to download and test the RPMs, to build RPMs for other distributions, to help improve the quality of the RPMs so that Linux distributions would include them by default, and to help build DEBs (the RPM equivalent for Debian-based distributions, such as Ubuntu). If you are interested in helping, you can find documentation, a bug submission page and source at https://projects.coin-or.org/CoinBinary.

Existing Projects Continue to Evolve. There are now more than 30 projects on COIN-OR which continue to evolve at their own pace. Many have enjoyed new releases in the past few months, and many exciting develops are in the works. Here are just a very few of the highlights that have been accomplished since spring 2008. This report is based on the self-reporting by Project Managers in the COIN-OR Annual Report.

- CSDP: 10 new papers in which CSDP has been used are available.
- DFO: A book of Derivative Free Optimization has been submitted for publication by Katya Scheinberg, Andrew Conn, and Luis Vicente and is expected early in 2009.
- GAMSlinks: New interfaces including one to the Optimization Services project have been added, and extensions to the support the GAMS Branch-and-Cut-Heuristic Facility have been added.
- IPOPT: A Java interface to Ipopt was contributed by Rafael de Pelegrini Soares (VRTech Industrial Technologies).
- Optimization Services: Support was added for the mixed-integer nonlinear solver, Bonmin, and a new feature was added that allows models to be built in MATLAB to call COIN-OR solvers using the Optimization Services project.

First COIN-OR "Vendor" Workshop at INFORMS. No-cost three-hour workshops are held the day prior to the INFORMS Annual Meeting to give attendees an extended opportunity to learn about products and services offered by conference exhibitors. This year, the first COIN-OR vendor workshop was held thanks to the organizing efforts of Brad Bell (University of Washington), Robin Lougee-Heimer, and Kipp Martin. The goal of the workshop was to provide hands-on experience and a gentle introduction to some of the many tools available on COIN-OR. During the workshop attendees learned about, and received a copy of, many of the software projects that are freely available from COIN-OR. Instruction and help for installing and running solvers for linear, integer, and nonlinear optimization on attendee's own laptop computers was provided, along with the guided opportunities to compile and link their code with select COIN-OR utilities. The workshop also included how-to's and hands-on experience for combining the COIN-OR algorithmic differentiation utility and nonlinear optimizer, IPOPT, as well as general background on open-source and directions for publishing new projects on COIN-OR. The workshop attracted 40 attendees, booting up 20 laptops.

Workshop on Open-Source Software at CPAIROR 2008. A 4-hour *Workshop on Open-Source Software for Integer and Constraint Programming* was organized by Robin Lougee-Heimer and Ionut Aron at CPAIOR 2008 conference. Its immediate goals were to give a more comprehensive description of the existing open source projects in the CP and IP communities, and to encourage contributions. The long term goals were to foster collaboration between the two communities and encourage platforms through which software that integrates techniques from both fields can be openly developed, distributed, tested, and improved. The workshop consisted of six 30-min presentations and an hour-long panel discussion. For more information on the workshop, see https://projects.coin-or.org/Events/wiki/CpAiOr2008. For more on the many other COIN-OR events at the INFORMS DC meeting, see https://projects.coin-or.org/Events/wiki/InformsDc.

2008 Annual Report Available. For the second year in the row, the COIN-OR Foundation has worked to compile the most comprehensive list possible of its very distributed activities by asking the community to post its progress on a public wiki. Community news is vital to COIN-OR's continued success. Many project managers depend on reports from users to justify their participation in open-source, seek funding, and attract new users. Sharing news on how you're using COIN-OR is easy way to give back to the community. To see the reports from the Foundation's committees, project managers, and users that comprise the 2008 Annual Report, visit http://www.coin-or.org/coin-or-foundation/records.html.

2008 ICS Prize

(continued from page 1 ◁)

There is no universally agreed-upon method for determining the allocation of goods to bidders in combinatorial auctions, thus limiting their practical use. The Day-Raghavan paper developed an approach for determining winners of the auction as well as payments that satisfy two important properties:

- the payment method is incentive-compatible for bidders to bid their true values for bundles of goods, thus avoiding a need for extensive knowledge of the bidding of their competitors, and avoiding substantial underbidding;
- the payments are in the core; that is, there is no coalition of bidders that would be willing to pay more for any bundle of goods than the prices charged to the winning bidders.

This paper overcomes some key weaknesses of the Vickrey-Clarke-Groves (VCG) auction mechanism, which is an alternative mechanism that is compatible with bidders bidding their true values. First, the VCG mechanism can result in very low payments, thus making auctions impractical. Second, the VCG mechanism can result in payments that are not in the core, which would be perceived as unfair by any coalition who has bid more on items than they are sold for.

The authors' research fits well within the interface of Operations Research and Computer Science in their modeling and computational achievements. They present a model that achieves the two properties given above. They also provide a practical solution approach, using column generation, for finding Pareto-optimal solutions. Their approach is practical for governmental combinatorial auctions and has already been used in the United Kingdom in auctions for allocating spectrum.

The Prize Committee consisted of James Orlin (Chair), MIT, Mike Trick, Carnegie Mellon, and Pascal Van Hentenryck, Brown University. For further information regarding the ICS Prize, visit http://computing.society.informs.org/prize.php.

2008 ICS Student Paper Award

(continued from page 1 ◁)

The Award Committee consisted of David Morton (Chair), University of Texas at Austin, Alper Atamturk, University of California, Berkeley, and Nick Sahinidis, Carnegie Mellon University. The ICS Student Paper Award is sponsored by the Mica Foundation of Denmark and is accompanied by a plaque and a $500 honorarium. For more information regarding the award, see http://computing.society.informs.org/prizeStudent.php.

News from Related Communities

ACM SIGecom: E-commerce
http://www.sigecom.org/exchanges/

ACM SIGEVO: Genetic and Evolutionary Computation
http://www.sigevo.org/newsletter.html

ACM SIGKDD: Knowledge Discovery in Data
http://www.kdnuggets.com/news/

ACM SIGMIS: Management Information Systems
http://www.sigmis.org/

ACM SIGMOD: Management of Data
http://www.sigmod.org/record

ACM SIGWEB: Hypertext, Hypermedia and the Web
http://www.sigweb.org/resources/links-cover.shtml

INFORMS Information Systems Society
infosys.society.informs.org/Publications/Newsletters/Current.pdf

INFORMS Simulation Society
http://www.informs-sim.org/

INFORMS Transportation Science & Logistics Society
http://castlelab.princeton.edu/wiki/index.php/TSL_newsletters

If you are a newsletter editor of a group that is relevant to OR & Computing and you want to exchange links, please send to the *ICS News* Editor at ICSnewsEditor@mail.informs.org.

Humor

Whew! Next time I'll read the documentation.

Sniglets — words that should be in the dictionary, but aren't.

analog-retentive — those people who obstinately cling to outmoded technology.

animousity — vigorously clicking your pointer device because a page is loading too slowly.

backward combatability — a property of hardware or software revisions in which previous protocols, formats, layouts, etc. are irrevocably discarded in favor of "new and improved" protocols, formats and layouts, leaving the previous ones not merely deprecated but actively defeated.

blurker — someone who reads a blog or blogs regularly but never comments or contributes to the discussion.

comperrandish — the emotion felt when a computer error message continuously pops up, but no programs are affected by it.

compudextrous — able to use keyboard and mouse with either hand.

docuphobia — fear of documentation.

e-mnesia — the condition of having sent or received an e-mail and having no recollection of it whatsoever.

funnify — make a dry, even boring, message humorous, such as what John Chinneck has done for my announcements.

graphware — software to solve graph problems.

java-vu — phenomenon of having seen your code before writing it.

negabytes per second (NBps) — a measure of data transfer that is so slow that it seems to be flowing backwards.

od hac — improvised for one specific purpose, in a kludgy manner (as an undesirable hack).

random excess memory — memory you were talked into buying in order to solve some problem that never did get resolved.

simulite — a simulation package without any interface.

software bloat — the result of adding new features to a program or system to the point where the benefit of the new features is outweighed by the extra resources consumed and complexity of use.

Vocabularian — a person who makes up new words.

Daffynitions

Algorithms — A band that features a former Vice President.

Bandwidth — Limited by the size of the stage.

Cache — Required when your credit card is maxed out.

Firmware — Software with permanent bugs hardwired into it.

GUI (pronounced "gooey") — What your computer becomes after spilling your coffee on it.

Keyboard — An instrument used for entering errors into a system.

Mathematical Program — What is on the marquee when an OR/CS person speaks.

Queueing expert — Someone who knows how to control wait.

Upgrade — Take old bugs out, put new ones in.

Acknowledgments

The Editor thanks all contributors and those who provided service to ICS. In preparing for the interview with Marcel Neuts, I received useful information from Ulf Grenander and some of Marcel's former students: Jef Teugels (Purdue, '67), Seshavadhani Kumar (Delaware, '83), and Mouli Chandramouli (Arizona, '90). I also received meeting information from Mary Magrogan, INFORMS Director of Subdivision Services. Thanks to Springer for sending a copy of the book to our reviewer. Thanks to Harlan Crowder for his pictures from the DC INFORMS meeting, which we use here. The cartoon to signify the *Humor* section is courtesy of John Zakour ▹. Last, but not least, I again thank my Assistant, Matthew MacLeod, for his help during the past two years.

Reminder
ICS Symposium is January 11–13, 2009
Charleston, South Carolina
http://ics09.meetings.informs.org/