informs

ICS NEWS

INFORMS Computing Society Newsletter

Inside This Issue

"One day I'm going to get help for my procrastination problem and research articles ..."

Message from the Chair

Willliam Cook
Industrial and Systems Engineering
Georgia Institute of Technology
bico@isye.gatech.edu

As they say, time flies.

I took over as the chair of the society in January 2012, following the great path set by Bob Vanderbei (2010/11), Robin Lougee-Heimer (2008/09), John Chinneck (2006/07), Ariela Sofer (2004/05), David Woodruff (2002/03), Irv Lustig (2000/01), Ramayya Krishnan (1998/99), Richard Barr (1996/97), and on back to our roots. My knowledge of ICS history goes back only as far as our online collection of newsletters: Volume 18, Number 2, 1997. It would be very nice to fill out the archive with the first 17 volumes, so if anyone has a collection they might be willing to scan in please let me know.

Many thanks to Ojas Parekh (Sandia Labs) for taking over as the ICS Webmaster. It is a tough job to learn the ins and outs of the EZ Publish system adopted by INFORMS, but, thanks to many tips from our old Webmaster Bill Hart, Ojas is now busy updating the many ICS Web pages. If you spot a dead link or incorrect information on our pages, please let Ojas know at odparek@sandia.gov.

I hope to see everyone at INFORMS Phoenix in October and at the ICS Conference in Santa Fe, January 2013!

Message from the Editor

Yongpei Guan
Industrial and Systems Engineering
University of Florida
guan@ise.ufl.edu

I would like to first bring your attention that there are lots of changes this year for the society. Officers, board of directors, and editorial board for the journal have been updated. Besides these, in this letter, we have our regular collection of updates— a Mathematical Programming Glossary update from Allen Holder and an IJOC update from John Chinneck. We also add a new project update for SCIP 3.0 provided by Timo Berthold. Please take a while reading the extended abstracts for the 2011 ICS prize and 2011 ICS best student papers. The 2012 ICS awardees will be announced at INFORMS Phoenix. In addition, please do not forget to learn the updates of your colleagues by reading the members in the news and consider attending ICS-2013. Finally, special thanks to Warren Powell for contributing an awesome article entitled A Unified Framework for Stochastic and Dynamic Programming for this newsletter.

ICS-2013

William E. Hart and Jean-Paul Watson
Sandia National Laboratories
{wehart,jwatson}@sandia.gov

The 13th INFORMS Computing Society Conference will take place January 5 - 8, 2013, in Santa Fe, New Mexico, at the Eldorado Hotel - just footsteps from the legendary, historic Santa Fe Plaza. ICS is focused on contributions at the interface of computer science, artificial intelligence, operations research, and management science. The conference organizers invite submissions discussing novel theoretical and applied research consistent with the ICS focus.

We especially encourage submissions targeting this year's conference theme: Modeling and Analytics for the Real World. Areas of special interest include: - Modeling Languages and Systems - Uncertainty Analysis - Resiliency and Robustness - Multi-Objective and Multi-Criteria Analysis - Decision-Support - Large-Scale Systems Analysis

Other interest areas relating to the ICS focus areas include: - Design and Analysis of Algorithms - Heuristic Search and Learning - Computational Probability and Analysis - Constraint Programming and Hybrid Optimization - Knowledge and Data Management - Simulation - Solvers - Telecommunications and Electronic Commerce - Applications in Biology, Medicine, and Health Care

The main conference will begin on Sunday, January 6. On January 5, the organizers will host the CooprFest 2013 workshop - details to follow!

Presentation-Only Submissions

An abstract of no more than 200 words should be submitted through the submission page at http://www.easychair.org/conferences/?conf=ics2013. Names and affiliations of all authors should be provided, with the presenting author listed first. All accepted presentation-only abstracts will be printed in the conference program. Important dates are shown as follows:

Submission deadline: **October 1, 2012**
Notification of acceptance: **November 1, 2012**
Author/presenter registration: **November 1, 2012**
Early registration: **November 1, 2012**
Hotel registration: **December 1, 2012**

If you have any questions at all, please do not hesitate to contact one of the co-chairs of ICS 2013:

William E. Hart, Sandia National Laboratories - wehart@sandia.gov
Jean-Paul Watson, Sandia National Laboratories - jwatson@sandia.gov

Up-to-date information concerning ICS can be found at: http://www.informs.org/Community/Conferences/ICS2013

Report on the INFORMS Journal on Computing

John Chinneck
Carleton University
joc@mail.informs.org

It's been a busy year for the INFORMS Journal on Computing! In the last year we've received 178 new paper submissions and 105 revised submissions for a total of 283 items for review. Thank goodness for our excellent editors who not only manage to keep on top of this flow, but are also able to recruit high-quality reviewers. We published 45 research articles and featured 11 online book reviews in the last four issues (we publish four times per year). And the print queue is still quite lengthy, though articles appear online quite quickly after acceptance. All of these are indicators of the popularity of the JOC among authors.

The JOC scores very well on the "Article Influence" score, which measures its impact on a field based on 5-year citations per article. On this measure eigenfactor.org ranks the JOC 14th out of 86 journals in its "Operations Research" category. Using a similar measure, ISI places us at 16th of 72 journals in their "Operations Research and Management Science" category. Statistics on our review times and citations are available via links on the page at http://www.informs.org/Pubs/IJOC/Stats-History.

We have two Level 1 sponsors this year (IBM T.J. Watson Research Center and the INFORMS Computing Society), and two Level 2 sponsors (GAMS Development Corporation and LINDO Systems Inc.). We greatly appreciate their assistance in helping fund the high quality work of the Journal.

We have added some new Associate Editors over the past year: Alfa Attahiru of the University of Manitoba is now assisting Winfried Grassmann in the "Computational Probability and Analysis" Area. Panagiotis Ipeirotis of the Stern School of Business at New York University is not assisting Alexander Tuzhilin in the "Knowledge and Data Management" Area. Sadly we lost a valued Associate Editor in Alberto Caprara who died in a mountaineering accident in April.

Of course the biggest news is that there will be a new Editor-in-Chief of the INFORMS Journal on Computing starting on January 1, 2013: David Woodruff of the University of California at Davis. David will already be familiar to you as the current Area Editor for "Heuristic Search and Learning" (since 1998). He also served as an Associate Editor from 1998 to 2002. So he has a long relationship with the Journal and a good understanding of our traditions and motivations. The JOC will be in good hands for years to come.

I will have served as the Editor-in-Chief for five and a half years when my second term comes to an end on December 31, 2012. It's certainly been interesting to watch developments at that fascinating interface of OR and CS! In those years we've revised the descriptions of several of the journal Areas, and added a new one (Applications in Biology, Medicine, and Health Care). Plus we added the Book Reviews. We also made the transition to online manuscript handling, developed a new web site, and then redesigned it as we move to a new content management system. But the best part has been working with the very fine and dedicated folks who are the heart and soul of journal publishing: the JOC editors, the referees, Managing Editor Kelly Kophazi and the INFORMS Publications staff. And of course the authors, without whom the Journal could not exist. Thanks to you all!

As always, we are looking for excellent research at that interesting intersection between operations research and computer science. Send us your best work: maybe you'll join that elite group of highly cited papers that influences the research community. You can find the JOC web site at http://www.informs.org/Pubs/IJOC.

INFORMS Journal on Computing – New Editor-in-Chief

Longtime ICS (and CSTS) member David Woodruff was recently named Editor-in-Chief of the INFORMS Journal on Computing. He chaired the ICS meeting in Monterey and has served as Chair of the ICS. Woodruff, who is a Professor in the Graduate School of Management at UC Davis, will take the reigns of the IJOC from John Chinneck in January 2013. This is his first term for the period of January 1, 2013 – December 31, 2015.

EIC Search Committee: W. David Kelton(Chair), Cheryl Gaimon, Ramesh Sharda, and Bob Vanderbei.

ICS Member Shoemaker Elected to National Academy of Engineering

Christine Shoemaker, Cornell's Joseph P. Ripley Professor of Engineering, has been elected to the National Academy of Engineering, among the highest professional distinctions for an engineer.

Shoemaker, a Professor in the Cornell School of Civil and Environmental Engineering (CEE) and School of Operations Research and Information Engineering, was cited "for development of decision-making optimization algorithms for environmental and water resources problems."

Her research focuses on cost-effective, robust solutions for environmental problems by using optimization, modeling and statistical analyses. This includes development of general purpose, numerically efficient nonlinear and global optimization algorithms utilizing high performance computing and applications to data from complex, nonlinear environmental systems. Shoemaker received her Ph.D. in mathematics under the supervision of Richard Bellman in optimal control, and she chose to focus on environmental applications.

Shoemaker's research is interdisciplinary; she has supervised PhD students from a number of fields including Operations Research and Information Engineering and Applied Mathematics, in addition to students from her home field CEE. She has had NSF funding from four different Directorates. Her projects are often in collaboration with other faculty and include physical and biological groundwater remediation, pesticide management, ecology, climate modeling, carbon sequestration, and surface water pollutant transport in large watersheds.

Professor Shoemaker is a Distinguished Member of the American Society of Civil Engineering (ASCE). She has been elected a Fellow in the following professional societies: American Geophysical Union (Hydrology section), ASCE, and INFORMS (Institute for Operations Research and Management Science). She also won a Humboldt Research Prize from Germany. She initiated and led a 9-year multidisciplinary international project (sponsored by SCOPE and United Nations Environment Program) that brought information and workshops about groundwater contamination to developing countries at a time (1987-1996) when those regions were doing little to prevent contamination from industrial chemicals. Such contamination is often irreversible or extremely expensive to remove because it is in groundwater, so prevention is the best strategy.

Prof. Shoemaker was the first woman faculty member in the Cornell College of Engineering to be promoted to tenure. In 1985 she was the first woman to be a Cornell Engineering Department Chairperson. She received a national award from the Society of Women Engineers in 1991 for her scholarship and efforts to encourage women engineers during years when there were very few woman students or faculty in engineering.

Membership in the National Academy of Engineering honors those who have made outstanding contributions to "engineering research, practice or education, including, where appropriate, significant contributions to the engineering literature," and to the "pioneering of new and developing fields of technology, making major advancements in traditional fields of engineering, or developing/implementing innovative approaches to engineering education."

2011 ICS Prize Goes to Hochbaum at Berkeley

The winner of the 2011 ICS Prize for the best English language paper dealing with the Operations Research/Computer Science interface is Dr. Dorit Hochbaum for her paper "Polynomial Time Algorithms for Ratio Regions and a Variant of Normalized Cut," which has been published in IEEE Transactions on Pattern Analysis and Machine Intelligence in 2010.

> This paper presents an efficient, highly novel approach for solving a group of well-known combinatorial optimization problems arising in computer vision and image analysis, including the ratio-regions problem and a variant of the normalized-cut problem. Each problem expresses two conflicting objectives by a single nonlinear, ratio objective. Such conflicting objectives could be, for example, the desire to cluster similar pixels together while limiting the total number of clusters. Although studied for over a decade, researchers have suspected these problems to be NP-hard and hence have proposed approximate, continuous-based algorithms for their solution. The author recast each problem as a single-parameter parametric integer program with monotone inequality constraints. For a fixed parameter, this integer problem can in turn be solved as a minimum-cut problem. Fortunately there are just a linear number of parameter break points to evaluate, and so the overall algorithm is fast in theory and effective in practice. In addition, the parametric approach provides information on the trade-off between the two conflicting objectives. Besides solving these specific problems, this paper also sheds light on the difficulty of other related NP-hard problems.
>
> In short, this paper provides a beautiful contribution to computer vision by taking a very innovative angle and demonstrating how to derive algorithms with strong performance guarantees and excellent experimental behavior.

Extended Abstract (contributed by Dorit Hochbaum): The normalized cut is a well known optimization criterion in image segmentation, named so by Shi and Malik [SM00]. It is defined on a graph where each object (pixel) is represented by a node, and there is a *similarity* weight between each pair of neighboring objects. The problem is a close relative of the Cheeger constant problem, [Chee70]. Both problems are NP-hard and within a factor of two of each other. Both problems have been addressed with the "spectral method" that involves finding the Fiedler eigenvalue. Much work in the literature is devoted to approximations and speedups of methods to find the eigenvector. I was presenting to my seminar class a 2006 Nature paper by Sharon et al. [SGS06] that gave a new aggregation method for solving approximately the eigenvector problem for the normalized cut problem. The problem in [SGS06] was stated as finding a subset S in the graph that minimizes the ratio of the capacity of the cut arcs separating it from its complement, divided by the total similarity between all nodes of the set S. In the presentation I put the integer programming formulation on the board, realizing that it is an integer program on monotone constraints. Such integer programs are solved in polynomial time, as shown in [Hoc02]. Obviously, this model is not the same as the normalized cut model, otherwise NP would have been equal to P. To distinguish it from NC we refer to it as normalized cut prime, NC'. The main content of the paper [Hoc10] is showing that the NC' criterion and another, related, optimization criterion, "ratio region", are polynomial time solvable.

The optimization criterion NC' has since been shown to be an effective clustering model. Not only is the quality of the solutions very high, and in some cases the algorithms are able to detect features, in images, that are hidden even to the human eye, but also the algorithms are computationally efficient. This efficiency makes it possible to process optimally images with millions of pixels and thus 3D images and videos, as in [FHY11]. Previously such images were impossible to analyze comprehensively.

One recent successful application of the new technique is in evaluating drug effectiveness and ranking of drugs according to their impact on individual cells, [HHY12]. Another, in [YFH12], enhances the physical properties of detectors with the algorithm that effectively removes noise. A study in [HLB12] investigating NC''s performance as compared to the spectral approach, shows that the NC' algorithm better approximates, in practice, the objective value of normalized cut. It also provides better quality visual results, meaning that the subjective quality of the clustering is better. In a way, this algorithm demonstrates that a more computationally heavy model, is not necessarily a more realistic or useful model. You can have your cake and eat it too: Get a better model and pay less in terms of run time!

References

[Chee70] J. Cheeger. A Lower Bound for the Smallest Eigenvalue of the Laplacian. *Problems in Analysis*, R.C. Gunning, ed., Princeton Univ. Press, 195–199,1970.

[FHY11] B. Fishbain, D.S. Hochbaum and Y. Yang. Video Segmentation and Target Tracking Using Pseudoflow Algorithm Through Joint Utilization of Intensity and Coarse Motion Data. *manuscript*, University of California, Berkeley, 2011.

[HHY12] Dorit S. Hochbaum, Chun-Nan Hsu, Yan T. Yang. Ranking of Multidimensional Drug Profiling Data by Fractional Adjusted Bi-Partitional Scores. *Bioinformatics* 28:12, 106–114, 2012.

[Hoc02] D. S. Hochbaum. Solving integer programs over monotone inequalities in three variables: A framework for half integrality and good approximations. *European Journal of Operational Research* 140:2, 291-321, 2002.

[Hoc10] D. S. Hochbaum. 2 Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:5, 889–898, 2010.

[Hoc10a] D. S. Hochbaum. Replacing spectral techniques for expander ratio and normalized cut by combinatorial flow algorithms." arXiv:1010.4535v1 [math.OC] Oct 2010.

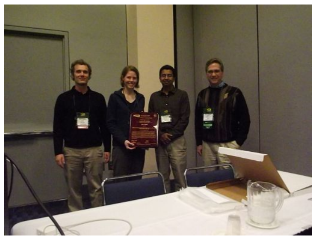
[HLB12] D. S. Hochbaum, C. Lu and E. Bertelli. 2012. Evaluating the Performance of Image Segmentation Criteria and Techniques. *manuscript*, University of California, Berkeley, 2012.

[SGS06] E. Sharon, M. Galun, D. Sharon, R. Basri and A. Brandt. 2006. Hierarchy and adaptivity in segmenting visual scenes. *Nature* 442, 810–813, 2006.

[SM00] Shi, J. and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

[YFH12] Y. Yang, B. Fishbain, D. S. Hochbaum, E. Norman, E. Swanberg. The Supervised Normalized Cut Method for Detecting, Classifying and Identifying Special Nuclear Materials. To appear, *INFORMS J Computing*.

2011 ICS Prize Committee: Sam Burer (chair), David Kelton, and Pascal Van Hentenryck



2011 ICS Best Student Paper Award Goes to Hunter at Virginia Tech

The 2011 ICS Student Paper Award Winner is Susan Hunter (Virginia Tech) for the paper "Optimal Sampling Laws for Stochastically Constrained Simulation Optimization on Finite Sets." Her advisor is Professor Raghu Pasupathy.

Extended Abstract (contributed by Susan R. Hunter and Raghu Pasupathy): The simulation-optimization (SO) problem is a nonlinear optimization problem where the objective and constraint functions, defined on a set of candidate solutions or "systems," are observable only through consistent estimators. The consistent estimators can be defined implicitly through a stochastic simulation model — a formulation that affords virtually any level of complexity. Due to this generality, the SO problem has received great attention from both researchers and practitioners in the last decade. Variations of the SO problem are readily applicable in such diverse contexts as vehicular transportation networks, quality control, telecommunication systems, and health care.

The SO variation we consider is an optimization problem having a finite number of candidate solutions characterized by multiple performance measures. One performance measure is primary and called the objective function, while the rest are secondary and called the constraint functions. For example, in the call-center staffing problem, we wish to select the staffing policy that minimizes the primary performance measure, the expected staffing cost, subject to a bound on the secondary performance measure, the expected waiting time for service. Another example is the vaccine allocation problem, in which we wish to select the vaccine allocation that minimizes the primary performance measure, the expected fraction of the population infected, subject to a bound on the secondary performance measure, the expected cost of the vaccine policy. Given that the performance measures can only be observed through a Monte Carlo simulation, the problem is to identify the solution having the best objective function value from among those solutions whose constraint values cross a pre-specified threshold, using only the simulation output. The efficiency of a solution to this problem is measured in terms of the total simulation effort expended.

The main technical contribution of this work lies in identifying the nature of simulation budget allocations that ensure efficient solution algorithms. Specifically, this work uses large-deviation principles to first characterize the likelihood of correctly identifying the solution to the SO problem as a function of the simulation budget allocation. This characterization is then used within an optimization problem to estimate the allocation that maximizes the likelihood of identifying the correct solution to the underlying SO problem. To aid implementation, the paper also presents a way to sequentially estimate the allocation as simulation observations progressively become available.

The broader implications of this work are threefold. First, this work provides a framework for assessing the efficiency of generic SO algorithms. Second, it provides an implementable mechanism that guides the efficient allocation of simulation expenditure when solving stochastically constrained SO problems on finite sets. For instance, when a tractable model of random variables comprising the simulation is assumed, the paper provides a practical way of apportioning simulation effort across the solution space, while guaranteeing efficiency in a certain precise sense. Third, this work constructs the basic building blocks for identifying optimal sampling strategies within more general constrained SO problems.

2011 ICS Student Paper Award Committee: Shabbir Ahmed (chair), Peter Frazier, and Dominique Orban.

ICS Members in the News

Miguel F. Anjos (miguel-f.anjos@polymtl.ca), Ph.D., was awarded a Tier 2 Canada Research Chair in Discrete Nonlinear Optimization in Engineering: http://www.chairs-chaires.gc.ca /chairholders-titulaires/profile-eng.aspx?profileId=2899. Tier 2 Chairs are awarded to exceptional emerging researchers, acknowledged by their peers as having the potential to lead in their field. Professor Miguel F. Anjos is currently the Canada Research Chair in Discrete Nonlinear Optimization in Engineering Mathematics & Industrial Engineering at Ecole Polytechnique de Montreal.

John Gunnar Carlsson (jgc@me.umn.edu), Ph.D., received a DARPA Young Faculty Award this year (http://go.usa.gov/Gxc) for his project entitled Strategically Allocating Resources in a Geographic Environment (SARGE). Dr. John Gunnar Carlsson is currently an Assistant Professor in Industrial and Systems Engineering at the University of Minnesota. He received his Ph.D. in Computational and Mathematical Engineering from the Stanford University in 2009.

Janos D. Pinter (janos.d.pinter@gmail.com), Ph.D., DSc, recently got his book, co-authored with Giorgio Fasano and entitled Modeling and Optimization in Space Engineering, published as Springer Optimization and Its Applications series, Volume 73 in 2012. Topical downloadable information is available at http://www.springer.com/mathematics/book/978-1-4614-4468-8. Dr. Janos D. Pinter is currently the Proprietor & Research Scientist at Pinter Consulting Services, Inc., Canada. He is also the chair for the EUROPT Managing Board, where EUROPT refers to The Continuous Optimization Working Group of EURO http://www.iam.metu.edu.tr/EUROPT/.

Report on the Mathematical Programming Glossary

Allen Holder, Rose-Hulman Mathematics
holder@rose-hulman.edu

The Mathematical Programming Glossary has had little change since last February, at which time the new wiki-like format was announced. The new format is an option from the introductory page, and it is receiving some traffic. Removing the autobot queries from the various search engines, the Glossary averaged about 22,000 hits per week, of which only about 600 were for the wiki-version. The pdf supplements were downloaded regularly at an average rate of about 330 per week. The Glossary has 73 citations as recognized by scholar.google.com.

The wiki-format was tested on several platform and browser combinations before releasing, but I recently noticed that the layout appeared odd with my antiquated versions of linux and Mozilla. This was one of the tested combinations, and I'm looking into what has changed. The IOL servers are being updated, and it could be that the latest version of php is serving the html code differently.

Suggestions for new terms are welcome. I want to again thank the constraint programming crew of Andrea Rendl, Serdar Kadioglu, Roger Kameugne, and J. Christopher Beck, who added many, many terms over the last couple of years.

SCIP Optimization Suite 3.0 Released

Timo Berthold
berthold@zib.de

Version 3.0 of the SCIP Optimization Suite has been released. It is comprised of SCIP 3.0, SoPlex 1.7, ZIMPL 3.3 and initial releases of GCG 1.0, a generic branch-cut-and-price solver, and UG 0.7, a parallel framework for mixed integer (non-)linear programming.

The SCIP Optimization Suite 3.0 provides many new features, performance improvements, and bug fixes. Some highlights are: six new presolving and propagation plugins, an iterative refinement procedure for computing high-precision LP solutions, further improved support of MINLP, and beta-versions of an AMPL and a MATLAB interface. The SCIP Optimization Suite can be obtained via http://scip.zib.de. The individual software packages and developers are listed below and available at the corresponding websites:

SCIP 3.0: S. Heinz, G. Gamrath, M. Pfetsch and T. Berthold
SoPlex 1.7: A. Gleixner and M. Miltenberger
ZIMPL 3.3: T. Koch
UG 0.7: Y. Shinano
GCG 1.0: G. Gamrath, M. Bergner and C. Puchert

A Unified Framework for Stochastic and Dynamic Programming

Warren B. Powell
Princeton University

Warren is a professor in the Department of Operations Research and Financial Engineering at Princeton University, where he has taught since 1981. He is director of CASTLE Laboratory (http://www.castlelab.princeton.edu) where he has worked on the modeling of complex dynamic systems in transportation and energy. He is the author of *Approximate Dynamic Programming: Solving the curses of dimensionality* and co-author of *Optimal Learning*. He recently started a new laboratory, PENSA, focusing on stochastic optimization in energy systems (http://energysystems.princeton.edu).

Stochastic programming and approximate dynamic programming have evolved as competing frameworks for solving sequential stochastic optimization problems, with proponents touting the strengths of their favorite approaches. With less visibility in this particular debate are communities working under names such as reinforcement learning, stochastic control, stochastic search and simulation-optimization, to name just a few. Put it all together and you get what I have come to call the jungle of stochastic optimization.

The competing communities working in stochastic optimization reflect the diversity of applications which arise in different problem settings, resulting in the development of parallel concepts, terminology and notation. Problem classes are distinguished by the nature of the decisions (discrete/continuous, scalar/vector), the underlying stochastic process, the transition function (known/unknown) and the objective function (convex? continuous?). Communities have evolved methods that are well suited to the problem classes that interest them. In the process, differences in vocabulary have hidden parallel developments (two communities doing the same thing with different terminology and vocabulary).

These differences have hidden important contributions that might help other communities. Computer scientists have ignored the power of convexity to solve problems with vector-valued actions. At the same time, the stochastic programming community has ignored the power of machine learning to approximate high-dimensional functions [8]. Years ago, I found that combining these two central ideas made it possible to solve a stochastic dynamic program with a decision vector with 50,000 dimensions and a state variable with 10^{20} *dimensions* [15]. In another problem, the same methods solved a stochastic, dynamic program with 175,000 time periods [11].

Stochastic programming, dynamic programming, and stochastic search can all be viewed in a unified framework if presented using common terminology and notation. One of the biggest challenges is the lack of a widely accepted modeling framework of the type that has defined the field of deterministic math programming. Misconceptions about the meaning of terms such as "state variable" and "policy" have limited dynamic programming to a relatively narrow problem class. For this reason, I will begin with a proposal for a common modeling framework which is designed to duplicate the elegance of "$\min cx$ subject to $Ax = b, x \geq 0$" that is so familiar to the operations research community. I then turn to the issue of defining what is meant by the word "policy." This article draws heavily on the ideas in [10].

Modeling stochastic, dynamic programs There are five elements to almost any stochastic, dynamic program: States, actions, exogenous information, the transition function, and the objective function.

- The state variable S_t - Incredibly, the dynamic programming literature seems to universally use the concept of a state variable without actually defining it. In [9][Chapter 5](downloadable from http://adp.princeton.edu), I suggest the following definition:

Definition: *A* **state variable** *is the minimally dimensioned function of history that is necessary and sufficient to compute the decision function, the transition function, and the cost/contribution function.*

It is useful to think of three types of state variables:

- The physical/resource state R_t - This captures the controllable elements of the problem which, in operations research, often refers to resources being managed. This might be the amount of inventory, the location of an aircraft, the status of a machine or the price of a stock. It might also be called the physical state, as in the location and velocity of a robot.
- The information state I_t - I_t includes exogenous information that is needed to compute the cost/contribution function, the decision function (such as the constraints), and the transition function. I_t would include exogenous information (such as market demands and wind speeds), as well as any relevant information from history that has already been observed and which is needed to make decisions in the future.
- The knowledge state K_t (also known as the belief state) - This is a set of probability distributions describing our knowledge about unobservable parameters. In some problem classes such as the multiarmed bandit problem, the belief state is the only state variable.

The stochastic programming community maintains a strict division between R_t, which always appears as a right-hand side constraint, and I_t, which is exogenous information that is not affected by current or prior decisions. However, there are applications where I_t is directly or indirectly influenced by decisions (large values of sales may influence random market prices).

An important concept is the *post-decision state* which we denote S_t^x (if our decision is x) or S_t^a (if we are using action a). This is the state at time t immediately after a decision is made, which includes *only* the decision needed to compute the transition function (since we have already computed the cost and made a decision), which has to capture the information needed for future decisions.

By requiring that the state variable be both necessary and sufficient, all stochastic dynamic systems are Markovian by construction. At the same time, we are going to open a fresh line of thinking for the stochastic programming community that depends on scenario trees which captures the entire history.

- Decision variables - It is useful to adopt three notational systems for decision variables: a_t for discrete actions (popular in computer science and the Markov decision process community), u_t for low-dimensional, continuous controls, and x_t for discrete or continuous but typically vector-valued decisions for problems common in operations research.

  When we specify a model, we avoid the problem of determining a decision other than to specify that it is the result of a *policy* which is a function that maps states to actions. We treat the determination of policies as separate from the model. While it is common to refer to a policy as $\pi(s)$, I prefer to write it as a function $X_t^\pi(S_t)$ (if it is a time-dependent function to determine x_t) or $A^\pi(s)$ (if it is a stationary policy to determine an action a). In this notation, π determines the type of function and any tunable parameters.

- Exogenous information W_t - There seems to be no common notation for the random variables in a stochastic system. Control theorists like w_t, probabilists prefer capital letters, so W_t seems like a compromise. However, unlike the control theory community, we insist that any variable indexed by t is known ("measurable") at time t. I have also adopted the notational convention of putting hats on specific random variables, such as $\hat{D}_{t+1}$ for new customer demands or $\hat{p}_{t+1}$ for the change in prices.

- Transition function - Also known as the system model, state model, plant model, transfer function or just "model," this describes the evolution of the system over time, written as

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

  Note that S_t is assumed to be fully known at time t, x_t is a decision that depends on the known information in S_t, while W_{t+1} is random at time t.

  In operations research, the transition function is often written as a system of linear equations, although only for the portion of the state variable that is controllable. We might write the resource transition function as

  $$R_{t+1} = S^R(R_t, x_t, W_{t+1}) = A_t x_t + \hat{R}_{t+1},$$

  while the information transition function $I_{t+1} = S^I(I_t, \cdot, W_{t+1})$ might represent systems of equations such as

  $$\begin{array}{rcll} p_{t+1} & = & p_t + \hat{p}_{t+1}, & \text{prices} \\ E_{t+1} & = & E_t + \hat{E}_{t+1}. & \text{energy from wind.} \end{array}$$

  There are many applications in engineering (such as modeling a chemical plant) where the transition function might consist of "500 lines of Matlab code." In others (such as modeling the effect of tax policy on climate change) the transition function is completely unknown. Problems with an unknown transition function are referred to as "model free."

- The objective function - Using the standard notation of dynamic programming, we write the cost function as $C(S_t, x_t)$ to reflect the possible dependence of costs on the state variable (costs, such as prices, can be stochastic). Assuming we are minimizing the expected sum of costs, we would write the objective function as

$$\min_{\pi\in\Pi} \mathbb{E} \sum_{t=0}^{T} C(S_t, X_t^\pi(S_t)). \tag{1}$$

We have written the objective (1) assuming an undiscounted, finite horizon formulation, which is more familiar to the stochastic programming community, but we could have adopted a discounted, infinite horizon objective, or one that minimized average costs. The objection function in (1) must be accompanied by the transition function, and a model of the exogenous process for $W_1, W_2, \ldots$.

There is a tendency to equate "dynamic programming" with Bellman's optimality equation. In fact, a number of authors will "model" a dynamic program by writing out Bellman's optimality equation:

$$V(s) = \min_a \left(C(s,a) + \gamma \sum_{s'} P(s'|s,a)V(s') \right).$$

A dynamic program, however, is a sequential decision problem, given by (1). Bellman's optimality equation is a) a way of mathematically characterizing an optimal policy and b) a strategy for designing approximate policies. The challenge we now face is the problem of specifying what it means to search over the policies $\pi \in \Pi$.

Policies The dynamic programming community is very familiar with the word "policy" which is widely understood to mean a mapping from states to actions. This is one reason why the state variable *must* include *all* the information needed to make a decision, compute the cost function and compute the transition function. There are many papers in dynamic programming where the policy is written in a lookup table form: given a discrete state s, here is the discrete action a. However, there is no reason to limit the definition of policies to such a restricted format.

Policies come in many forms, but in my work I have been able to divide them into four fundamental classes:

- Myopic cost function approximations - A myopic policy is of the form

$$X^M(S_t) = \arg\min_{x_t\in\mathcal{X}_t} C(S_t, x_t),$$

where $\mathcal{X}_t$ is the feasible region which may depend on S_t. In some settings, we can modify the problem to get better results over time, either by modifying the cost function itself, or possibly by modifying the constraints. We can represent this using

$$X^{CFA}(S_t|\theta) = \arg\min_{x_t\in\mathcal{X}_t(\theta)} C^\pi(S_t, x_t|\theta).$$

where θ represents any tunable parameters needed to adjust the function.

- Lookahead policies - Also known as rolling/receding horizon procedures, tree search, roll-out policies and, in control theory, model-predictive control, lookahead policies are popular in engineering practice. A deterministic lookahead policy involves solving

$$X_t^{LA-Det}(S_t) = \arg\min_{x_t\in\mathcal{X}_t} \left(c_t x_{tt} + \sum_{t'=t+1}^{t+H} c_{t'} x_{tt'} \right), \tag{2}$$

where $\arg\min_{x_t}$ optimizes over the entire (deterministic) vector $x_t = (x_{tt}, \ldots, x_{tt'}, \ldots, x_{t,t+H})$ over a specified horizon H, but the decision function $X_t^{LA-Det}(S_t)$ captures only x_{tt}. Of course, this has to be solved subject to constraints at each point in time and across time periods (we represent these constraints in $\mathcal{X}_t$). The stochastic programming community likes to incorporate uncertainty in the lookahead model, and solves

$$\begin{aligned} X_t^{LA-SP}(S_t) &= \arg\min_{x_t} \Big(c_t x_{tt} + \\ &\quad \sum_{\omega\in\hat{\Omega}_t} p(\omega) \sum_{t'=t+1}^{t+H} c_{t'}(\omega) x_{tt'}(\omega) \Big). \end{aligned} \tag{3}$$

Here, $\hat{\Omega}_t$ represents a subset of random outcomes over the interval t to $t+H$. Equation (3) is a classical two-stage stochastic programming formulation, where we first choose x_{tt}, then observe ω (which might be a sequence of random variables over time periods $t+1, \ldots, t+H$), and then choose $x_{tt'}(\omega)$ for all $t' > t$ given ω.

- Policy function approximations - PFAs are used when the structure of the policy $X^\pi(S_t)$ (or more likely $A^\pi(S_t)$) can be written in some analytic form. One example is our (q, Q) inventory re-ordering policy which we can write

$$A^\pi(R_t|\theta) = \begin{cases} 0 & \text{If } R_t \geq q, \\ Q - R_t & \text{If } R_t < q, \end{cases} \tag{4}$$

where $\theta = (q, Q)$. An alternative is to use a statistical model. If x_t and S_t are scalar, we might use

$$X^\pi(S_t|\theta) = \theta_0 + \theta_1 S_t + \theta_2 S_t^2.$$

Other examples include computing a control u_t using a statistical model such as a neural network (popular in engineering). The distinguishing feature of a PFA is that it does not have an imbedded optimization problem.

- Policies based on value function approximations - VFA policies are based on Bellman's equation, and have the form

$$X_t^{VFA}(S_t) = \arg\min_{x_t \in \mathcal{X}_t} (C(S_t, x_t) + \bar{V}_t(S_t^x)). \tag{5}$$

  Here, we have used the concept of the post-decision state variable S_t^x which avoids the imbedded expectation common in dynamic programming. A strategy that has attracted considerable attention under the name "approximate dynamic programming" (but studied widely in the reinforcement learning community) is to approximate $\bar{V}_t(S_t^x)$ using a linear model of the form

$$\bar{V}_t(S_t^x) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(S_t^x), \tag{6}$$

  where the independent variables $\phi_f(S_t^x)$ are known as "basic functions" or "features." However, a value function approximation can be represented using Benders cuts, linear approximations (linear in the state variable), or piecewise linear, separable approximations.

Three of the four classes of policies involve the use of a function approximation (cost function, policy function and value function). There are three fundamental ways of approximating functions: lookup tables (a special case), parametric models and nonparametric models. While lookup tables can be written as a parametric model, computationally they behave more like nonparametric models. In addition, a powerful class of models are known as semiparametrics, which are basically a hybrid of parametric and nonparametric. Not to be overlooked are approximation strategies aimed specifically at approximating convex problems which include strategies such as Benders' cuts, piecewise linear functions and shape-restricted statistical learning methods.

It is possible (and popular) to mix and match these fundamental strategies to create hybrids. Examples include a deterministic lookahead policy with value functions as terminal rewards, lookahead policies using cost function approximations, value functions with policy function approximations [17], and stochastic programming with scenario trees and policy function approximations [2].

The search over policies $\pi \in \Pi$ in equation (1) might first involve a search over major policy classes (including hybrids). Then, each class of policy (or hybrid) tends to be characterized by a vector of parameters that we call θ. In a stochastic programming model, θ might include the planning horizon or the strategy for generating scenarios. In a policy function approximation, θ would be the parameters (or rules) that specify the policy.

It is possible that someone has worked with a policy that does not fit any of these four (or hybrids), but this is what I have encountered in my own work. It is easy to describe problems that are particularly well-suited to each of these four classes. The important idea is to get away from equating dynamic programming with lookup tables.

With our new vocabulary and perspective, it is possible to identify close relationships between communities that have previously been viewed as completely distinct. Below I show how to link stochastic search and dynamic programming, and then dynamic programming and stochastic programming.

From stochastic search to dynamic programming

Problems in stochastic search are typically written

$$\min_x \mathbb{E} F(x, W), \tag{7}$$

where x is a deterministic set of parameters and W is a random variable. Stochastic search has been viewed as distinctly different from sequential decision problems (dynamic programs) because decisions x_t in sequential problems are random variables in the future. However, this is not the right way to look at them. In a dynamic program, the optimization problem is (in most cases) a search for a deterministic *policy*. We might write

$$F^\pi(S_0) = \mathbb{E}\hat{F}^\pi(S_0, W) = \sum_{t=0}^{T} C(S_t, X_t^\pi(S_t)),$$

where $S_{t+1} = S^M(S_t, X_t^\pi(S_t), W_{t+1})$. The optimization problem is then

$$\min_\pi \mathbb{E}\hat{F}^\pi(S_0, W). \tag{8}$$

Finding the best deterministic policy (which includes finding both the structure of the policy and any parameters that characterize the policy) corresponds directly to the stochastic search problem in (7).

This perspective opens up a powerful set of tools for solving dynamic programs. For example, imagine that we have a policy given by

$$X_t^{VFA}(S_t|\theta) = \arg\min_{x_t \in \mathcal{X}_t} \left(C(S_t, x_t) + \sum_{f \in \mathcal{F}} \theta_f \phi_f(S_t^x) \right).$$

While there is a substantial literature that tries to estimate θ using Bellman error minimization (so that $\bar{V}(S_t)$ predicts the value of being in a state), growing attention has been given to the idea of directly searching for the best value of θ to minimize costs. This is the same as solving equation (8) with $\pi = \theta$ which, of course, is the same as solving equation (7).

Stochastic search (known as *direct policy search* in the ADP/RL literature), can work extremely well, but it is not well suited to all problems. If we are unable to compute derivatives with respect to θ, then algorithms for derivative-free stochastic optimization generally limit the number of dimensions of θ. Particularly difficult are time-dependent problems where θ_t is a function of time. However, anyone working in dynamic programming should have a working knowledge of the tools of stochastic search.

From dynamic programming to stochastic programming

The transition from dynamic programming to stochastic programming is somewhat more difficult if done carefully. There has been growing recognition of the links between stochastic programming and dynamic programming (see [3] and [13]). Thorough presentations of stochastic programming can be found in [5], [1], and [14].

We start with the classical statement of Bellman's equation for discrete state and action spaces (see [12])

$$V(s) = \min_{a \in \mathcal{A}} \left(C(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V(s') \right).$$

The stochastic programming community works purely with time-dependent, finite horizon problems which we can write as

$$V_t(S_t) = \min_{a_t \in \mathcal{A}_t} \left(C(S_t, a_t) + \gamma \sum_{s' \in \mathcal{S}} P(S_{t+1} = s'|S_t, a_t) V_{t+1}(s') \right).$$

We next replace the one-step transition matrix with an expectation, giving us

$$V_t(S_t) = \min_{a_t \in \mathcal{A}_t} \left(C(S_t, a_t) + \gamma \mathbb{E}\{V_{t+1}(S_{t+1})|S_t\} \right), \qquad (9)$$

where $S_{t+1} = S^M(S_t, a_t, W_{t+1})$. There is an implicit assumption that we are using an optimal policy starting at time $t+1$ onward. We are going to temporarily fix the policy represented by $A^\pi_{t'}(S_{t'})$, and replace the value function with the expected sum of costs from $t+1$ and the end of the planning horizon,

$$V^\pi_t(S_t) = \min_{a_t \in \mathcal{A}_t} \left(C(S_t, a_t) + \mathbb{E}\left\{ \sum_{t'=t+1}^{T} \gamma^{t'-t} C(S_{t'}, A^\pi_{t'}(S_{t'})) \middle| S_t \right\} \right).$$

We cannot compute the expectation, so we are going to replace it with a Monte Carlo sample. This is also a good time to switch from our discrete action a_t to vector-valued actions x_t. This gives us

$$\begin{aligned} \bar{V}^\pi_t(S_t) \quad = \quad & \min_{x_t \in X_t} \Big(C(S_t, x_t) + \\ & \frac{1}{|\hat{\Omega}|} \sum_{\omega \in \hat{\Omega}} \sum_{t'=t+1}^{T} \gamma^{t'-t} C(S_{t'}(\omega), X^\pi_{t'}(S_{t'}(\omega))) \Big). \end{aligned}$$

The state $S_{t'}$ consists of two components: the resource state $R_{t'}$ that is controlled by our decisions $x_t, \ldots, x_{t'}$, and the exogenous information that we previously introduced as $I_{t'}$. In stochastic programming, it is common practice to view the exogenous information state as the entire history $h_{t'}$, and we will continue to reference the history $h_{t'}$ to develop the link with stochastic programming. Thus, we can write $S_{t'} = (R_{t'}, I_{t'}) = (R_{t'}, h_{t'})$.

We are primarily interested in making a decision at time t, so we are going to make the transition from approximating a policy over the entire simulation to time T to solving a *lookahead model* over the interval t to $t+H$, where H is the planning horizon. For example, we may want to model an energy storage problem over an entire year by solving sequences of 24-hour optimization problems. However, even this shorter horizon problem can be exceptionally difficult, so we have to develop special algorithmic strategies.

Remembering that we are starting at time t in a given initial state S_t, we can write our history as

$$h_{tt'} = (S_t, W_{t+1}, W_{t+2}, \ldots, W_{t'}),$$

where t' ranges from t to $t+H$. We are going to drop the reference to the unspecified policy $X^\pi_{t'}(S_{t'})$ and allow ourselves to choose actions in the future optimally, given the histories $h_{tt'}(\omega)$, $\omega \in \hat{\Omega}$. At this point we are also going to recognize that this optimization problem is a lookahead policy determined by solving the optimization problem

$$\begin{aligned} X^\pi_t(S_t) = & \arg\min_{x_{tt}} \Big(C(S_t, x_{tt}) + \\ & \min_{x_{t,t+1},\ldots,x_{t,t+H}} \frac{1}{|\hat{\Omega}|} \sum_{\omega \in \hat{\Omega}} \sum_{t'=t+1}^{t+H} \gamma^{t'-t} C(S_{t'}(\omega), x_{tt'}(h_{tt'}(\omega))) \Big), \end{aligned}$$

where there is a vector $x_{tt'}$ for each history $h_{tt'}(\omega)$, and where the decision x_{tt} affects the constraints for $x_{t,t+1}, \ldots, x_{t,t+H}$. We are now solving a stochastic optimization problem over a limited horizon H and a limited set of outcomes $\hat{\Omega}$, and yet even this problem is quite difficult. We are going to make one last tweak to get it into the more compact form

$$X^\pi_t(S_t) = \arg \min_{x_{tt},\ldots,x_{t,t+H}} \frac{1}{|\hat{\Omega}|} \sum_{\omega \in \hat{\Omega}} \sum_{t'=t}^{t+H} \gamma^{t'-t} C(S_{t'}(\omega), x_{tt'}(\omega)), \quad (10)$$

where $x_{tt}, \ldots, x_{t,t+H} = (x_{tt}(\omega), \ldots, x_{t,t+H}(\omega)),\ \forall \omega \in \hat{\Omega}$. We need to solve this optimization problem subject to the constraints for $t' = t+1, \ldots, t+H$ and all $\omega \in \hat{\Omega}$,

$$\begin{aligned} A_t x_{tt}(\omega) &= b_t, && (11) \\ x_{tt}(\omega) &\geq 0, && (12) \\ A_{t'}(\omega) x_{tt'}(\omega) - B_{t'-1}(\omega) x_{t,t'-1}(\omega) &= b_{t'}(\omega), && (13) \\ x_{tt'}(\omega) &\geq 0. && (14) \end{aligned}$$

In this formulation, we interpret $x_{tt'}(\omega)$ exactly as it is written - a single vector $x_{tt'}$ for each outcome $\omega \in \hat{\Omega}$ rather than the history $h_{tt'}(\omega)$. Instead of having one vector $x_{tt'}$ for each history $h_{tt'}$ (which is a node in the scenario tree), we now have a vector $x_{tt'}$ for each $\omega \in \hat{\Omega}$. This creates a problem that we are allowed to see into the future, so we add the *nonanticipativity constraints*

$$x_{tt'}(\omega) - \bar{x}_{tt'}(h_{tt'}) = 0, \quad \forall \omega \in \mathcal{H}_{tt'}(h_{tt'}), \tag{15}$$

where $\mathcal{H}_{tt'}(h_{tt'})$ is the set of outcomes $\omega \in \hat{\Omega}$ where the observations $W_{t+1}, \ldots, W_{t'}$ match $h_{tt'}$.

The challenge with this strategy is that the scenario tree (the set of histories) explodes very quickly. A common strategy is to generate a set of outcomes for W_{t+1}, and then sharply limit further branching in the scenario tree for later time periods ([1], [14], [7]). We can do this because we are only interested in the decision x_{tt} that we are going to implement now.

Given the complexity of solving the lookahead policy, there is by now a substantial research literature that has grown up around various strategies for finding near-optimal solutions. However, it is important to realize that an optimal solution of (10) does not mean the resulting policy is optimal. Indeed, bounds on the quality of the solution to (10) do not directly translate to bounds on the performance of the policy in equation (1). This is true even if we choose $t + H = T$, because the lookahead model is limited by the use of a scenario tree.

While explicit lookahead policies are popular in stochastic programming, considerable attention has been given to using a particular class of value function approximations for solving the lookahead model. We use the notational system of [14] to work backward from a formulation used in stochastic programming to the notation we have been using in approximate dynamic programming. This progression starts with

$$Q_t(x_{t-1}, \xi_{[t]}) = \min_{x_t} \left(c_t x_t + \mathbb{E}\{Q_{t+1}(x_t, \xi_{[t+1]})|\xi_{[t]}\}\right). \tag{16}$$

Here, Q_t is called the *recourse function*, but this is just different terminology and notation for our value function V_t. $\xi_{[t]}$ is the history of the exogenous information process up to time t (which we refer to as h_t). The resource vector R_t is a function of x_{t-1} and, if we have an exogenous component such as $\hat{R}_t$, then it also depends on W_t (which is contained in $\xi_{[t]}$). This means that the state variable is given by $S_t = (R_t, h_t) = (x_{t-1}, \xi_{[t]})$.

We note that it is mathematically equivalent to use x_{t-1} instead of R_t, but in most applications R_t is lower dimensional than x_{t-1} and would be more effective computationally as a state variable. Indeed, we would argue that while x_{t-1} is a sufficient statistic to describe the resource state R_t, it is not necessary because it carries more dimensions than are necessary.

These observations allow us to write (16) as

$$\begin{aligned} Q_t(x_{t-1}, \xi_{[t]}) &= \min_{x_t} \left(c_t x_t + Q_t^x(R_t^x, h_t)\right) & (17)\\ &= \min_{x_t} \left(c_t x_t + Q_t^x(x_t, h_t)\right). & (18) \end{aligned}$$

In the notation of dynamic programming (but retaining the linear cost structure), this would be written

$$V_t(S_t) = \min_{x_t} \left(c_t x_t + V_t^x(S_t^x)\right). \tag{19}$$

Equation (18) is a deterministic optimization problem, which is much more amenable to solution using the tools of math programming. Our only challenge, then, is finding an approximation of $V_t^x(S_t^x) = Q_t^x(x_t, h_t)$. The most popular strategy in stochastic programming is to use Benders' decomposition, where $Q_t^x(x_t, h_t)$ is replaced with a series of cuts, producing the linear program

$$V_t(S_t) = \min_{x_t, v} \left(c_t x_t + v\right), \tag{20}$$

where

$$v \geq \alpha_t^k(h_t) + \beta_t^k(h_t) x_t(h_t), \quad \text{for } k = 1, \ldots, K. \tag{21}$$

It is standard notation in the stochastic programming community to index these parameters as $(\alpha_{t+1}^k(h_t), \beta_{t+1}^k(h_t))$ because the cuts are approximating the problem at time $t + 1$, but the parameters are $\mathcal{F}_t$-measurable, and for this reason it is more consistent with our notation to index them by time t.

The optimization problem (20) with (21) is a linear program indexed by the history h_t. The parameters $(\alpha_t^k(h_t), \beta_t^k(h_t))$ are generated by simulating our way to h_{t+1} from h_t, solving the optimization problem at node h_{t+1}, and then using the dual information to update the parameters $(\alpha_t^k(h_t), \beta_t^k(h_t))$. This type of updating is completely familiar to the approximate dynamic programming community. Indeed, it should be easy to see that we are approximating the recourse function around the post-decision state at time t given by $\mathbb{E}\{Q_{t+1}(x_t, \xi_{[t+1]})|\xi_{[t]}\} = V_t^x(S_t^x) = V_t^x(R_t^x, h_t) = V_t^x(x_{t-1}, h_t)$.

While Benders' cuts are very popular in the stochastic programming community, they are hardly the only way to approximate the value of the future. An alternative strategy is to use a function that is linear in the (post-decision) resource variable, as in

$$\bar{V}_t(S_t) = \min_{x_t \in \mathcal{X}_t} \left(c_t x_t + \sum_i \bar{v}_{ti} R_{ti}^x\right), \tag{22}$$

Yet another strategy is to use an approximation that is piecewise linear but separable in R_t^x, as in

$$V_t(S_t) = \min_{x_t \in \mathcal{X}_t} \left(c_t x_t + \sum_i \bar{V}_{ti}(R_{ti}^x)\right). \tag{23}$$

Note that the approximations $\bar{v}_{ti}$ and $\bar{V}_{ti}(R_{ti})$ are not indexed by the history h_t, making these methods computationally much more compact.

These approximation strategies highlight a critical difference that separates approximate dynamic programming from stochastic programming. The value function approximations (22) and (23) are calculated over the entire simulation $t = 0, \ldots, T$ rather than just within a planning horizon $t, \ldots, t + H$ (see [15] and [16] for illustrations). This can be an important property when we are interested in the value functions themselves. In [11], we estimate value function approximations of the form given in (23) for 175,000 time periods. This produces a policy of the form

$$X_t^\pi(S_t) = \arg\min_{x_t \in \mathcal{X}_t} \left(c_t x_t + \sum_i \bar{V}_{ti}(R_{ti}^x) \right), \tag{24}$$

where x_t is a vector with hundreds or even thousands of dimensions.

By contrast, the stochastic programming community is using Benders' cuts to form a "policy" for solving the lookahead model over time periods $t, \ldots, t + H$. The policy that is then used for the full model is the decision produced at time t from solving the lookahead model. After this decision, we roll forward to $t + 1$ and solve a new problem over the horizon $t + 1$ to $t + 1 + H$ and then repeat the entire process.

The only reason this is necessary is because of the use of the scenario tree which is generated from the current state S_t, and which limits the lookahead model to relatively shorter horizons. If we dropped the indexing of the cuts in (21) on the nodes in the scenario tree, then it would be possible to use Benders' cuts over the entire horizon $t = 0, \ldots, T$ in one large model. An alternative strategy would be to index the cuts based on clusters formed around the information state (rather than the entire history). The important idea is to draw on the tools of machine learning [4] rather than depending on lookup tables (nodes in the scenario tree).

Closing thoughts While both the stochastic programming and dynamic programming communities work on sequential decision problems, there are some important differences in cultures between the two communities. While it is important to understand differences in terminology and notation, it is also useful to understand the differences in motivating applications and research styles.

Research in the stochastic programming community seems to be characterized by:

- Works exclusively on time-dependent problems.
- Primary tools are explicit lookahead policies and value functions based on Benders' cuts to exploit convexity.
- Emphasizes convergence proofs and bounds on policies for solving an approximate lookahead model rather than the value of a policy over long horizons.
- Exploits convexity to handle vector-valued decisions.
- Depends heavily on the concept of scenario trees and lookup tables rather than general purpose statistical learning algorithms.

By contrast, the dynamic programming community seems to be characterized by:

- Most of the research seems to be on infinite horizon problems, but applications to time-dependent problems are common.
- Most attention is on policies based on value function approximations and policy function approximations.
- Focus is on finding the best policy in terms of its performance over a long (or infinite) horizon.
- More emphasis on discrete action spaces, or low-dimensional continuous controls, without assuming convexity.
- Extensive use of machine learning techniques to approximate value functions.

These are generalizations, of course, but represent my sense of these communities. My hope is that this discussion will help to bridge the gap.

References

[1] Birge, J. R. & Louveaux, F. (1997), *Introduction to Stochastic Programming*, Springer Verlag, New York.

[2] Defourny, B., Ernst, D. & Wehenkel, L. (2013), 'Scenario Trees and Policy Selection for Multistage Stochastic Programming using Machine Learning', *Informs J. on Computing* (to appear).

[3] Dupacova, J. & Sladky, K. (2001), 'Comparison of Multistage Stochastic Programs with Recourse and Stochastic Dynamic Programs with Discrete Time', *Z. Angew. Math. Mech.* **81**, 1–15.

[4] Hastie, T., Tibshirani, R. & Friedman, J. (2009), *The elements of statistical learning: data mining, inference and prediction*, Springer, New York.

[5] Higle, J. & Sen, S. (1996), *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming*, Kluwer Academic Publishers.

[6] Jacobs, J., Freeman, G., Grygier, J., Morton, D., Schultz, G., Staschus, K. & Stedinger, J. (1995), 'SOCRATES-A system for scheduling hydroelectric generation under uncertainty', *Ann. Oper. Res*, **59**, 99–133.

[7] King, A. J. & Wallace, S. (2012), *Modeling with stochastic programming*, Springer Verlag, New York.

[8] Powell, W. B. (2009), 'Feature Article–Merging AI and OR to Solve High-Dimensional Stochastic Optimization Problems Using Approximate Dynamic Programming', *Informs J. on Computing*, **22**, 2–17.

[9] Powell, W. B. (2011), *Approximate Dynamic Programming: Solving the curses of dimensionality*, 2nd. edn, John Wiley & Sons, Hoboken, NJ.

[10] Powell, W. B., Simao, H. P., Bouzaiene-Ayari, B. (2012), 'Approximate Dynamic Programming in Transportation and Logistics : A Unified Framework', *European J. of Transportation and Logistics*, (to appear).

[11] Powell, W. B., George, A., Lamont, A. & Stewart, J. (2011), 'SMART: A Stochastic Multiscale Model for the Analysis of Energy Resources, Technology and Policy', *Informs J. on Computing*.

[12] Puterman, M. L. (1994), *Markov Decision Processes*, 1st edn, John Wiley and Sons, Hoboken.

[13] Sen, S. & Zhou, Z. (2012), Multi-stage Stochastic Decomposition: A Sequential Sampling Algorithm for Multi-stage Stochastic Linear Programs.

[14] Shapiro, A., Dentcheva, D. & Ruszczynski, A. (2009), *Lectures on stochastic programming: modeling and theory*, SIAM, Philadelphia.

[15] Simao, H. P., Day, J., George, A., Gifford, T., Powell, W. B. & Nienow, J. (2009), 'An Approximate Dynamic Programming Algorithm for Large-Scale Fleet Management: A Case Application', *Transportation Science* **43**(2), 178–197.

[16] Topaloglu, H. & Powell, W. B. (2006), 'Dynamic Programming Approximations for Stochastic, Time-Staged Integer Multicommodity Flow Problems', *Informs Journal on Computing* **18**, 31–42.

[17] Wu, T., Powell, W. B. & Whisman, A. (2009), 'The Optimizing-Simulator: An Illustration using the Military Airlift Problem', *ACM Transactions on Modeling and Simulation* **19**(3), 1–31.

IJOC Seeks Book Reviews Editor

The IJOC Book Reviews Editor seeks and approves reviews of books at the interface between operations research and computer science. This position has a two year term and is an outstanding way to develop and maintain contact with researchers interested in these topics as well as contribute in an important and visible way to ongoing research. Detailed information can be observed at http://www.informs.org/Pubs/IJOC/Book-Reviews. If you are interested or if you would like to nominate someone, please email DLWoodruff@UCDavis.edu.

Acknowledgments

The Editor would like to thank all contributors who helped to make this newsletter available.