Skip all front matter: Jump to Page 1

Technical Report

Portfolio Optimization by Means of Multiple Tandem Certainty-Uncertainty Searches

A Technical Description

Brian G. Chow

RAND Corporation

RR-270-A/OSD
March 15, 2013

The research described in this report was conducted as part of a series of previously released RAND studies carried out in RAND Arroyo Center and the RAND National Defense Research Institute (NDRI).

Published 2013 by the RAND Corporation
1776 Main Street, P.O. Box 2138, Santa Monica, CA 90407-2138
1200 South Hayes Street, Arlington, VA 22202-5050
4570 Fifth Avenue, Suite 600, Pittsburgh, PA 15213-2665
RAND URL: http://www.rand.org
To order RAND documents or to obtain additional information, contact
Distribution Services: Telephone: (310) 451-7002;
Fax: (310) 451-6915; Email: order@rand.org

Preface

Conducting optimization under conditions of uncertainty has long been a very difficult problem. Thus, when analysts have done optimization under uncertainty, they have introduced severe limitations to restrict how uncertainties can be factored in. This paper describes a new approach to optimization under uncertainty that is aimed at finding the optimal solution to a problem by designing a number of search algorithms or schemes in a way that reduces the dimensionality constraints that analysts have had to contend with until now.

The specific purpose of this paper is to convert a provisional patent application entitled *Portfolio Optimization by Means of a Ranking and Competing Search* by the author into a published volume available for public use. The provisional patent application was filed with the United States Patent and Trademark Office on July 30, 2012. Given the goal of making this a publication for public use, this paper has been structured differently—more in accord with a scientific paper than a patent application. Also, some background materials and examples from the author's past studies have been added to illustrate the new approach and contrast it and its associated algorithms with those of existing approaches, and some editorial changes have been made to make it easier for general audiences to comprehend. The ideas and techniques presented in this paper may be used by anyone for any purpose with citation.

This paper may be of interest to designers of optimization search algorithms. Businesses in both the public and private sectors may also find this paper of use, because they can incorporate the new approach and the developed algorithms into their optimization models to better deal with the future, which is rife with uncertainty.

The research behind this new search approach and the multiple algorithms that go with it was conducted as part of a series of previously released RAND studies by Brian G. Chow, Richard Silberglitt, Scott Hiromoto, Caroline Reilly, and Christina Panis:

- *Toward Affordable Systems II: Portfolio Management for Army Science and Technology Programs Under Uncertainties* (2011)
- *Toward Affordable Systems III: Portfolio Management for Army Engineering and Manufacturing Development Programs* (2012).

These in turn built on a number of earlier studies on optimizing under conditions of certainty, including

- *Toward Affordable Systems: Portfolio Analysis and Management for Army Science and Technology Programs*, Brian G. Chow, Richard Silberglitt, and Scott Hiromoto (2009)
- *A Delicate Balance: Portfolio Analysis and Management for Intelligence Information Dissemination Programs*, by Eric Landree, Richard Silberglitt, Brian G. Chow, Lance Sherry, and Michael S. Tseng (2009).

The *Toward Affordable Systems* series was conducted in the RAND Arroyo Center and sponsored by the Deputy Assistant Secretary of the Army (Cost and Economic Analysis), Office of Assistant Secretary of the Army (Financial Management and Comptroller). The research behind *A Delicate Balance* was executed within the Acquisition and Technology Policy Center of the RAND National Defense Research Institute (NDRI) and sponsored by the National Security Agency.

RAND Arroyo Center, part of the RAND Corporation, is a federally funded research and development center (FFRDC) sponsored by the United States Army. Also an FFRDC, NDRI is sponsored by the Office of the Secretary of Defense, the Joint Staff, the Unified Combatant Commands, the Navy, the Marine Corps, the defense agencies, and the defense Intelligence Community.

Summary

American mathematician George Dantzig developed the simplex algorithm to solve linear programming problems, but he also pioneered solving such problems under uncertainty in 1955. To date, linear and nonlinear programming problems under uncertainty have been extensively studied. Those approaches that have found applications in businesses, whether in the public or private sector, have had to impose severe limitations on the numbers of decision variables, uncertain parameters, and uncertain scenarios that can be used. Otherwise, the combinatorial possible solutions would grow exponentially and prohibit even today's most powerful computers (or those in the foreseeable future) from exhausting all the possibilities in finding the optimal solution.

This paper introduces a new approach to allow these limitations to be greatly relaxed and describes a number of search algorithms or schemes that have been shown to have practical applications. This approach and its associated search algorithms have a key feature—they generate typically 10,000 uncertain scenarios or future states of the world according to their uncertainty distribution functions. While each of these scenarios is a point in the larger uncertainty space, the originally uncertain parameters are specified for the scenario and are, thereby, "determined" or "certain." Thus, the solvable mixed-integer linear programming model can be used "under certainty" (i.e., deterministically) to find the optimal solution for that scenario. Doing this for numerous scenarios provides a great deal of knowledge and facilitates the search for the optimal solution—or one close to it—for the larger problem under uncertainty. This approach allows one to decompose the problem under uncertainty into 10,000 solvable problems so that one can learn about the role each project plays in these 10,000 samples of the uncertainty space. Doing so allows one to avoid the impossible task of performing millions or trillions of searches to find the optimal solution for each scenario, yet enables one to gain just as much knowledge as if one were doing so.

The approach is to use transparent reasoning, as opposed to mathematical formulas, to design search schemes or algorithms to find the global optimum and not get trapped at one of the local optima. This approach relies on arguments from devil's advocates to uncover the shortcomings of an algorithm in terms of why under certain situations it will not lead to the global optimum. Once the weaknesses of a given algorithm are identified, hopefully the original algorithm can be modified to remove the shortcomings, or another algorithm can be designed to plug the reasoning hole of the original algorithm.

Experience with this approach has been good. However, if the shortcomings in these algorithms cannot be eliminated, this approach would have to rely on the simplicity of nonmathematical reasoning so that many analysts or even “crowd wisdom” can be used to devise completely different algorithms to do the job. Because all approaches, including this one, face the risk of potentially missing the global optimum, this approach based on reasoning can open a new way for drawing in talents from the nonmathematical world to devise search schemes to tackle this very difficult task of optimization under uncertainty.

These reasonable search algorithms are easy to understand. Implementing them amounts to creating a flow chart and does not require the use of complicated mathematics or formulas; as a result, the approach allows for wider adoption by analysts and organizations that possess different skill sets.

As described in two illustrative search schemes (SS-8 without replacement and SS-8 with replacement) discussed in the paper, the approach draws on the common-sense and commonly practiced ideas used in business decisions and daily activities.

The SS-8 without replacement search scheme is based on the idea of how to create a project team. Suppose a project sponsor has some “use-it-or-lose-it” money at the end of a fiscal year. While a project must be issued now within a broad study area, the sponsor will assign specific tasks over the course of the one-year project, but which tasks those will be is unclear. The company’s policy and the sponsor’s requirement, however, are such that the project leader must specify the team members at the project’s start, after which it will not be possible to change them. Under such circumstances, it makes sense to draw up a list of tasks that the sponsor *may* ask the project team to do and to start by then selecting the person (by analogy, the first project selected) suitable for the largest number of potential task combinations that can be anticipated. The next step would be to find the person (by analogy, the second project selected) to best complement the first in technical and managerial skills so that the pair is suitable for the largest number of possible task combinations. Similarly, the third person (by analogy, the third project selected) would be found to best complement the first two, and so on.

The second search scheme (SS-8 with replacement)[1] accounts for the possibility that if a different person were selected as the first person, the skill sets and personal

[1]The terms *without replacement* and *with replacement* refer to whether search is conducted with a unique choice of the first team member (the first version) or whether the search is conducted with each of the possible people as the first team member.

chemistry to complement this different person might lead to a team composition different from the one based on SS-8 without replacement.

Mathematically, these approaches are very convenient. Let there be N persons to choose from in forming the project team. Instead of looking at 2^N—or easily millions or trillions of possible team compositions—the two search schemes together would generate only N possible project teams (by analogy, any project could be the first, but thereafter the choices would be determined by looking at results for the uncertain scenarios used), thus enormously reducing the complexity of the search.

This type of reasoning in designing and using complementary algorithms can give analysts a way—which may either be more transparent than mathematics or at the least a supplement to it—to uncover and mitigate logical lapses. Also, analysts using this approach may feel more confident that, even if these algorithms do not find the global optimum, the local optima they find should be near the global one, because the logic of these algorithms are used often and work well in the analysts' other daily activities.

Applications of this approach were developed in a series of studies called *Toward Affordable Systems* (*Toward Affordable Systems II* and *Toward Affordable Systems III)* that were sponsored by the Deputy Assistant Secretary of the Army (Cost and Economic Analysis), Office of Assistant Secretary of the Army (Financial Management and Comptroller). This paper discusses two of these applications. Applications of this approach that have appeared in *Toward Affordable Systems II* involved 75 projects or decision variables and 75 independent uncertain parameters. Each parameter corresponds to a project that has a 90 percent chance of successful completion and a 10 percent chance of failure.

The applications that appeared in *Toward Affordable Systems III* involve 26 or 183 projects; 26 uncertain costs in procuring a system derived from each project; and one more uncertain parameter corresponding to the budget for acquiring, operating, and maintaining the needed systems.

The applications with 26 or 183 projects are compared to two typical approaches: benefit/cost ratios and mixed-integer linear programming. The comparison shows that this new approach will save money for any given confidence level for meeting requirements or will yield higher confidence for equal cost.

Each of the algorithms developed in this paper takes minutes or hours to find the optimal solution, even for uncertainty problems involving substantially more decision variables (75 used here versus typically ten in other methods), uncertain parameters (75 used here versus typically a few in other methods), and uncertain scenarios (10,000 used here versus a few in other methods—or, alternatively, more than 10,000 used but then restricting variables and/or parameters to around ten) than other methods would have allowed. Thus, the relatively shorter run time offers the possibility of performing several complementary algorithms for the same problem, enhancing the chance of finding the global optimum.

The objective function is chosen to be the highest chance of meeting the requirements within a budget. This can be a way to introduce the idea of a confidence level in dealing with uncertainty. Then again, if analysts prefer using more conventional objective functions, such as minimizing expected cost or regret, this method can be modified to use such objective functions with other changes in the formulation and search algorithms, which may be akin to something as simple as moving from a simple average to weighted sums.

This approach is also suitable for parallel computing, because the 10,000 runs for each data point, the runs of different data points, and the runs for different algorithms can all be performed independently and simultaneously. The current advances in parallel computing and the rapid decline in cost of on-demand computer power favor this multiple search approach.

This paper proposes a common platform so that solutions derived from different approaches and search algorithms can be objectively compared to determine which gives the best solution. This implies that the platform is supported by a library of test problems with known solutions so that different algorithms can be tested and compared. As the platform and its database accumulate more and more comparisons, there will be better confidence about which algorithm works the best for which types of problems.

This paper also proposes to extend the applications of the approach and associated algorithms in several dimensions:

- Apply it to different types of problems beyond the current focus on project portfolio problems under uncertainty. One may start the expansion with other resource allocation problems, such as production planning.
- Use other objective functions for the uncertainty problem, such as the minimization of expected total cost or regret.
- Program the multiple search algorithms for parallel computing to shorten the run time.

Finally, this paper proposes a systematic examination of approaches and their search algorithms, with the goal of combining their individual strengths and mitigating their weaknesses to give users ways to better perform optimization under uncertainty.

Because uncertainties are inherent in input data and the future, better ways to factor uncertainties into consideration are critically important for any type of decisionmaking.

(This page is intentionally left blank.)

Abbreviations

ASIC	application-specific integrated circuit
C	number of constraints
CIFS/SMB	common Internet file system/server message block
CPLEX	IBM ILOG CPLEX Optimization Studio
CPU	central processing unit
CRT	cathode ray tube
CSV	comma-separated values
EEPROM	electrically erasable programmable read-only memory
EPROM	erasable programmable read-only memory
FP	feasible percentage
FPGA	field programmable gate array
FSWs	future states of the world
GAMS	General Algebraic Modeling System
GPS	global positioning system
HTML	HyperText Markup Language
IMP	implementation budget
LAN	local area network
LCD	liquid crystal display
N	number of projects
OP	optimal portfolio or global optimal portfolio
PDA	personal digital assistant
Q	number of uncertain input parameters
R	number of requirements
R&D	research and development
RAM	random access memory
RHEL	Red Hat Enterprise Linux
RLC	remaining life-cycle budget
ROM	read-only memory
RP	number of projects outside the working optimal portfolio
RRD	remaining research and development budget

scenario	short for uncertain scenario
SP	number of projects in the working optimal portfolio
SS	search scheme
SS8-NR	Search Scheme 8 without Replacement
SS8-SR	Search Scheme 8 with Single Replacement
SSH	Secure Shell Protocol
TIMP	total implementation
TR_j	minimum total value that all selected projects must provide to meet requirement j
TRLC	total remaining life-cycle
TRRD	total remaining research and development
USB	universal serial bus
WAN	wide area network
WiMAX	worldwide interoperability for microwave access
WOP	working optimal portfolios

Acknowledgments

I wish to thank Christina Panis for her able programming, testing, and running multiple algorithms. I benefited greatly from the reviews by Paul Davis and Richard Hillestad. They offered numerous comments and suggestions that led me to address critical issues and made this a better paper. I also wish to thank Bruce Held, Tim Bonds, Susan Marquis, Michael Greenberg, and Bill Welser for their encouragement and support. Paul Steinberg served as an excellent communications analyst and made this technical paper much more comprehensible. Last and most important, I am extremely grateful to Richard Silberglitt for inviting me to join with him in developing PortMan, a portfolio management framework, and allowing me total freedom in developing a new approach and the associated search algorithms. Without him, there would have been no paper.

(This page is intentionally left blank.)

Chapter One: Introduction

Background

Nikolaos Sahinidis classified the theory and methodology that have been developed to cope with the complexity of optimization problems under uncertainty into three main approaches.[2] *Stochastic programming* covers the two-stage uncertainty programming paradigm in which the first-stage variables are those that have to be decided upon before the actual realization of the uncertain parameters at the second stage in the future. The second category is *fuzzy mathematical programming*. Unlike stochastic programming, fuzzy programming allows constraints to be violated within some lower and upper bounds. The third category is *stochastic dynamic programming*, which deals with multistage decision processes. The approach proposed in this paper belongs to the first approach—stochastic programming.

Stochastic programming itself was pioneered by Dantzig and Beale. Both independently proposed a stochastic model formulation in 1955.[3] Extensive studies have since followed.[4] The most important formulation for practical applications remains one that expresses the uncertainty problem in a linear programming model[5] in which both the objective function and constraints are linear with respect to the decision variables. Once the uncertainty problem is linearized, many developed methods, including Dantzig's

---

[2] Nikolaos Sahinidis, "Optimization Under Uncertainty: State-of-the-Art and Opportunities," *Computers and Chemical Engineering 28*, Elsevier Ltd., 2004, pp. 971–983. He further classified stochastic programming into recourse models, robust stochastic programming, and probabilistic models, and he further classified fuzzy programming into flexible programming and possibilistic programming.

[3] George Dantzig, "Linear Programming Under Uncertainty," *Management Science*, Vol. 1, Nos. 3 and 4, April–July 1955, pp. 197–206; and E. M. L. Beale, "On Minimizing a Convex Function Subject to Linear Inequalities," *Journal of the Royal Statistical Society*, Vol. 17, No. 2, 1955, pp. 173–184.

[4] As of February 27, 2013, a Google Scholar search indicated that Dantzig's paper had been cited 1,190 times, and Beale's paper had been cited 537 times. Some recent articles are Teemu Pennanen, "Convex Duality in Stochastic Optimization and Mathematical Finance," *Mathematics of Operations Research*, Vol. 36, No. 2, May 2011, pp. 340–362; Dimitris Bertsimas, Vineet Goyal, and Xu Andy Sun, "A Geometric Characterization of the Power of Finite Adaptability in Multistage Stochastic and Adaptive Optimization," *Mathematics of Operations Research*, Vol. 36, No. 1, February 2011, pp. 24–54; Anthony Man-Cho So, Jiawei Zhang, and Yinyu Ye, "Stochastic Combinatorial Optimization with Controllable Risk Aversion Level," *Mathematics of Operations Research*, Vol. 34, No. 3, August 2009, pp. 522–537; Xin Chen, Melvyn Sim, Peng Sun, and Jiawei Zhang, "A Linear Decision-Based Approximation Approach to Stochastic Programming," *Operations Research*, Vol. 56, No. 2, March–April 2008, pp. 344–357; and John Birge and Francois Louveaux*, Introduction to Stochastic Programming*, Second Edition, Springer Series on Operations Research and Financial Engineering, 2010.

[5] In this paper, a linear programming model—whether with or without mentioning "deterministic" or "certainty"—means the same thing. The seemingly superfluous descriptor is used to help better distinguish from models used under uncertainty.

simplex, can be applied to find the optimal solution.[6] However, the dimensionality (i.e., the number of variables, parameters, and scenarios) continues to be a limitation to the size of the uncertainty problem that can be dealt with.

A study by Zhang, Prajapati, and Peden is a good illustration of this dimensionality issue, and it is the focus of this paper, which aims to relax the dimensionality limitation. In studying production planning under uncertainty, they observed: "To perform an all-inclusive study of production planning with all the uncertainties, however, is impossible, and a more viable approach is to address one or a few uncertainties in a stochastic model to derive optimal solutions."[7] Consequently, they allowed only the demand to be uncertain. Moreover, they restricted the uncertain demand to only three uncertain scenarios, or future states of the world (FSWs).[8] They studied the other uncertain parameters through sensitivity analyses using one-by-one sensitivities around their solution. Unfortunately, sensitivity analysis can only tell one that the identified solution remains optimal with respect to uncertainties around it; using sensitivity analysis does not make the variables and parameters used in the sensitivity analysis a part of the optimization process itself. This paper aims for a new approach that allows optimization to be performed for a problem that has more decision variables, uncertain parameters, and uncertain scenarios treated than analysts have been able to deal with until now.

Since the mixed-integer linear programming used in this paper is non-convex, it is important to review the key methods that are being used to find the optimal solution for non-convex problems, for which there may be many local optima solutions. Trying to get beyond these local solutions is the focus of various non-convex programming methods. Bellman developed dynamic programming,[9] which provides an efficient algorithm for the knapsack problem.[10] However, dynamic programming does not deal with the uncertain aspects of the problem. Another method is simulated annealing,[11] which initially searches broadly but, as the search goes on, focuses the search in a more

---

[6] A good standard text on linear programming is David Luenberger and Yinyu Ye, *Linear and Nonlinear Programming*, Third Edition, International Series in Operations Research and Management Science, Springer, 2008.

[7] Xinhui Zhang, Meenakshi Prajapati, and Eugene Peden, "A Stochastic Production Planning Model Under Uncertain Seasonal Demand and Market Growth," *International Journal of Production Research*, Vol. 49, No. 7, April 1, 2011, pp. 1957–1975.

[8] An FSW in a given run is defined as an outcome determined by the specific value for each uncertain parameter. In this paper, typically 10,000 FSWs are generated to form samples or to constitute a subset of the uncertainty space. In some literature, a FSW is called a scenario.

[9] Richard Bellman, *Dynamic Programming*, Princeton University Press, 1957.

[10] The "Knapsack problem" attempts to maximize the value of items chosen to include in the pack while satisfying a weight constraint and sometimes a cost constraint. In its general formulation, this is in the class of problems for which there is no known algorithm that can solve the general problem in polynomial (fast) time or verify that a solution is within a certain distance of optimality in polynomial time.

[11] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, Vol. 220, No. 4598, 1983, pp. 671–680.

concentrated area. Genetic algorithms[12] attempt to break away from local solutions through a process of mutation and evolution. Cutting plane algorithms[13] seek to exclude more and more of the feasible region by inserting "cuts" to eliminate parts of the region not expected to contain the overall optimal answer.

Typically, these approaches work well on some classes of problems and not so well on others. None of these algorithms can be guaranteed to converge in polynomial time[14] for the general non-convex problem or for the project portfolio problem addressed here. Moreover, the author is unable to find anything in the literature that indicates that these methods are particularly suited to solving the project portfolio problem under uncertainty that this paper addresses.

Analysts at RAND have a long and strong presence in the research and development (R&D) of methods and tools for planning under uncertainty. From the start, Dantzig and Bellman performed the groundbreaking work mentioned above at RAND. Davis has provided highlights of RAND's treatment of uncertainty in national security analysis over the last two decades.[15] Early RAND work developed optimal portfolios using DynaRANK[16] but assumed a good deal of linearity. More recent work, such as with the Portfolio Analysis Tool (PAT), has not sought optimization;[17] rather, it has sought to

---

[12]David E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*, Addison Wesley, 1989, p. 41.

[13]There are quite a few articles on the use of cutting planes to solve non-convex or non-smooth problems.

[14]According to the National Institute of Standards and Technology, polynomial time is the execution time of a computation, m(n), that is no more than a polynomial function of the problem size, n. More formally $m(n) = O(n^k)$ where k is a constant (Paul E. Black, "Polynomial Time," in *Dictionary of Algorithms and Data Structures* [online], Paul E. Black, ed., U.S. National Institute of Standards and Technology. August 13, 2004). Polynomial time is fast. However, it is not fast enough for solving the types of problems this paper addresses with brute force. That is to say, it is too slow to find the optimal solution by exhaustive search of all the possible solutions, when these possible solutions grow exponentially.

[15]Paul Davis, *Lessons from RAND's Work on Planning Under Uncertainty for National Security*, Santa Monica, Calif.: RAND Corporation, TR-1249-OSD, 2012.

[16]Richard Hillestad and Paul Davis, *Resource Allocation for the New Defense Strategy: The Dynarank Decision Support System*, Santa Monica, Calif.: RAND Corporation, MR-996-OSD, 1998.

[17]This paper, of course, discusses instances in which optimization is feasible and useful, instances in which relevant uncertainties and constraints can be expressed mathematically with uncertain parameters, and instances in which those parameters can be characterized with known distribution functions. Not all problems of uncertainty meet these criteria. In strategic planning, for example, policymakers frequently do not have well-defined and stable multi-attribute objective functions and may not yet recognize all the considerations at work and how they interact. Further, some of those considerations will change. In other cases, successful optimization requires understanding complex joint probability distributions because variables of the problem are correlated. Such interactions are frequently not understood even qualitatively, much less with the accuracy that would allow meaningful joint distributions to be specified. Much of RAND's work on exploratory analysis under uncertainty and on robust decisionmaking under deep uncertainty relates to cases for which optimization is not feasible, although optimization tools can be a very useful part of the toolbox for analysis. For details, see Davis,

emphasize iterative discussions with policymakers in a search for “balance” that reflects values and constraints that are not understood ahead of time but that can emerge as policymakers see consequences of different allocations in the right framework.[18] Other RAND work has also emphasized planning for flexibility, adaptiveness, and robustness. While this paper’s measure of a portfolio’s probability of success in the experimental space is a particular kind of robustness measure, others at RAND and elsewhere have used “regret.”[19] Other relevant RAND work deals with dimensionality problems by characterizing the scenario space, looking for ways to segment that space, finding and parameterizing illustrative cases for each segment to use in a “spanning set of test cases,” and screening the mathematically permissible portfolios to find those worthy of more detailed study by virtue of being attractive (near the efficient frontier) by at least one of the possible assumption sets used for testing.[20]

The approach presented here builds off the previous work, both inside RAND and out. But it takes a different approach. Instead of using mathematical equations and techniques for developing search algorithms, the approach discussed in this paper relies on using reasoning to determine why an algorithm so designed would have a reasonable chance of reaching the global optimal solution, as opposed to being trapped in one of the local optima solutions. Moreover, past applications of this approach indicate that algorithms based on reasoning can skip many intermediate search steps and, thus, find the global optimal solution quickly. Consequently, multiple search algorithms can be performed for the same problem within the same run time that would be required to perform a single algorithm used in other methods, thus enhancing the chance of finding the global optimal solution. This is especially so when these search algorithms are designed to cover each other’s shortcomings in possibly missing the global optimal solution.

Objectives

As noted, the approaches that have been most practical and that have found applications in the public or private sector often must severely limit the numbers of

---

2012; and Robert Lempert, David Groves, Steven Popper, and Steven Bankes, “A General, Analytic Method for Generating Robust Strategies and Narrative Scenarios,” *Management Science*, Vol. 52, No. 4, April 2006, pp. 514–552.

[18]Paul Davis and Paul Dreyer, *RAND’s Portfolio Analysis Tool (PAT): Theory, Methods, and Reference Manual*, Santa Monica, Calif.: RAND Corporation, TR-756-OSD, 2009.

[19]Robert Lempert, Steven Popper, and Steven Bankes, *Shaping the Next One Hundred Years: New Methods for Quantitative Long-Term Policy Analysis*, Santa Monica, Calif.: RAND Corporation, MR-1626-RPC, 2003.

[20]Paul Davis, Russell Shaver, Gaga Gvineria, and Justin Beck, *Finding Candidate Options for Investment Analysis: A Tool for Moving from Building Blocks to Composite Options (BCOT)*, Santa Monica, Calif.: RAND Corporation, TR-501-OSD, 2008.

decision variables, uncertain parameters, and scenarios treated.[21] Otherwise, the combinatorial possibilities would grow exponentially and prohibit even the most powerful computers (now or in the foreseeable future) from exhausting all the possibilities in finding the optimal solution. This paper introduces a new approach that greatly relaxes the restrictions and describes a number of search algorithms that have been demonstrated with practical applications. Also, this paper proposes establishing a common platform[22] so that solutions derived from different approaches and search algorithms can be objectively compared to determine which give the best solution.

Being able to consider many more facets of uncertainty is especially important because real-world decisions in business and government are rife with uncertainties, as in cases in which decisionmakers allocate resources without knowing confidently the consequences of the various options.

The approach and associated algorithms in this paper can be applicable to a number of resource allocation situations:

- Which internal R&D projects should a company select to fund to best meet an uncertain future, when it cannot afford to fund them all?
- Which new products should a company promote to produce the highest expected profit under an uncertain future, when its marketing budget and personnel are limited?
- What mix of products should a factory produce to generate the highest expected profits in the future, when its capacity and manpower are limited and the factors of production for products that the factory can produce are different?
- What mix of oil and gas wells and at what locations (drilling holes near producing wells or in new fields) should an exploration company invest in for highest expected profit?
- What numbers and types of trucks and aircraft should a transportation company acquire for long-term planning?
- How much operating and investment funding should be allocated to each department within a company, especially when it is facing a budget cut?

---

[21] For this context, an uncertain scenario or, simply, a scenario corresponds to a particular choice of settings for all the uncertain parameters of the problem. That is, it is a computational case. The choices of settings may be based on random draws from the parameters' probability distributions.

[22] In such a common platform, a number of problems with various types of decision variables, uncertain parameters, and scenarios will be designed as standard problems for testing. Better yet, these standard problems would have their solutions already known. Those who wish to test their approaches and search algorithms will apply them to the standard problems to find the optimal solutions or portfolio of selected projects. The platform is equipped to run these optimal solutions and compare, for example, which one has the highest probability of meeting all requirements within budget.

Magnitude of the Dimensionality Problem

It is worthwhile to briefly review just how serious dimensionality problems are. For example, let there be 75 projects to choose from. Because one can choose to include or reject each of the projects, there are 2^{75} (3.8×10^{22}) possible portfolios.[23] Assume that there are 10,000 scenarios to study and that studying each possible portfolio under each scenario with a linear programming model[24] takes 0.1 seconds. Using this approach to exhaustively study all possible combinations to find the optimal portfolio would take 1.2×10^{18} years. Even if one allowed for continued growth in computational power, this would not reduce this number meaningfully. With a smaller challenge, say only ten projects but with 10,000 scenarios, finding the solution by exhausting all combinations would still take 119 days.

To cope with such dimensionality, analysts use a variety of simplifying techniques. For example, they may consider only a small subset of the possible portfolios (perhaps ten), basing their choice on their insights about the varied challenges raised by different classes of scenarios and varied ways to cope with them. Alternatively, they might construct the ten portfolios to represent the range of opinions being expressed by stakeholders. Yet another approach is to use a screening approach to eliminate all but a relatively small number of portfolios.[25] If the purpose is to find the optimal portfolio, such an approach is severely limited, and it is highly unlikely that the optimal portfolio will be among the ten chosen for analysis in the first place. Even if the number of portfolios chosen for analysis were greatly expanded (for example, to 100), the number of possible projects considered would still be small. Thus, a new search strategy that examines many more possible portfolios would be of great interest.[26]

[23] If partial-funding options are included, then the number of cases is even greater, say 4^{75} if the choices are to fully fund, to fund at the 2/3 level, to fund at the 1/3 level, or not to fund.

[24] The linear programming models considered in this paper are mixed-integer linear programming models—models in which the objective function is linear, the constraints are linear, and some of the variables must be integers.

[25] As an example, Davis et al., 2008, describes an approach that generates all the composite options (portfolios) and then eliminates all except those that are relatively close to the efficient frontier in at least one perspective about the relative importance of objectives and at least one set of assumptions about the values of a few key parameters. The surviving options are then assessed in more detail using both objective and subjective considerations. With modest numbers of building-block programs (what would be called projects here)—say, 15—it is possible to do the screening with a desktop program. With larger but still relatively small numbers, the authors used a genetic algorithm method developed by Paul Dreyer at RAND.

[26] Because projects can be funded at different levels, the limitation is actually much more severe. Considering three different levels of funding, each project has four, not two, choices: not selected, selected at the first level, selected at the second level, and selected at the third level. Thus, the number of possible portfolios is 4^{75} rather than 2^{75}. In the case in which 100 possible portfolios can be analyzed, the limit of six (funded or not funded) projects will be further reduced to merely three projects, each of which has four possible choices.

The specific purpose of this paper is to describe a new approach to optimization under uncertainty that is aimed at finding the optimal solution to a problem by designing a number of search algorithms in a way that relaxes the limitations analysts have had to rely on in the past.[27]

Organization of This Document

The technical approach is discussed in Chapter Two. Chapter Three discusses two illustrative examples of the approach. These examples are drawn from the applications of the approach in the *Toward Affordable Systems* series of studies[28] sponsored by the Deputy Assistant Secretary of the Army (Cost and Economic Analysis), Office of Assistant Secretary of the Army (Financial Management and Comptroller). Chapter Four provides an overview of the approach and some suggestions for its further use. This paper also includes an appendix that illustrates how computer and network resources can be configured to execute the approach and its algorithms in alternative ways.

[27]Another purpose of this paper is to convert a provisional patent application for the approach and associated algorithms into an open publication for public use. The ideas and techniques presented in this paper may be used by anyone for any purpose with citation. The provisional patent application is entitled *Portfolio Optimization by Means of a Ranking and Competing Search*. The provisional patent application was filed with the United States Patent and Trademark Office on July 30, 2012, under the following description: Methods and products in accordance with various embodiments find and use an optimal portfolio under uncertainty.

[28]Brian Chow, Richard Silberglitt, and Scott Hiromoto, *Toward Affordable Systems: Portfolio Analysis and Management for Army Science and Technology Programs*, Santa Monica, Calif.: RAND Corporation, MG-761-A, 2009; Brian Chow, Richard Silberglitt, Scott Hiromoto, Caroline Reilly, and Christina Panis, *Toward Affordable Systems II: Portfolio Management for Army Science and Technology Programs Under Uncertainties*, Santa Monica, Calif.: RAND Corporation, MG-979-A, 2011; and Brian G. Chow, Richard Silberglitt, Caroline Reilly, Scott Hiromoto, and Christina Panis, *Toward Affordable Systems III: Portfolio Management for Army Engineering and Manufacturing Development Programs*, Santa Monica, Calif.: RAND Corporation, MG-1187-A, 2012.

(This page is intentionally left blank.)

Chapter Two: Technical Description

This chapter describes the approach. It starts by discussing the basic ideas that underlie it. It next provides a mathematical formulation of the portfolio optimization process. It then discusses the two specific ideas that underlie the approach: (1) the use of multiple algorithms to search for the optimal portfolio (OP) and (2) a new process to design algorithms. It then describes the steps for any given search scheme (SS), the specific steps for eight SSs, and flow charts illustrating the common SS approach and two variants. Next, it discusses OPs and products to produce them. Finally, it describes two methods for identifying requirements for new projects. The appendix describes how computer and network resources can be configured to implement the method.

Basic Ideas Underlying the New Approach

While the approach described here can be applied to many different types of problems, this paper uses investments in R&D projects to describe the ideas and the problem formulation as a way to make the approach more specific and understandable.[29] Let there be N proposed projects, not all of which can be funded. The user must decide which combination of projects to fund. If each project must be either fully funded or not funded, there are 2^N combinations or possible portfolios. The OP is defined in what follows as the portfolio of projects with the highest probability of meeting all constraints (C) for the scenarios of the search space. More properly, it is the portfolio that meets all the constraints in the largest fraction of the scenarios used to evaluate the portfolios.[30] Both N and C are integers, which are 1 or greater. The objective function for this uncertainty problem is to maximize the probability of meeting all constraints.

When there are uncertainties, the future can end up in many different states or scenarios. If a given portfolio of selected projects can meet all constraints for a state, that state is called a feasible state. If not, it is called an infeasible state. The probability is called the feasible percentage (FP), which is the number of feasible states over the sum of the feasible and infeasible states. Thus, the probability is the chance that a given portfolio can meet all constraints under uncertainties. When a portfolio has the highest FP, it is called the OP or optimal solution. A feasible state can also be called an

---

[29]Here, a project is defined as an endeavor that requires money to develop. If the project is successfully developed, it can be used to meet requirements, which include earning a good return on investment. Thus, a project can be, for example, an R&D program, a measure to reduce vehicular injuries, a product to market, or a product to be made by a machine, because the corresponding decision variable is whether to fund a program or measure, to promote a product, or to produce a product.

[30]If there are U uncertain input parameters or variables, the uncertainty space will have U dimensions, and the space of cases treated is formed by the combined random draws on the U uncertain input parameters; thus, it is a subset of the overall space.

acceptable state, because one would consider the state acceptable, given that it meets all the thresholds in the constraints.

The constraints here include requirements and budgets available. *Requirements* are the goals that the user wants the funded projects and their systems to fulfill. It can be a single requirement, such as a certain required rate of return on investment, or multiple requirements, such as the amount of renewable energies produced by the selected projects to reach a certain percent of the total energy consumption by 2025 *and* the amount of greenhouse gases reduced by a certain percent by 2020.

As for *budget constraints*, there is typically a constraint on the total remaining R&D (TRRD) budget[31] available to fund the selected projects.[32] There is also a total implementation (TIMP) budget to pay for the acquisition, operation, and maintenance of systems derived from those R&D projects that are funded and successfully completed. Once the projects are completed, one often does not need to acquire a number of systems developed under every successful project, because doing so may overspend and excessively meet all the requirements and would not be the lowest cost to meet all requirements. Rather, one can choose systems from some successful projects: implement (i.e., acquire, operate, and maintain) a number of copies for each of these systems so as to meet all requirements within budget constraints.

The goal, then, is to select any combination of any number of projects to form the OP for a given TRRD budget and a given TIMP budget. The OP will have the highest probability of meeting all given requirements and other constraints under a given set of uncertainties on some input parameters.[33]

As discussed above, the combinatorial possibilities are daunting. What follows describes two basic ideas for dealing with this issue. The first is the use of multiple algorithms to search for the OP or optimal solution. The second is to design individual search algorithms based on a new process to avoid some of the severe limitations that analysts have been forced to rely on in the current approaches. Both ideas are discussed below after first formulating the problem mathematically.

---

[31] A project can be already ongoing, in which case the spent R&D cost is sunk and excluded, and the future R&D budget is called the remaining R&D budget.

[32] While the approach and associated algorithms can be applied to uncertainty problems other than R&D projects, this paper uses R&D projects to more clearly describe the ideas.

[33] In a generalized version, the OP could be defined so as to give more or less weight to portfolios in terms of various other criteria, such as minimizing cost or maximum regret.

A Mathematical Formulation of the Portfolio Optimization Process

This section expresses the technique and its processing steps in a mathematical formulation. There are N projects for a user to choose from: no project, any one project, any two projects . . . or all projects. This is to say that one can choose any combination of any number of projects. The OP is defined as the particular combination of projects that has the highest probability or FP to meet all constraints under uncertainties as represented by a set of probability distribution functions.[34] Thus, the objective function is to maximize FP.

Let there be N binary decision variables (or simply variables), x_1, x_2... x_i… x_N, each of which can be either 0 or 1. The 0 means that the project is not selected for the OP, while the 1 means the project is selected. The string of N variables is represented by [*x*; N].

Each *x* variable has M attributes or coefficients. They are represented by [C; N, M], or C_{ij}, where i runs from 1 to N and j runs from 1 to M. Each project is characterized by two key attributes: the remaining R&D budget required for completion (RRD_i) (which corresponds to the TRRD budget above)[35] and the implementation budget required for acquiring the systems developed and for operating and maintaining them throughout their service life (IMP_i) (which corresponds to the TIMP budget above).[36] The attributes also include the contribution or value of a project to each of R requirements, V_{ij}, where i runs from 1 to N and j runs from 1 to R.[37] Moreover, among the M coefficients, there are Q coefficients or parameters with uncertainties, which are represented by [CU; N, Q], or CU_{ij} where i runs from 1 to N, and j runs from 1 to Q. Thus, [CU; N, Q] is a subset of [C; N, M]. The uncertainties for each CU_{ij} are governed by a probability distribution, $P(CU_{ij})$. There are Q × N such distributions, which are represented by [P(CU); N, Q].

[34] Currently, only binary or triplet distributions are used. However, it would be easy to generalize to accept other distributions. Thus, this paper assumes that all relevant uncertainties can be expressed mathematically via parameters characterized by uncertainty probability distributions. It also assumes that interactions among variables are properly accounted for by constraints. For example, a constraint can be used to disallow the non-sensible solution in which both mutual-exclusive cases (e.g., funding a full program and funding the same problem that is sized down) appear.

[35] This approach currently assumes that both the RRDj and the TRRD budget are known without any uncertainty. On the other hand, the IMP_i and the TIMP budget are allowed to be uncertain.

[36] For some applications, RRD_i and IMP_i can be combined and simplified into a single term, remaining life-cycle budget (RLC_i) (which corresponds to the TRLC budget discussed later in the text).

[37] V_{ij} are independent of each other. However, if dependency exists among a small number of projects, new projects can be created by combining these dependent projects to make the revised V_{ij} independent. In those cases, the third type of constraints described later in the text would also have to be used to make the V_{ij} independent.

An OP must meet three types of constraints.[38] The first type consists of ***budget constraints***—for example,

$$\sum_{i=1}^{N}(\mathrm{RRD_i})\times(x_i)\leq \text{TRRD},$$

where TRRD budget is the total remaining R&D budget available, and

$$\sum_{i=1}^{N}(\mathrm{IMP_i})\times(x_i)\leq \text{TIMP},$$

where TIMP budget is the total implementation budget available.

The second type consists of ***requirement constraints***—for example,

$$\sum_{i=1}^{N}(\mathrm{V_{ij}})\times(x_i)\geq \mathrm{TR_j};\quad j=1.....R,$$

where TR_j is the minimum total value that all selected projects must provide to meet requirement j.

The third type consists of ***other constraints,*** such as those among the projects. For example, let one assume that there are two projects, A and B, to develop the same system, except that project B will require a higher R&D budget but can result in a better performance and/or a cheaper system. Then, one may consider only funding at most one of the two projects—that is,

$$x_a + x_b \leq 1.$$

Mathematically, the objective is to maximize the FP, FP([*x*; N]; [CU; N, Q]), where [*x*; N] are the variables, and the maximization is to have the highest probability of meeting all constraints over the [CU; N, Q] uncertainty space.

Currently, there is no theoretical solution to this optimization problem. The idea is to find a solution or to identify a portfolio of projects for funding that will produce either the highest FP or one so close to it that it makes no practical difference in terms of the benefit or return on investment. Because there is no guarantee that any algorithm will lead to the optimal solution, establishing a common platform for checking the algorithms developed here in test problems with known solutions and comparing algorithms from different methods for the same problem is highly recommended. This implies that the

[38] Embodiments of the present invention also apply to a different formulation in which the model counts the penalty points and insists that the total does not exceed a certain number. The number of penalty points for violating each constraint can depend on the degree of violation.

platform needs to be supported by a library of test problems with known solutions so that different algorithms can be tested and compared. As the platform and its database accumulate more and more comparisons, there will be a better and better confidence about which algorithm works best in which types of problems.[39]

The approach is to randomly generate what the FSWs may turn out to be, according to the Q × N probability distributions. A typical number of FSWs is 10,000 and, sometimes, 100,000. Then, each generated FSW is represented by a set of randomly drawn values over the Q × N probability distributions. Each generated state now becomes a deterministic or certainty state, because all uncertain parameters are fixed at specific values. By examining one certainty state at a time, the uncertainties are first sidestepped. Without uncertainties, the OP for each certainty state can be found using other methods. A mixed-integer linear programming model is used for all certainty cases.[40] The idea is to deal with uncertainties by generating a large number of certainty states or cases. If all constraints can be met, the case is called a feasible case. For each feasible case, a mixed-integer linear programing model is used to examine which of the N projects are selected to constitute the feasible portfolio. In sum, projects that appear frequently in these feasible cases are the most desirable ones to form the OP.

Use of Multiple Algorithms to Search for the OP

As part of the approach, a number of search schemes (SSs)—described below—have been designed. By itself, each SS has its own logic in having a reasonable chance of reaching the OP or a near-optimal portfolio whose FP is very close to the optimal FP. In the problems to which this new approach and associated algorithms have been applied, the statistical fluctuation around a solution is typically 1 percentage point or less. Thus, this paper assumes that it makes little difference to the user whether this method recommends a portfolio of selected projects that has a 95-percent FP of meeting all requirements under given budgets or a different portfolio that yields 94 percent. The user can use either portfolio and have practically the same confidence level in meeting all the requirements or goals.

To better understand the multiple algorithms approach, one can use a mountain-climbing analogy. Mathematically speaking, a linear programming problem under

---

[39]The types of problems have multiple dimensions here. First, they refer to different kinds of problems in resource allocations, such as R&D projects and production planning issues. Second, even within a problem, such as R&D projects, the nature of uncertainties and the numbers of decision variables and constraints may affect how well a particular algorithm performs. Third, ultimately, as the future is full of uncertainties, an algorithm may have applications far beyond resource allocations. The common platform should be aimed to test and compare algorithms over many types of problems.

[40]Other methods or models, such as nonlinear programming, can also be used for the certainty states.

certainty is a convex problem, which, as discussed in Chapter One, was solved by Dantzig with the simplex algorithm and, later, by others with different methods. By analogy, this convex problem is like a mountain with a single peak. However, many of the problems this paper is interested in addressing would require some of the decision variables to be integers (for example, the decision to either fully fund a project or not fund it at all). Such a mixed-integer linear programming model would make the problem non-convex. Compared to a convex problem, a non-convex problem is like a mountain with multiple peaks, with the highest peak being the global optimum or optimal solution.

A simple example about how to reach the mountaintop can illustrate the basic idea of using multiple search algorithms to find the OP. A real mountain, like a real-world non-convex problem, has multiple peaks. Let one call all of them *local peaks* or *local optima*, except for the top of the mountain, which is called the *global peak* or *global optimum*. A mountain climber would not want to mistake a local peak for a global peak and stop climbing once he or she got there, but this may be the case because the visibility is too poor for the climber to tell whether there is a higher peak far away. The same is true in searching for the solution of a non-convex problem.

For a mountain that has not yet been explored (an analogy for a new problem), the idea is to have a group of several expert climbers with different climbing styles and different routes chosen (good routes based on different logics about why the routes can reach the global peak rather than the local ones) to start climbing from one place at the base of the mountain. Although all the climbers start from the same place, each of them uses a different set of important climbing instructions to help them get beyond a local peak and continue climbing to reach the global peak. Moreover, several other groups start from different places of the mountain. Some of these expert climbers can have the same climbing style or instructions as those starting from the same place, while some have styles and instructions that are not used by the other climbers.

Once all the climbers have reached their own highest points, the locations and heights of their highest points are compared. Like the above mountain-climbing example, the basic idea proposed here is to use multiple SSs[41] (to be described below) to deal with a resource allocation problem under uncertainty (a non-convex problem) as ways to find their respective working optimal portfolios (WOPs) and corresponding FPs.[42] Then, the portfolio with the highest FP is the OP. Thus, all WOPs found by different search schemes are *local* OPs, except the one with highest FP, which is called the *global* OP or simply the OP.

[41]The use of multiple SSs is defined very broadly. In the extreme case, one can treat all the genetic programming algorithms as one SS to be used with other existing algorithms or those here in forming multiple SSs as recommended here.

[42]WOPs are used to reserve the term OP for the one WOP that yields the highest FP.

In designing these SSs or algorithms, it is important that every SS have a transparent reason behind it so that one can argue why it has a reasonable chance of reaching the global OP instead of stopping at a local OP. It is also important to create counter-arguments about why the global OP could have been missed. Only then can other algorithms be designed to include the missing factors left behind by the original algorithm. This approach relies on this type of reasoning and devil's advocates to ultimately develop a set of complementary SSs that are most likely to include at least one SS capable of finding the global OP or one close to it.

The experience in using the SSs reported in this paper has been that the WOPs or local OPs that they individually reach are often similar in both project composition and FP. In one case with 26 possible projects and three different SSs, the local OPs are composed of either 18 or 19 projects, of which the 18 projects are common among all three; thus, the difference lies in whether there is a 19th project and which project it is. Moreover, their FPs are 93.33 percent, 93.63 percent, and 93.64 percent, with these small differences most likely the result of statistical fluctuations arising from the use of 10,000 scenarios to represent the much larger number of possible uncertain scenarios.

The similarity in project composition and FP in the local OPs does not mean that these three local OPs are actually at the global OP or very near it. It is possible that all different climbing strategies could have missed a critical factor or process, and that, thus, the global peak has not been reached. This is true for the approach presented here, as well as for all existing approaches for problems under uncertainty. However, the approach presented here also relies on another strategy—using transparent arguments in developing SSs—which is discussed later in Figures 2.3 and 2.4. For now, it is sufficient to say that these arguments can lead to a new SS that is complementary, just as the one in Figure 2.3 leads to the one in Figure 2.4. Using the two algorithms together would make it more likely to reach the global peak or optimal solution, because each one covers the weakness of the other in missing the global peak.

An SS is specified by four elements:

1. The rules that guide the search.
2. A second set of rules either to confirm that the first set of rules under Element 1 has resulted in the OP or to allow one to find a portfolio with a higher FP, which is now the OP unless the search continues.
3. The objective function used in the mixed-integer linear programming model (under certainty) for determining the frequency of appearance of each of the N projects in the feasible cases. Recall that the objective function for the

uncertainty problem is to maximize the probability in meeting all constraints. As discussed in the next section, this approach decomposes an uncertainty problem into typically 10,000 certainty problems. For these certainty problems, another objective function would be needed to define and find the optimal solution for each certainty problem.

4. The objective function used in the mixed-integer linear programming model (under certainty) for determining the FP of the OP. The objective function is typically the same as that used in Element 3. Other objective functions can also be used instead without affecting the FP, because the feasibility is determined by meeting all the constraints and is independent of whichever objective function that is mentioned in this paper is used.

As for determining which and how many SSs to use in a new problem, one can consider several factors. First, is the new problem similar to the problems in which SSs have been applied? If so, one can select those SSs that have worked the best in the past for the new problem.

Second, for the new problem, the user can run a larger number of SSs relative to those combinations of TRRD budget and TIMP budget the user is most interested in selecting. Based on these results, the user can then choose a smaller number of SSs to determine the OP and FP at other combinations of the two budgets. Even after the choice is narrowed to a particular budget combination, the user can still check the FP at that budget combination again by using a larger number of SSs.

Third, if the user has been using another method in the past to determine the OP, it is both revealing and important to compare the FP from that OP with the FP from using the SSs in this document. The establishment of a common platform for objectively determining which algorithm or algorithms is best for a given type of problems is highly recommended.

Fourth, whenever certain local OPs are close in their FPs, the user can increase the number of runs from 10,000 to, say, 100,000 to gain higher confidence in deciding which one indeed has the highest FP and, thus, should be selected as the global OP.[43]

Fifth, it is often difficult to know which algorithm best suits the problem at hand. An approach with multiple SSs—especially mutually reinforcing ones that plug each other's logical holes in possibly missing the global optimum under various situations—offers a much better chance that at least one SS will find the OP.

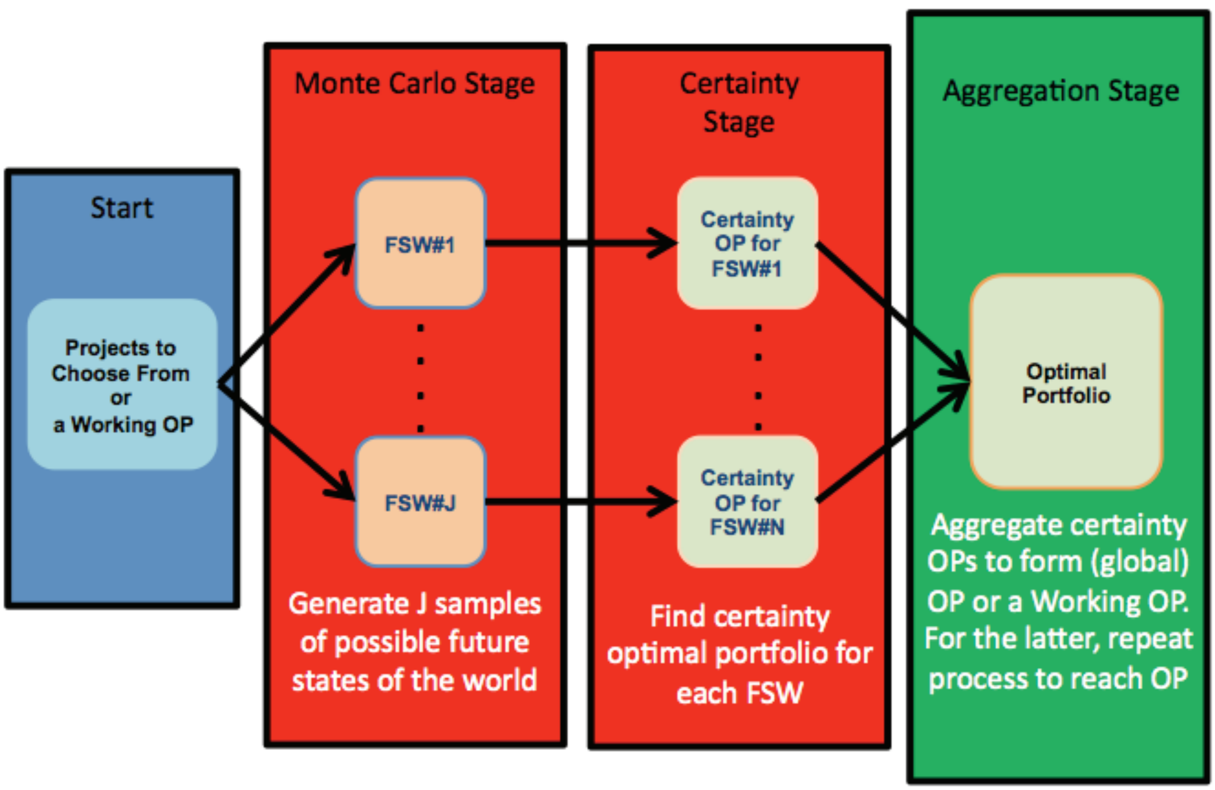
[43] Another way is to run multiple sets of 10,000 or 100,000 runs with different seeds for the random draws.

Sixth, because the many runs for a given budget, different budgets, and different search algorithms can all be run independently and simultaneously, this approach can take advantage of the current advances in parallel computing and the rapid decline of cost of on-demand computer power.

A New Process to Design Algorithms

In addition to multiple searches (described in the previous section), the second basic idea of this approach is to relax the dimensionality constraints analysts have had to contend with until now. Figure 2.1 shows the schematic approach. It starts with a given set of projects to choose from (on the far left). The goal is to find the OP or optimal solution, which is a portfolio composed of a group of selected projects. Such a selection is optimal because the OP will give the highest likelihood or FP to meet all requirements within a given budget for project development (the TRRD budget) and another budget for acquiring, operating, and maintaining the systems from those projects to meet requirements (the TIMP budget).

Figure 2.1. Schematic Approach for Optimization



During the *Monte Carlo stage* (see Figure 2.1), a number of possible FSWs (typically 10,000) are generated. For any given FSW, every uncertain parameter becomes certain. One can immediately take advantage of the linear programming model or other linear or nonlinear models to find the OP for each FSW.

During the *certainty stage* (see Figure 2.1), one can see the different selections or compositions of projects in these (10,000) certainty OPs under various (10,000) FSWs. This is a critical step because it involves using the solvable deterministic or certainty linear programming model to find an OP for each of the 10,000 (certainty) FSWs, avoiding the impossible job of singling out the OP by running through trillions of possible portfolios by brute force. By taking advantage of the solvable mixed-integer linear programming model, one can learn a lot about the uncertainty space and its impact on the global OP than one can by using current approaches.

The last stage—the *aggregate stage* (see Figure 2.1)—is to find the global OP or simply the OP for the original problem under uncertainty. All the SSs described in the rest of this paper are designed to take advantage of the knowledge gained from these (10,000) certainty OPs to identify the global OP. Some algorithms can find the global OP from these certainty OPs in one step. Others would go through some WOP and repeat and refine the process to arrive at the global OP.

The certainty search at the certainty stage and the uncertainty search at the aggregate stage constitute the foundation of this tandem certainty–uncertainty search approach. When multiple such searches join forces, this is called *portfolio optimization by means of multiple tandem certainty-uncertainty searches*.

It should be noted that the three limitations in the optimization process have been greatly relaxed. First, this method and algorithms have been successfully applied to 75 and 183 projects instead of ten projects or packages of projects, and even larger numbers of projects are possible. Second, instead of as few as three FSWs, 10,000 FSWs have been routinely used in the applications of this paper’s approach and associated algorithms. Third, instead of a few uncertain parameters, 75 uncertain parameters have also been used in prior demonstrations. Two applications of this approach are discussed in Chapter Three.

Steps for a Given Search Scheme

While every SS has its own logic in how it can reach the global OP, some steps are similar. Many SSs go through various WOPs before reaching the final WOPs for these SSs. Among the final WOPs from the selected SSs, the final WOP that has the highest FP is the global OP. These steps are based on two ideas: ranking for project inclusion in the WOPs and having final WOPs corresponding to multiple searches to compete for being selected as the final OP, which has the highest FP.

The general structure of an SS consists of four steps:

- Step 1 is to develop a core group of projects for a given TRRD budget. These core projects are to be included in the WOP for any given TIMP budget. The objective function can be any one mentioned in this section for core group determination, project frequency listing, replacements, or FP calculation.
- Step 2 is to fill the WOP, either by adding to the core projects from Step 1 or by starting from zero and completely ignoring Step 1. The additions are based on including the highest frequency projects appearing in a number of runs (typically 10,000). The objective function can be any one mentioned below. Moreover, it can be the same as, or different from, the one used in the other steps, such as Step 1.
- Step 3 is to perform single, double, triple, etc., replacements on the WOP obtained under Step 2. Moreover, Step 3 can also be skipped in an SS, especially when the user is convinced that this step does not result in an OP that yields an appreciably higher FP.
- Step 4 is to make a final determination of the model-recommended OP's FP. One may want to have a higher number of runs (for example, 100,000 runs) for this determination. As explained before, the choice of the objective function is irrelevant for the purpose of determining the FP.

Eight different SSs that have been developed are described below; they follow the four steps described above.

Search Scheme 1 (SS-1)
In various embodiments, this SS is run for a given TRRD budget and a given TIMP budget at a time. The purpose is to find the OP, which is a selection of projects from the original pool of N possible projects for inclusion, and the corresponding FP at this given combination of budgets. The steps are as follows:

1. Let Q be the number of uncertain input parameters. An example of an uncertain input parameter is that the unit cost of a (future) system derived from a project is uncertain. Randomly select a value for each uncertain input parameter according to its statistics or probability distribution. Once the values are selected, the problem becomes a certainty case on which a mixed-integer linear programming

model[44] with the objective function of minimizing the TRRD budget is run to see whether all constraints are met. If all constraints are met, the selected projects in the successful portfolio are recorded.

2. Repeat Step 1 a number of times, which is typically equal to 10,000 runs. Any run in which all constraints are met is called a feasible run. The number of feasible runs over the total number of runs multiplied times 100 provides an FP. All these runs result in a table that lists the frequency of appearance of each of the N possible projects in all these feasible runs.
3. All N projects are ranked based on their frequencies of appearance, with the one with the highest frequency ranked at the top.[45] SS-1 includes a number of projects (SP) from the top down and tallies their corresponding remaining R&D budgets until the total remaining R&D budget is fully committed.[46] Then, the FP for this WOP is determined, typically by another 10,000 runs.[47]
4. Since SP out of N possible projects are now in the WOP, the number of projects excluded from the WOP is N-SP, which is called RP. Each RP project is substituted for each SP project, and 10,000 runs are made for each substitution.[48] The SS-1 keeps the WOP with the highest FP. This process is repeated with any pair of replacements at a time among the RP projects, with any three replacements, etc., until there is no improvement in the FP or the improvement is less than a predetermined threshold amount. The end of this process yields a WOP with the highest FP for SS-1.

Search Scheme 2 (SS-2)

The SS-2 has the following steps, many of which are the same as SS-1:

1. This step is the same as in SS-1, ***except the mixed-integer linear programming model is run with an objective function to minimize the TRRD***

---

[44]While all demonstrations are performed with a linear programming model, the same SS works in nonlinear programming models or user-designed models for the purpose of finding the OP.

[45]For example, let there be 26 possible projects to choose from, and 8,000 out of the 10,000 runs are feasible runs. Among these feasible runs, let one assume that Project 5 appears most often, or 7,559 times, for a highest frequency of 7,559; Project 24, a frequency of 7,112; Project 2, a frequency of 6,543, etc.

[46]If there is insufficient budget to include the next highest frequency project in the portfolio, the method will add a lower-ranked project where the budget can still accommodate it. The user can also modify this search scheme for use in the SSs that involve replacements as follows: Whenever a project already in the WOP is replaced by a project outside the WOP that has a smaller R&D cost, the leftover R&D cost will be used to include the highest frequency project or projects outside the WOP that fit the leftover R&D budget.

[47]Because the purpose of these runs is to count the feasible runs, it makes no difference in the FP if the objective function is to minimize the TRRD budget, the TIMP budget, or the sum of the two (that is, the TRLC) budget.

[48]Therefore, there are SP times RP sets of runs, and each set is to have 10,000 runs. However, in some applications, some of the RP can be ruled out without making runs, because they can be determined to have no chance of making the FP higher.

budget but with no constraint on the TIMP budget. Further, with an unconstrained TIMP budget, the uncertainty in implementation cost is irrelevant in determining the OP. At this point, all other uncertainties, such as the possibility of project failure, are ignored. The idea behind this SS is to first ask, if all uncertainties turn out to be most favorable to each possible project, what projects would be selected for the OP? Such selected projects will form the core of the OP. Then, other projects are added to this core to deal with situations in which some of the core projects fail or contribute less favorably because of bad outcomes from uncertainties.

2. Same as SS-1, ***except the objective function is to minimize the TIMP budget, and each of the 10,000 runs is made with the same given TRRD budget and TIMP budget.*** The difference with SS-1 lies in always keeping the core projects in SS-2. The TRRD budget not committed to the core projects will be used to add highest frequency projects not already in the core until the TRRD budget is fully committed.
3. Same as SS-1.
4. Same as SS-1.

Search Scheme 3 (SS-3)

SS-3 is the same as SS-2, ***except the project frequencies in Step 2 are determined with an objective function of maximizing the total value contributions from the selected projects to the requirements***.

Search Scheme 4 (SS-4)

SS-4 is the same as SS-2, ***except in both Steps 1 and 2, the OP is determined with an objective function of minimizing total remaining life-cycle (TRLC) budget.***[49]

Search Scheme 5 (SS-5)

SS-5 is the same as SS-2, ***except in both Steps 1 and 2, the OP is determined with an objective function of maximizing the number of projects selected in the OP*****.**

Search Scheme 6 (SS-6)

SS-6 is the same as SS-2, ***except in both Steps 1 and 2, the OP is determined with an objective function of minimizing the TRRD budget.***

Search Scheme 7 (SS-7)

SS-7 is the same as SS-2, ***except in Step 1, all the projects in the nearest neighboring OP with a lower TRRD budget are used as the core projects for this OP, and in Step 2, the objective function is to also minimize the TRRD budget.***

[49]The TRLC budget is the sum of the TRRD budget and the TIMP budget.

Search Scheme 8 (SS-8)
The SS-8 is a scheme that has been used often in the applications of this new approach:

1. Same as SS-1.
2. Same as SS-1.
3. This SS relies on a logic that is similar to what a manager often uses to sensibly select and form a team to perform a task without using a model and mathematics. The manager would select the most key person first, then the next person best complementing the first person's skills and temperament, and then the third to complement the first two, and so on until budget runs out. In applying this SS, only the highest frequency or ranked project (P1) in the first set of 10,000 runs is kept in the WOP. For each mixed-integer linear programming run in the second set of 10,000 runs, P1 has already been included from the start. Then, the highest ranked project (other than P1) is added to the WOP. For the third set of 10,000 runs, P1 and P2 are always selected in the first place, and the highest ranked project (P3) is added to WOP. The process is repeated until the total remaining R&D budget is fully committed.
4. Same as SS-1, ***except that the first project in the above WOP is to be replaced by another project in the new WOP, and Step 3 is repeated.*** Moreover, whenever the list of projects included in the new WOP under this step is the same as any of the WOPs obtained thus far in Step 3, the process will stop, because further project additions will not yield a different WOP—they will simply repeat and yield one of the previous WOPs in Step 3. In essence, this process picks one project at a time from the project list as the first project. In theory, there can be as many WOPs as there are projects to choose from, but in practice the numbers can be less because some WOPs may be the same.

Other Search Schemes
There are two ways to generate additional SSs. The first way is to derive them from the many different combinations of objective functions for Steps 1–3 used in the above eight SSs. The second way is that the user can use objective functions different from those discussed here.

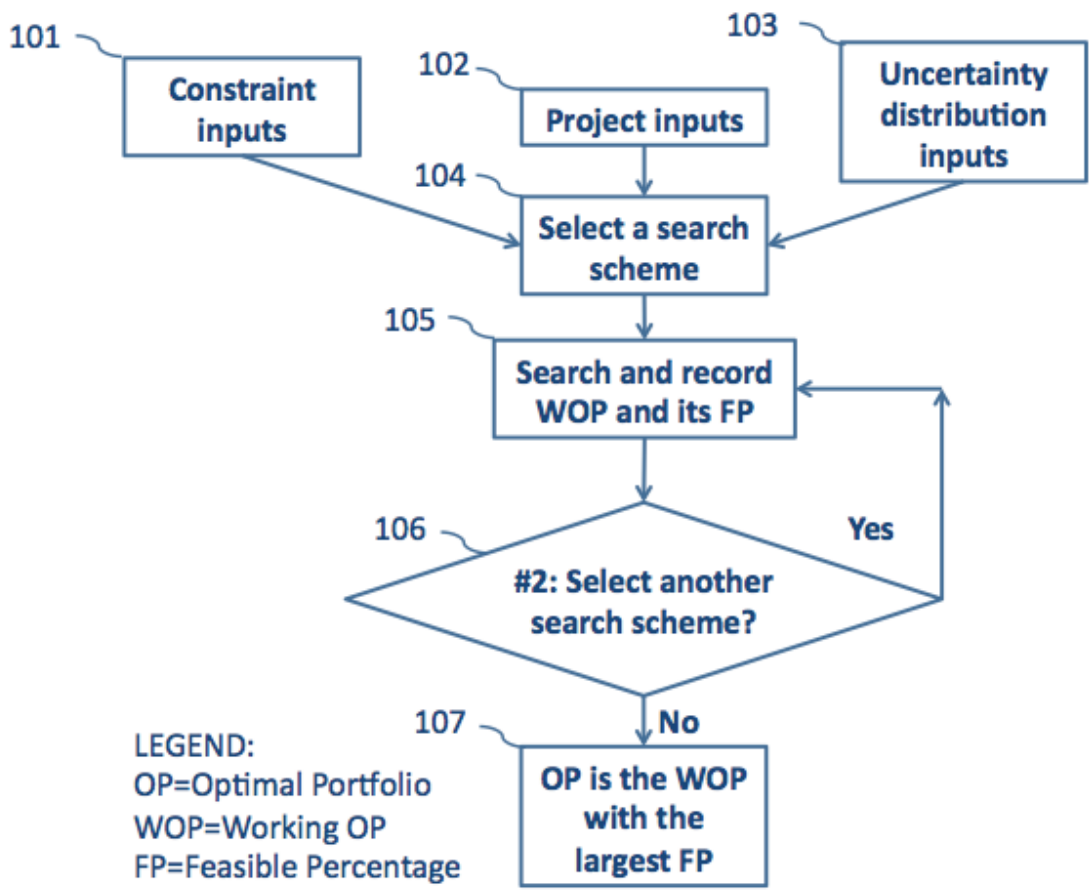
Using Flow Charts to Show the SS Process

In this section, the SS process discussed above is represented by a series of flow charts, starting with a flow chart showing the common steps applicable to all SSs before turning to flow charts showing the specific steps for SS-8.

Flow Chart Showing Common Steps Applicable to All SSs

As noted above, there are different SSs to find these feasible portfolios and select projects to form the OP. Figure 2.2 shows the common steps applicable to all SSs.

Figure 2.2. Flow Chart for Searching for the OP



The flow chart steps are ordered as shown by the numbers, running here from 101 to 107. Below, what happens in each of the boxes of the flow chart is discussed.

- **Box 101:** All three types of constraints are expressed mathematically, as shown above in this section.
- **Box 102:** The project attributes or coefficients, [C; N, M], are inputs to the model.
- **Box 103:** The Q × N probability distributions for the uncertain coefficients, [P(CU); N, Q], are also inputs.
- **Box 104:** An SS is chosen from among the eight discussed earlier.
- **Box 105:** A given SS will result in a WOP. The project composition in WOP and the FP are recorded for the SS used. The details of how an SS identifies the

WOP are illustrated with two examples in the next section, whose flow charts are shown in Figures 2.3 and 2.4.

- **Box 106:** One can search the OP with multiple SSs.
- **Box 107:** Each SS results in one WOP. Once the user no longer wants to try another SS, the OP is the WOP with the highest FP.

Flow Charts Showing the Full Process for Finding the OP for Two Variants of SS-8

The above figure (Figure 2.2) and discussion are for a generic SS. Two SSs are now discussed to show the full process in finding the OP. Both are based on SS-8 and were discussed earlier. The first uses the first three steps in the discussion above and is called SS-8 without replacement (SS8-NR). The second uses all four steps discussed earlier and is called SS-8 with single replacement (SS8-SR).

SS-8 Without Replacement (SS8-NR)

Figure 2.3 shows the details of Box 105 in Figure 2.2. SS8-NR is first specified in all four of the aforementioned elements discussed earlier. Once it is selected for Box 104 in Figure 2.2, the next step is Box 201.

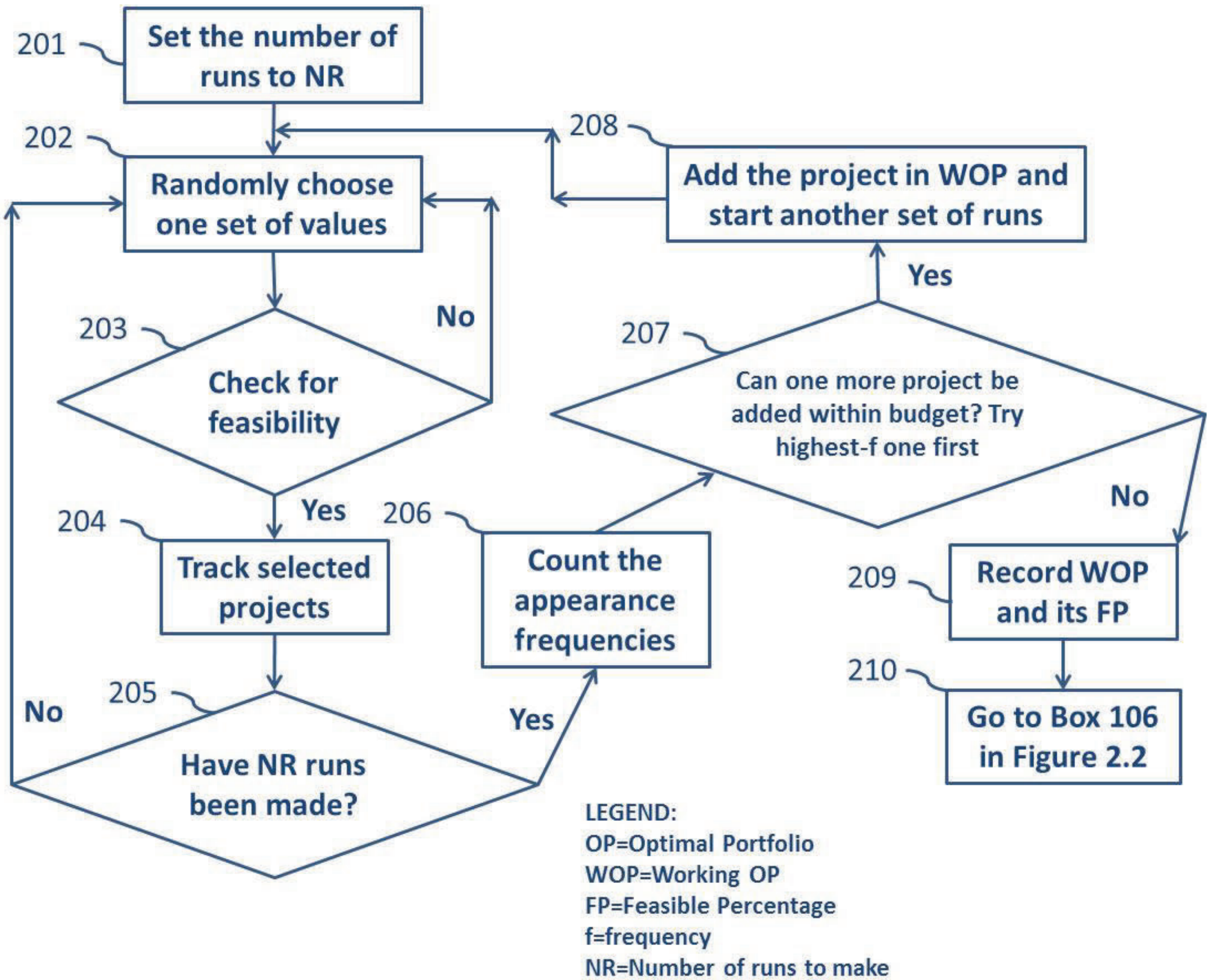
- **Box 201:** The number of runs is selected here. To be specific, 10,000 runs are used.
- **Box 202:** For each run, random draws are performed on Q × N probability distributions for the uncertain coefficients, [P(CU); N, Q]. Thus, each set of random draws has Q × N values corresponding to Q × N uncertain coefficients or parameters.[50] After the draws, the case becomes a certainty case.
- **Box 203:** For the certainty case in Box 202, a mixed-integer linear programming model is run to see whether all constraints can be met. If it does, it is called a feasible case.
- **Box 204:** A record is kept about which of N original projects are selected in each feasible case.
- **Box 205:** There are 10,000 runs based on 10,000 sets of random draws.
- **Box 206:** Based on the accounting done in Box 204, the frequency of appearances of each of N projects in all the feasible cases is tallied.
- **Box 207:** For the first time arriving at this box, the project with the highest frequency is selected as the first member of the final WOP for this SS. For a subsequent arrival at this box, some project(s) will have already been selected for the final WOP. After summing up the RRDs for these selected projects and

[50]Embodiments of the method and algorithms in this paper have been applied to a case in which, in addition to Q × N uncertain coefficients, the TIMP budget is uncertain. In such a case, the set of random draws merely needs to add one more random draw to cover the TIMP budget.

subtracting these committed RRDs from the TRRD budget, one gets the remaining budget for additional projects. One then determines whether the new project with the highest frequency (not compared with the frequencies of the projects already in the WOP before entering Box 207) has an RRD budget less than the remaining TRRD budget. If the remaining budget allows, add this highest-frequency project to the WOP. If this project has a budget that is too high, try the project with the second-highest frequency of appearance, etc.

- **Box 208:** Add the new project to the portfolio. The idea is to first select the highest-frequency project from the N projects for the final WOP. Then, ask the question: What is the next project that is best to go with the first chosen project to meet all constraints? Once the first two members of the WOP are selected, one asks a similar question: What is the next project that is best to go with the first two chosen projects to meet all constraints? One continues to do this until the total RRD budget is fully committed.
- **Box 209:** Once no more projects can be added to the portfolio, the portfolio becomes the WOP or the portfolio that is identified through SS8-NR to have the highest FP. One records which projects are selected for this WOP and their FPs. This box completes the description of details for Box 105.
- **Box 210:** Since the process with Box 105 is now complete, the next step is to go to Box 106 in Figure 2.2.

Figure 2.3. Flow Chart for SS-8 Without Replacement (SS8-NR)



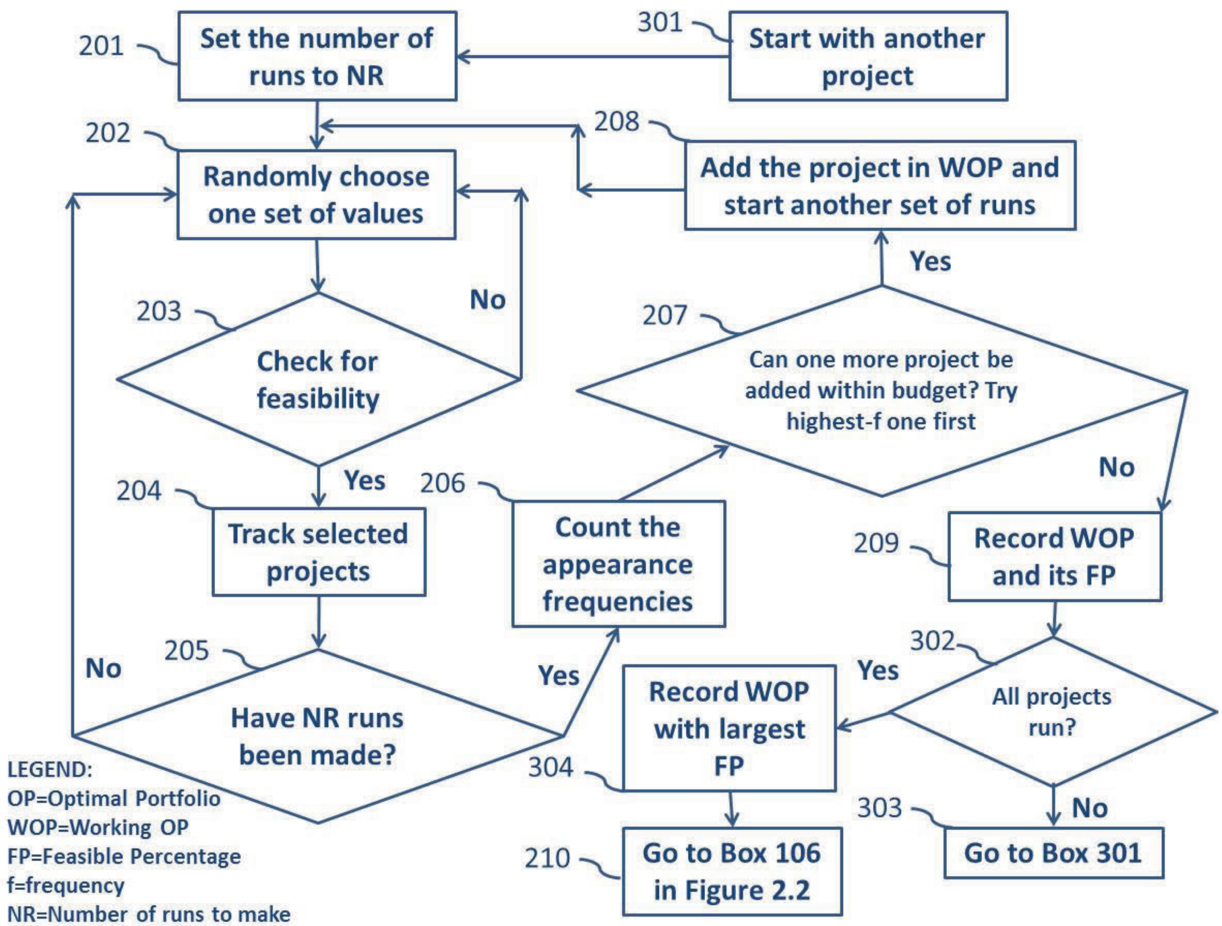
SS-8 with Single Replacement (SS8-SR)

This SS complements SS8-NR. SS8-NR starts by including the project that is most frequently selected in the feasible cases, while SS8-SR explores the possibility of the existence of a "second-best" project for a number of scenarios. While such a project is not the first choice for many feasible states and, thus, does not shows up as the highest-frequency project, it could well be the second choice for many feasible states. A possibility, though not high, could exist that this second-choice project might team up with other projects to form a WOP that has a higher FP than the one from SS8-NR. Clearly, the WOPs from both SS8-NR and SS8-SR should go to Box 107 in Figure 2.2 to see which has a higher FP and, thus, whose WOP will become the OP.

Figure 2.4 shows the details in Box 105 in Figure 2.2 with SS8-SR.

- **Box 301:** SS8-SR can only be performed after SS8-NR is performed. SS8-SR was described in Step 4 under the subsection on Search Scheme 8. The process is to replace the first project selected by SS8-NR with other projects on the list. If there are N projects and SS8-NR has already selected one to be the first project, SS8-SR selects each of the other projects to be the first project and will provide as many as (N–1) WOPs to compare with the WOP from SS8-NR. The WOP with the highest FP will be the optimal solution.
- **Boxes 201–209:** Once the first member of the WOP is chosen, one will run through the same process as that for SSR-NR to find the best project to go with the first project or member. Then, one will run through the same process again to find the best project to go with the first two members, etc., until the total RRD budget is fully committed.
- **Box 302:** This box ensures that all (N–1) single replacements are run.
- **Box 303:** If there are still more single replacements to run, this box routes to box 301 for another run.
- **Box 304:** Out of (N–1) WOPs, this box selects the one with the largest FP and sends the project composition for that WOP and its FP to Box 210.
- **Box 210:** This sends the content to Box 106 in Figure 2.2.

Figure 2.4. Flow Chart for SS-8 with Single Replacement (SS8-SR)



If both SS8-NR and SS8-SR are used, one can then simply select the WOP with the larger FP, as indicated in Box 107 of Figure 2.2. Typically, SS8-NR would produce the larger FP. However, by using SS8-SR, one can see whether a different first project can be combined with certain projects to produce a larger FP.

Further, if one goes through single replacement, double replacements,[51] triple replacements, etc., it becomes more and more likely that the true OP or a portfolio with a very similar FP will be found. Again, typically, SS8-NR gives the highest FP. Further, even if SS8-SR gives a higher FP than that of SS8-NR, it is typically not higher by any appreciable amount. Thus, one can consider only using SS8-NR for initial runs to draw tentative conclusions. However, one should perform SS8-SR or even SS8 with double replacements to confirm that the WOP from the SS8-NR has the highest FP or to replace the WOP with a higher one.

[51]For a double-replacement scheme, one starts a WOP by choosing any other pair of N projects to replace its first two members.

A Computer Program Product to Determine the Optimal Portfolio

Using multiple search schemes is encouraged to generate multiple SS-specific OPs. The OP for the problem is the one with the highest FP. Any combination of SSs can be used to determine the OP, which consists of projects selected for investment for a given TRRD budget and a given TIMP budget. The FP of the OP can also be calculated. A computer program product for use on a computer system has been developed by the author. It consists of a mixed-integer linear programming model and a simulation to perform the steps described above and is discussed in the appendix.

Method for New Projects and Method for Existing Projects

In addition to introducing a new approach and its associated search algorithms (as described above), two new methods have been developed. These two methods are intended to be useful for those planners who are interested in not only finding the optimization solution to meet a set of given requirements but also in evaluating whether it is more cost-effective to use the projects in the current pool to meet some requirements or to design new projects to do so. One method is to identify those requirements that are potentially more cost-effective for new projects, rather than existing projects, to meet. The other is to find a sweet spot among existing projects where one finds the biggest bang for the buck. From a different perspective, this method suggests that a planner may not want to spend too much money on the existing projects—that the money saved may be better spent on designing and funding new projects. Each method is described below.

A Method to Identify Requirements for New Projects

A simple example is used to help explain the purpose of the method. Let one assume, for example, that the Army (in this particular case) has a requirement that is easier and cheaper to meet by striking enemy positions from the air, but that the existing projects for the Army to choose from in meeting this requirement all pertain to developing ground-based systems. If the Army were to use these ground-based weapon systems to do the job, the job could still be done, but doing so would cost much more than using the airborne systems that are not in the existing pool but that are much more suitable for the job. This paper describes a method to help identify this ill-matched requirement and other such requirements that are much less obvious, so the user (the Army in this case) would seek the development of new weapons, such as helicopter, to do the job at a much lower cost. This method is described below.

The user has a pool of existing projects and is asked to select a subset of these projects that has the highest probability of meeting multiple requirements with a given budget. The user also wonders whether some of these requirements are more cost-effectively met with new projects instead of existing projects. This method is to first see how well the existing projects, if they could all be successfully completed, would be able to meet the multiple requirements.

The first step is to tally the values (i.e., project contributions of individual requirements) of the existing projects for each requirement. One would want the total value to well exceed the requirement level; a safety margin is needed because, in reality, some funded projects will not be completed successfully.

Building on this fact, the method asks the following for each requirement: If the highest-value project contributing to that requirement were to fail, could the rest of the projects still meet that requirement? If not, the requirement level is lowered to match the total value of the rest of the projects. Clearly, this lower level is easier to meet because it allows for the failure of any one project, even the highest-value project.

The user can use a single SS or a combination of SSs as described above to find the OPs that meet the above set of lower requirements and their FPs. If the FPs for the ranges of TRRD budget and TIMP budget the user is considering are still too low, the user can further reduce the requirement levels by taking away the two highest-value projects for each requirement. If the requirement levels are lower, there would be a better chance of meeting them, leading to a higher FP. If the FPs are still too low to satisfy the user, the user can take away the three highest-value projects, and so on to make the FP higher.

By learning of the trade-off between requirement levels and the FPs and identifying the types and levels of requirements that the existing projects find hard and expensive to meet, the user can tailor new projects more cost-effectively to meet these specific requirements.

Once these new projects are designed, they can be added to the pool of existing projects for an analysis, following the process as described in the steps above. From this analysis, the user can confirm whether funding new projects instead of additional already existing projects will indeed save money or increase FP.

Sweet Spot for the Highest Return on Investment

A sweet spot is defined as a combination of the TRRD budget and the TRLC budget where it strikes a balance between performance (FP) and affordability (budget). This paper describes a method that calculates the ratio of FP over TRLC budget at various combinations of TRRD and TRLC budgets. This ratio is highest at the sweet spot. The sweet spot suggests that the TRRD budget and the TRLC budget there will produce the highest return on the investment. It also suggests that if the user wishes to pay more money to attain an FP higher than that at the sweet spot, the user should consider designing new projects to meet requirements that the current pool of existing projects finds hard or expensive to meet. The sweet spot also identifies those requirements that are potentially cheaper and easier for new projects, as opposed to those in the current pool, to meet. Once the new projects are designed, they can be added to the existing projects for portfolio analysis and selection, following the steps suggested above.

(This page is intentionally left blank.)

Chapter Three: Applications of the Approach in Past Studies

As noted above, applications of this approach were developed in the *Toward Affordable Systems* series of studies that were sponsored by the Deputy Assistant Secretary of the Army (Cost and Economic Analysis), Office of Assistant Secretary of the Army (Financial Management and Comptroller).[52] In this chapter, how the approach was applied in two of those cases from the past studies is discussed.

First Case Selected from Past Studies

The first case is an application that appeared in *Toward Affordable Systems II*.[53] The purpose of this application is to decide how much money should be spent on the science and technology (S&T) projects that are already existing or ongoing. The Army would want to spend the money cost-effectively to meet its capability gaps in 11 categories. The Army also wants to consider an alternative in meeting some of the capability gaps. The alternative is to design and fund new projects to meet some of these requirements if this option turns out to be more cost-effective than using existing or ongoing projects. The model first suggests how much money to spend or how large a budget should be. Then, for this budget or any other budget that the Army considers, the model recommends which projects to select to have the highest chance of meeting all requirements within that chosen budget.

In this case, any solution or portfolio must meet 13 constraints. In addition to a constraint on the total remaining science and technology (S&T) budget and another on the total remaining life-cycle (TRLC) budget,[54] there are 11 requirement constraints corresponding to the categories of force operating capability gaps that the S&T projects are to develop weapon systems to fulfill. There are 75 Army Technology Objective (ATO) projects, which are the Army's highest-priority S&T efforts. Thus, there are 75 binary decision variables corresponding to whether these individual 75 projects are selected or rejected for the OP. There are 75 independent uncertain parameters, each

---

[52] Brian Chow, Richard Silberglitt, and Scott Hiromoto, *Toward Affordable Systems: Portfolio Analysis and Management for Army Science and Technology Programs*, Santa Monica, Calif.: RAND Corporation, MG-761-A, 2009; Brian Chow, Richard Silberglitt, Scott Hiromoto, Caroline Reilly, and Christina Panis, *Toward Affordable Systems II: Portfolio Management for Army Science and Technology Programs Under Uncertainties*, Santa Monica, Calif.: RAND Corporation, MG-979-A, 2011; and Brian G. Chow, Richard Silberglitt, Caroline Reilly, Scott Hiromoto, and Christina Panis, *Toward Affordable Systems III: Portfolio Management for Army Engineering and Manufacturing Development Programs*, Santa Monica, Calif.: RAND Corporation, MG-1187-A, 2012.

[53] Chow et al., 2011; and Brian Chow, Richard Silberglitt, Caroline Reilly, Scott Hiromoto, and Christina Panis, *Choosing Defense Project Portfolios: A New Tool for Making Optimal Choices in a World of Constraint and Uncertainty*, Santa Monica, Calif.: RAND Corporation, RB-9678-A, 2012.

[54] The total remaining S&T budget plus the total implementation budget equals the total remaining life-cycle budget.

of which signifies that each project has a binary uncertainty: a 90 percent chance of successful completion of their S&T efforts and a 10 percent chance of failure. For those projects that are selected for S&T funding and successfully complete their S&T phase, the planner can then select some of these successful projects to have the needed copies of weapon systems built and fielded to meet Army's mission requirements.

The first step is to study whether the projects in the current pool are really suitable to be used to meet all the capability gaps and whether some of the gaps could be more cost-effectively met with new projects whose designs are tailored to meet those gaps that are hard to meet with current projects. The method described in the section "A Method to Identify Requirements for New Projects" in the previous chapter is used. In using that method, it was found that there is only a 16-percent chance of meeting all constraints under the given uncertainties of project success or failure, even if all 75 projects are selected for funding. It was further found that the chance would be drastically increased to close to 100 percent if the levels in 4 of the 11 categories were reduced.[55] For the analysis below, the requirements have been reduced to such levels, considering that requirements above these levels are more cost-effectively filled by new, as opposed to existing and ongoing, projects.

SS-1, as described in Chapter Two, is used to find the optimal solution. The uncertain parameters are the above 75 success/failure uncertain parameters. For example, to find the FP at the sweet spot in Figure 3.1,[56] one sets the total remaining S&T budget at $2 billion and the TRLC budget at $35 billion.[57] SS-1 yields an FP of 91 percent and provides a list of selected projects among the 75 projects that constitute the OP. Similarly, the FPs for all other budget data points can be determined.

The approach and associated algorithms are written in in General Algebraic Modeling System (GAMS), a tool for solving mixed-integer linear programming optimization problems. The IBM ILOG CPLEX Optimization Studio (often informally referred to simply as CPLEX) was used. In the simulation to find the OP for a given budget, the CPLEX Solver was typically used tens of thousands of times according to SS-1.[58] For each data point, typically 10,000 runs are made, and each run takes about 0.1 second to find the optimal solution for that FSW. Another 10,000 runs are made to find the FP for that data point. In sum, it takes about half an hour for each data point (for example, the sweet spot is a data point). Further, when the number of runs is increased to

[55]The capability gaps that need to be met in category 4 are reduced by 25 percent. That means that only 75 percent of the gaps now need to be met. Similarly, gaps in category 6 are reduced by 42 percent; category 10, by 60 percent; and category 11, by 50 percent. (Chow et al., 2011, p. 49.)

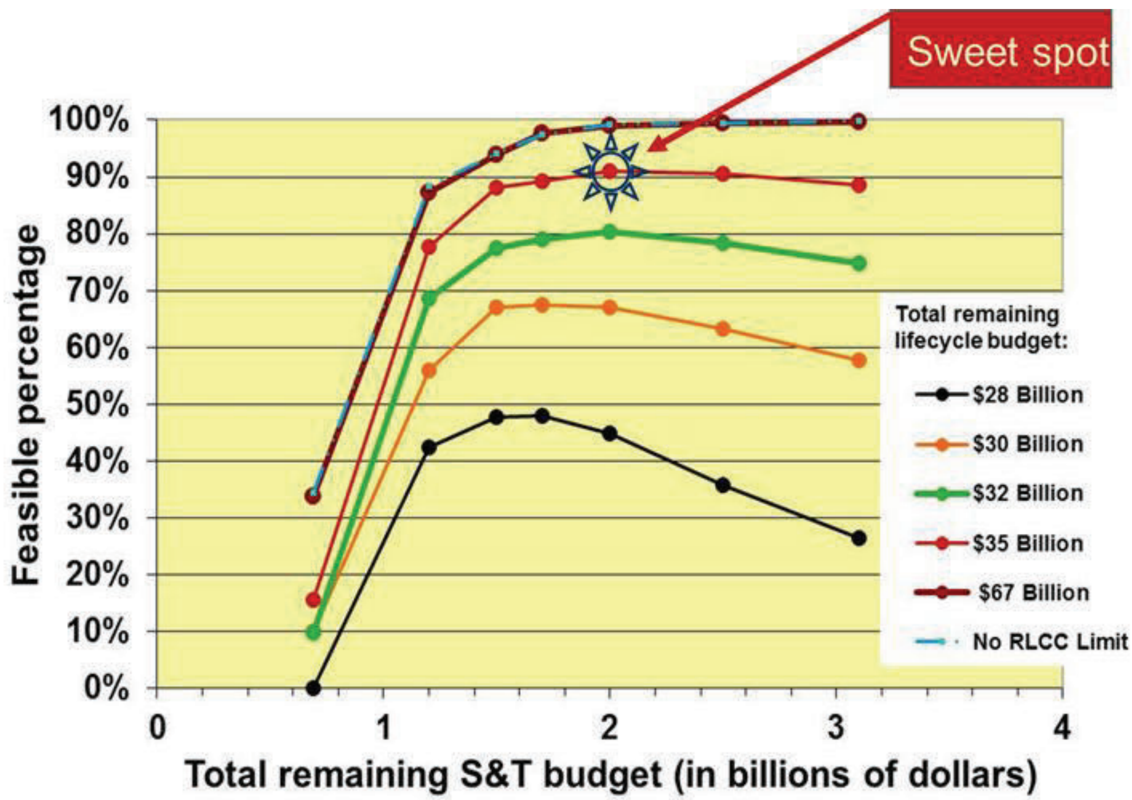
[56]Chow et al., 2011, p. 52.

[57]This also means that the total implementation budget is $33 billion.

[58]See the appendix for further details.

100,000, the FPs from the 10,000 runs would be typically within 1 percent deviation from those of the 100,000 runs.

Figure 3.1. The Sweet Spot Between Performance and Affordability in the TRLC Budget and S&T Budget



One can draw several observations from Figure 3.1:

- It is most cost-effective to spend $2 billion in total remaining S&T budget and $35 billion in TRLC budget (the starred point on the chart). This would allow the feasible percentage to reach 91 percent.
- The model specifically selects 53 projects to form the OP, which has the highest probability (of 91 percent) in meeting all requirements within budgets.
- There is a “knee” at around $1.5 billion in the total remaining S&T budget. Its location is insensitive to the TRLC budget. Below the knee, the confidence of meeting requirements drops drastically. Planners should make this result known to Army senior leadership so the Army can try hard to keep budget above that level. However, if the budget cannot be raised, the Army, Congress, and the public should know the consequence of that.

- If the budget has to be cut, the Army can find the least impactful places to cut by studying Figure 3.1 vertically and horizontally. At the sweet spot (and looking vertically), each billion dollars in TRLC budget yields 2.6 percentage points in feasibility. If the Army wants to spend $32 billion more—a TRLC budget of $67 billion—the extra money only buys 0.6 percentage points per billion dollars, far less than the 2.6. Thus, the extra money would not be well spent. If the Army originally planned to spend $67 billion and now thought of cutting, the model would say that a cut to $35 billion would only reduce the FP from 99 percent to 91 percent and would be a cost-effective cut. Further, if the Army were to plan to further cut the budget from $35 billion to $32 billion, the model would say that the $3 billion additional cut would drop the percentage points in feasibility by 3.5 per billion dollars, higher than 2.6. Any further drop below $32 billion would make the decline in FP even less cost-effective.
- If the Army were contemplating spending $2.5 billion in total remaining S&T budget, the model would suggest (now looking at Figure 3.1 horizontally) postponing $0.5 billion until implementation, whether there is a budget crunch or not, because the FP would not decrease.[59] The Army could delay $300 million in spending from the S&T budget to the implementation budget without affecting the likelihood requirements much (less than 2 percentage points).

The selected projects in the OP are compared to a selection based on the typical ratio of total expected value[60] over the remaining S&T cost of the project. A selection method based on this ratio would have started the selection of projects from the right of Figure 3.2 until the total remaining S&T budget is used up.[61] In contrast, this approach rejects projects that have a similar ratio to many other selected projects because, as good as these rejected projects are in filling capability gaps, there are already even better projects selected to do the job. Similarly, this approach selects projects for the OP when many other projects with this low ratio have been rejected, because, as poor as they are, others are even worse, and they are the best of the worse to fill the gaps.

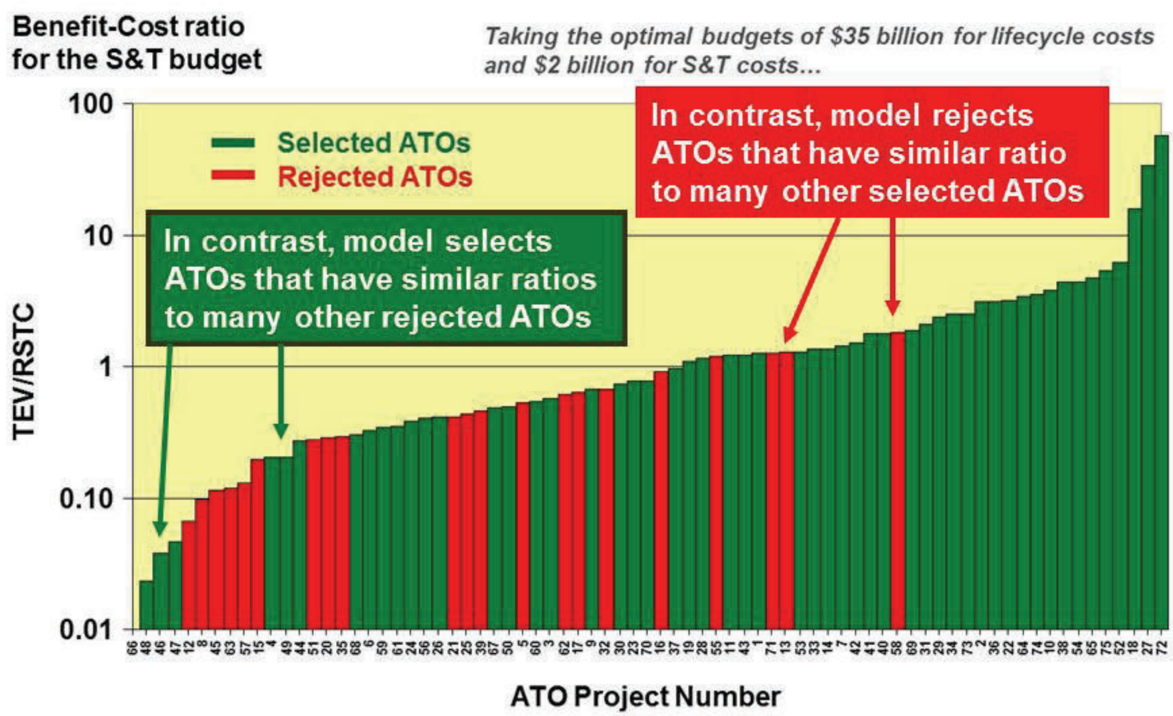
---

[59]The model actually indicates a slight increase from 90.5 percent to 90.9 percent, but clearly the increase cannot be counted on because it is within statistical fluctuation for a problem under uncertainty.

[60]The total expected value of a project is the sum of its contributions for filling the capability gaps in the 11 categories.

[61]Chow et al., 2011, p. 58.

Figure 3.2. Project Ordering According to the Ratio of Total Expected Value Over the Remaining S&T Cost



Second Case Selected from Past Studies

The second case is an application that appeared in *Toward Affordable Systems III*.[62] Instead of studying S&T projects, this study focuses on projects that are already in the engineering and manufacturing development phase or close to it. There are 26 such R&D projects. In addition, there are 157 projects, whose R&D efforts have been completed and which are ready to have weapon systems built and fielded to help meet capability gaps. The Army can choose among these 183 projects to meet requirements.

In this case, any solution or portfolio must meet 24 constraints. In addition to a constraint on the TRRD budget and another on the TRLC budget,[63] there are 22 requirement constraints corresponding to the categories of capability gaps in force protection that the R&D projects are to develop weapon systems to fulfill. There are 183 projects or decision variables for the Army to choose from to form the OP. There are 27 uncertain parameters. The cost in procuring a system derived from each project may end up higher than expected and is represented by a binary distribution of equal probability: 50 percent for no cost overrun and 50 percent for the cost to be doubled. Since there are 26 projects, the number of such uncertain cost parameters is also 26.

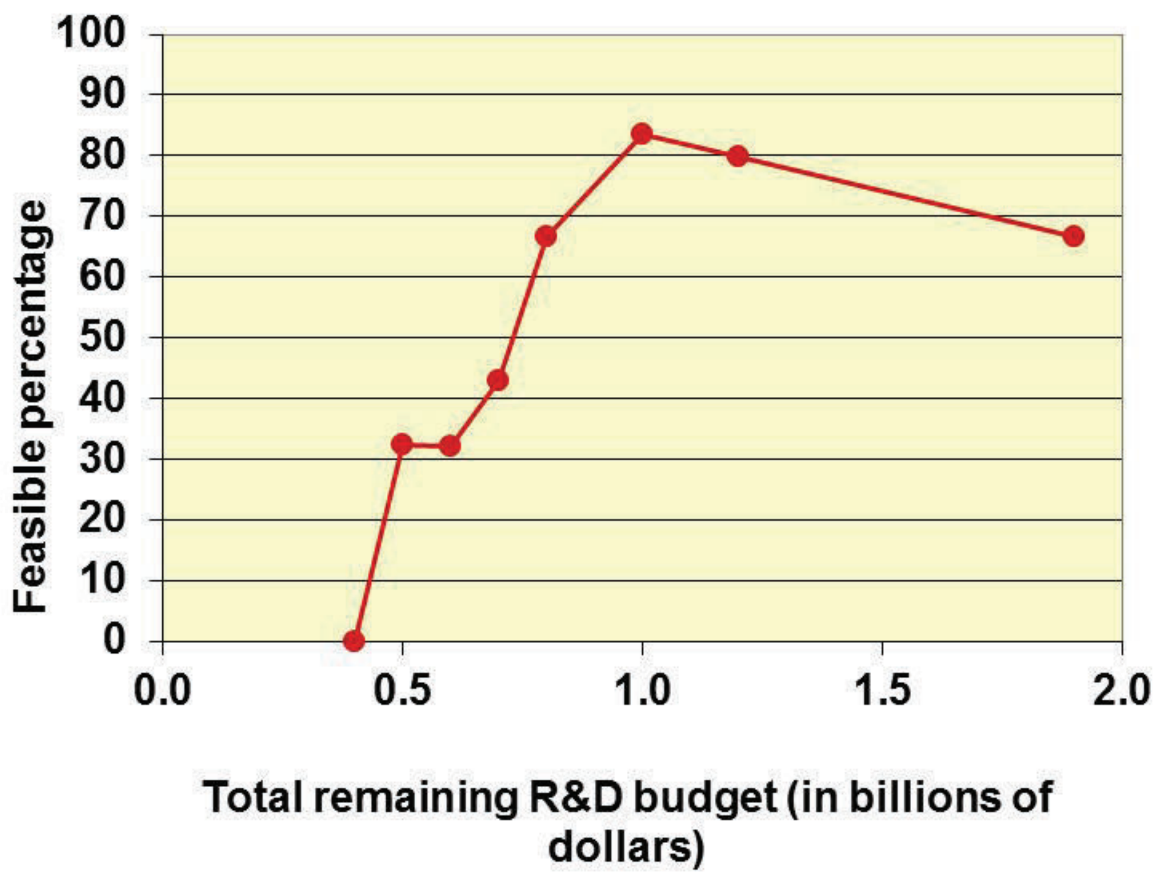
---

[62] Chow et al., 2012.
[63] The total remaining S&T budget plus the TIMP budget equals the TRLC budget.

There is one more uncertain parameter corresponding to the TRLC budget, which has an equal probability of being $20 billion, $22.5 billion, or $25 billion.

SS8-NR, as described in Figure 2.3 in the previous chapter, is used to find the optimal solution (Figure 3.3).[64] The search scheme calls for running 10,000 runs 26 times instead of once, making the time for determining the FP and OP at each data point about two hours, when higher efficiency in making multiple sets of runs is taken into account. As shown in Figure 3.3, this approach recommends spending $1 billion on the existing pool of projects to yield a feasible percentage of 83 percent in meeting all the requirements whenever there are uncertainties in both the costs of the weapons systems and the future budget of acquiring and fielding the weapon systems.

Figure 3.3. Likelihood of Meeting Threshold Requirements, with TRLC Budget Equally Likely to Be $20 Billion, $22.5 Billion, or $25 Billion



In *Toward Affordable Systems III*, the results from this approach are compared to those of two typical models: the benefit/cost ratio model and the certainty model.[65] The

[64]Chow et al., 2012, p. 30.
[65]Chow et al., 2012, p. 32 and p. 42.

benefit/cost ratio model is the same as what was described in Case 1 earlier in this chapter, where the ratio is the total expected value over the remaining R&D cost of the project. The certainty model is the deterministic mixed-integer linear programming model used here but without the uncertainty algorithm to refine the search for optimality.

Table 3.1 shows the results of the comparison between the model described here and the other two models. In comparing the model described here with the benefit/cost ratio model at equal budgets, this model yields the same FP in one case, a moderately higher FP in one case, and substantially higher FPs in seven cases. In comparing this model with the certainty model at equal budgets, this model yields the same feasible percentage in one case, moderately higher FPs in five cases, and substantially higher in three cases. Further, this model can be used to indicate where the other two models can be used without significantly lower chance to meet requirements with the same budget.

Table 3.1. Comparison of This Model with Other Models

Budget		Feasible Percentage		
TRRD $Billion	TRLC $Billion	This Model	Certainty Model	Benefit/Cost Ratio Model
With Full Consideration of RTBF Systems				
0.7	25	100%	100%	28%
1.0	20	68%	56%	0%
1.2	20	40%	0%	0%
With Approximate Consideration of RTBF Systems				
1.4	7	100%	74%	0%
1.3	6	95%	25%	0%
1.4	6	93%	25%	0%
1.6	6	86%	77%	0%
1.8	6	68%	60%	61%
1.9	6	63%	56%	63%
R&D=Research & Development				
TRRD=Total Remaining R&D				
TRLC=Total Remaining Lifecycle, which includes TRRD				
RTBF=Ready-To-Be-Fielded				

(This page is intentionally left blank.)

Chapter Four: Overview of the Approach and Suggestions for Expanding Its Use

The combinatorial possible solutions of problems under uncertainty grow exponentially with the number of decision variables, uncertain parameters, and uncertain scenarios. Even the most powerful computers cannot perform exhaustive searches in a reasonable amount of time when they are faced with an exponential growth of possible solutions. Yet the popular stochastic programming approaches use exhaustive searches to take advantage of pioneered works of Dantzig and Beale in decomposing the uncertainty problem into solvable, deterministic (certainty) linear programming problems.

The approach described here proposes to sample the uncertainty space with typically 10,000 FSWs. It takes advantage of the fact that once the FSW is specified, the problem becomes solvable. The approach is to use transparent reasoning, as opposed to mathematical formulas, to design search schemes or algorithms to find the global optimum and not get trapped at one of the local optima. This approach relies on arguments from devil's advocates to uncover the shortcomings of an algorithm in terms of why under certain situations it will not lead to the global optimum. Once the weaknesses of a given algorithm are identified, the original algorithm hopefully can be modified to remove the shortcomings or another algorithm can be designed to plug the reasoning hole of the original algorithm.

Experience with this approach has been good. However, if the shortcomings in these algorithms cannot be eliminated, this approach would have to rely on the simplicity of nonmathematical reasoning so that many analysts or even "crowd wisdom" can be used to devise completely different algorithms to do the job. Because all approaches, including this one, face the risk of potentially missing the global optimum, this approach based on reasoning can open a new way for drawing in talents from the nonmathematical world to devise search schemes to tackle this very difficult task of optimization under uncertainty.

These logically derived search algorithms are easy to understand, and the implementation amounts to creating a flow chart and using no complicating mathematics or formulas; as such, the approach allows for a wider adoption by analysts and organizations that possess different skill sets. The two illustrative search schemes (SS8-NR and SS8-SR) draw on common sense and commonly practiced ideas. SS8-NR is based on the idea of how to create a project team. Suppose a project sponsor has some "use-it-or-lose-it" money at the end of a fiscal year. While a project must be issued now within a broad study area, the sponsor will assign specific tasks over the course of the one-year project, but which tasks those will be is unclear. The company's policy and the sponsor's requirement, however, are such that the project leader must specify the team members at the project's start, after which it will not be possible to change them. Under such circumstances, it makes sense to draw up a list of tasks that the sponsor *may* ask the project team to do and to start by then selecting the person (by analogy, the first project selected) suitable for the largest number of potential task combinations that can be anticipated. The next step would be to find the person (by analogy, the second project selected) to best complement the first in technical and managerial skills so that the pair is suitable for the largest number of possible task combinations. Similarly, the third person (by analogy, the third project selected) would be found to best complement the first two, and so on.

The second search scheme (SS8-SR)[66] accounts for the possibility that if a different person were selected as the first person, the skill sets and personal chemistry to complement this different person might lead to a team composition different from the one based on SS8-NR.

Mathematically, these approaches are very convenient. Let there be N persons to choose from in forming the project team. Instead of looking at 2^N—or easily millions or trillions of possible team compositions—the two search schemes together would generate only N possible project teams (by analogy, any project could be the first, but thereafter the choices would be determined by looking at results for the uncertain scenarios used), thus enormously reducing the complexity of the search.

This type of reasoning in designing and using complementary algorithms can give analysts a way—which may be more transparent than mathematics or which can at the least supplement it—to uncover and mitigate logical lapses. Also, analysts using this approach may feel more confident that, even if these algorithms do not find the global optimum, the local optimum they find should be near the global one, because the logic of these algorithms is used often and has worked well in the analysts' other daily activities.

Each of the algorithms developed in this paper takes minutes or hours to find the optimal solution, even for uncertainty problems involving substantially more decision variables (75 used versus typically ten in other methods), uncertain parameters (75 used versus typically a few in other methods), and uncertain scenarios (10,000 used versus a few—or, alternatively, more than 10,000 but then restricting variables and/or parameters to around ten in other methods) than other methods would have allowed.

---

[66] The terms *without replacement* and *with replacement* refer to whether search is conducted with a unique choice of the first team member (the first version) or whether the search is conducted with each of the possible people as the first team member.

Thus, the relatively shorter run time offers the possibility to perform several complementary algorithms for the same problem, thus enhancing the chance of finding the global optimum.

The objective function is chosen to be the highest chance of meeting the requirements within a budget. This can be a way to introduce confidence levels in dealing with uncertainty. Then again, if analysts prefer to use the more conventional objective functions—such as minimization of expected cost or regret—this method can be modified to use those objective functions with other changes in the formulation and search algorithms, which may be akin to something as simple as moving from a simple average to weighted sums.

This method is also suitable for parallel computing, because these 10,000 runs for each data point, the runs of different data points, and the runs for different algorithms can all be performed independently and simultaneously. The current advances in parallel computing and the rapid decline in cost of on-demand computer power favor this multiple search approach.

This paper proposes a common platform so that solutions derived from different approaches and search algorithms can be objectively compared to determine which gives the best solution. This implies that the platform is supported by a library of test problems with known solutions so that different algorithms can be tested and compared. As the platform and its database accumulate more and more comparisons, there will be better confidence about which algorithm works the best for which types of problems.

This paper also proposes to extend the applications of the approach and associated algorithms in several dimensions:

- Apply it to different types of problems beyond the current focus on project portfolio problems under uncertainty. One may start the expansion with other resource allocation problems, such as production planning.
- Use other objective functions for the uncertainty problem, such as the minimization of expected total cost or regret.
- Program the multiple search algorithms for parallel computing to shorten the run time.

Finally, the paper proposes a systematic examination of approaches and their search algorithms, with the goal of combining their individual strengths and mitigating their weaknesses to give users ways to better perform optimization under uncertainty.

Because uncertainties are inherent in input data and the future, better ways to factor uncertainties into consideration are critically important for any type of decisionmaking.

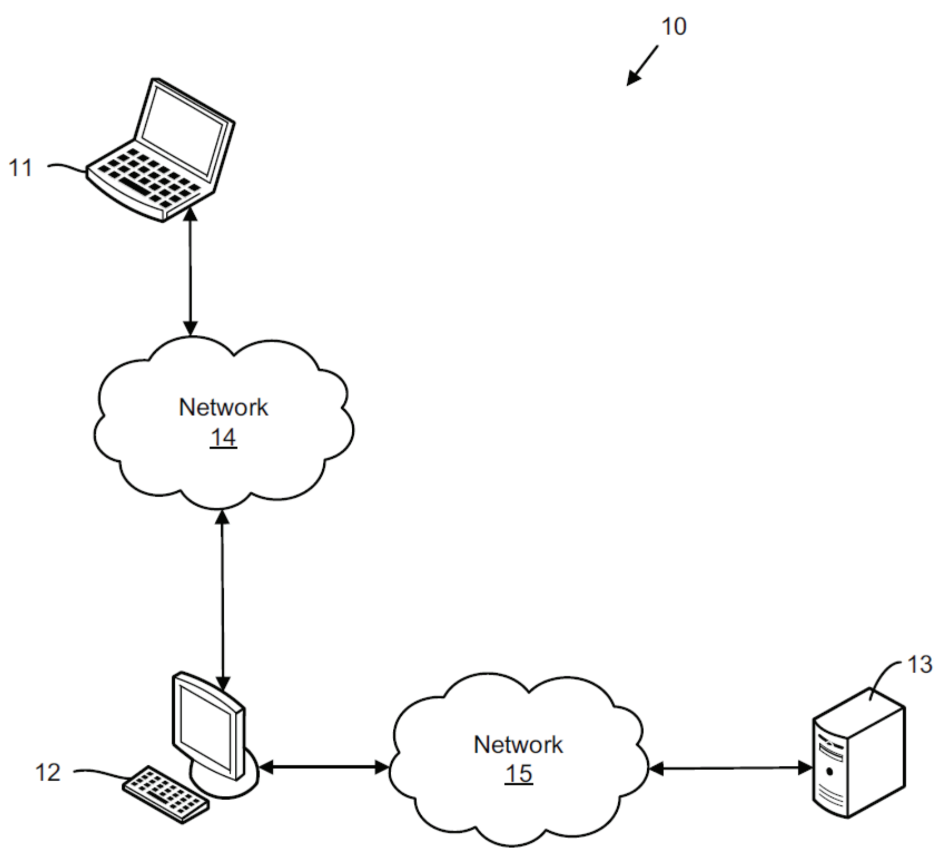
Appendix: Embodiments of Computer Resources Used in Portfolio Optimization by Means of Multiple Certainty-Uncertainty Searches

How would one perform the certainty-uncertainty search methods for the portfolio optimization approach described in the main text and highlighted in Figure 2.1 and in the flows charts shown in Figures 2.2–2.4? This appendix describes illustrative and alternative configurations of computer and network resources to implement the approach and its associated algorithms.

A System in Accordance with an Embodiment for Performing the Methods

Figure A.1 shows the computer resources needed to perform the methods in the portfolio optimization approach described in the main text. As with the flow charts shown in Figures 2.2–2.4 in the main text, the components are numbered for ease of discussion.

Figure A.1. A System in Accordance with an Embodiment for Performing the Methods



The system (10) includes a computer (11), another computer (12), a server (13), a network (14), and another network (15). Computer 11 is shown as a portable computer, but it could be any type of computer, such as a desktop computer, a server, or the like. Computer 12 is shown as a desktop computer, but it too could be any type of computer, such as a portable computer, a server, or the like. Server 13 is shown as a server computer, but it could be any type of computer, such as a desktop computer, a portable computer, or the like.

As shown in the Figure A.1, computer 11 communicates with computer 12 through network 14. Computer 12 communicates with server 13 through network 15. Network 14 may include a local area network (LAN), a wide area network (WAN), and/or combinations of LANs and/or WANs. Network 15 may include a local area network (LAN), a wide area network (WAN), and/or combinations of LANs and/or WANs. Networks 14 and 15 may be the same network and may be the Internet.

In various embodiments for this system (10), all the input data are recorded in an Excel file. Sometimes, this file may be prepared by the office of a project sponsor at computer 11 and then sent through email from computer 11 to computer 12 over network 14. Sometimes, this file is internally generated by a study team at computer 12. In either case, this input file is first stored on computer 12. Because the software used prefers comma-separated values (CSV) files in a specific layout, the data are formatted before they are uploaded from computer 12 over network 15 to server 13, which may be, for example, a server that has eight CPU cores at 2.3GHz with 32 gigabytes of memory. Server 13 may also have, for example, four terabytes of hard disk storage. While the applications often involve integer programming problems, which are memory-intensive, server 13 provides more than adequate capacity for the past and current applications.

Users and developers use the Secure Shell Protocol (SSH) of computer 12 to navigate on server 13 over network 15 and to run programs on server 13, typically using a software package called PuTTY. Moreover, the Common Internet File System/Server Message Block (CIFS/SMB) may be used to read files on server 13 and to write files to it.

The program to run the certainty-uncertainty searches for the optimal portfolio, such as executing the flocharts shown in Figures 2.2–2.4, may be written in General Algebraic Modeling System (GAMS), a tool for solving mixed-integer linear programming optimization problems. At its core, GAMS can use many “solvers”; each may have many different ways of evaluating the problem. Thus far, the IBM ILOG CPLEX Optimization Studio (often informally referred to simply as CPLEX) has been used. In a simulation to find the optimal portfolio (OP) for a given budget, the CPLEX Solver was typically used tens of thousands of times to see which projects were selected under each of numerous sets of possible outcomes randomly drawing from the uncertain input parameters. Both GAMS 23.3 and the CPLEX Solver may be run on Red Hat Enterprise Linux (RHEL) 5 or higher on server 13. GAMS provides several output files, but an output file has been scripted to fit the specific needs.

Thus, the input file, GAMS, CPLEX, the program, and the output files all reside on server 13. Further, server 13 and computer 12 may be inside a corporate firewall.

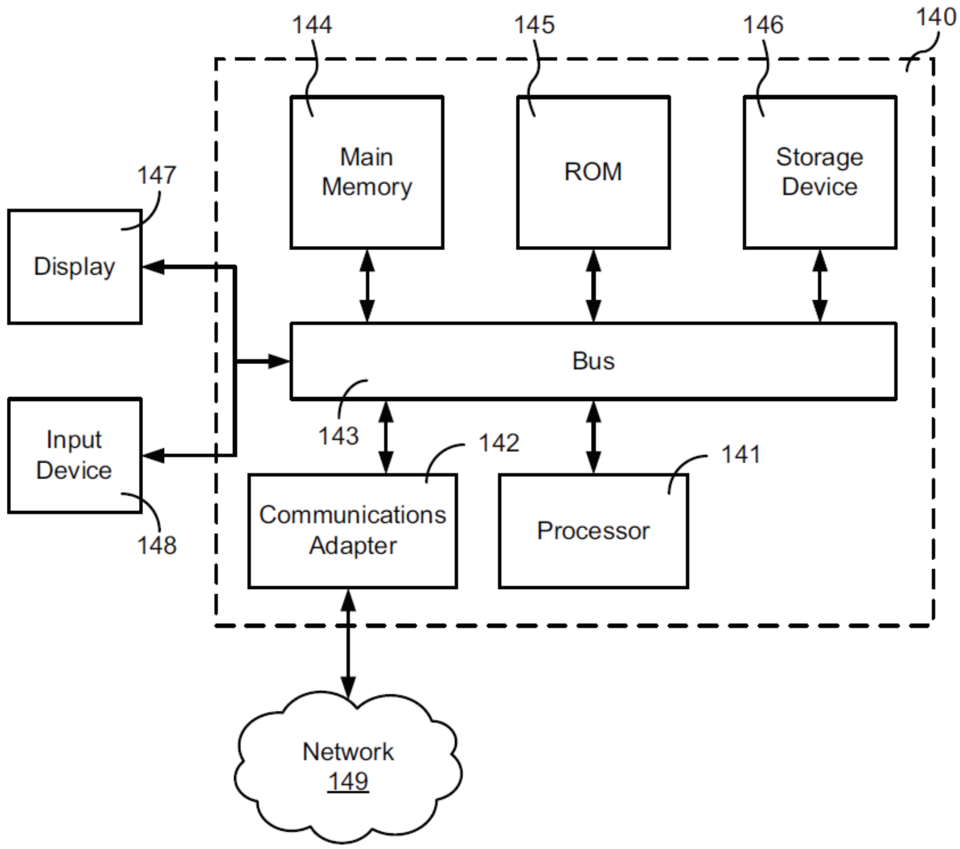
The output files, which may also be in the form of an Excel-compatible format, such as CSV, can be downloaded to a computer, such as computer 12 and/or computer 11 for post-run processing on the computer, including the preparation of graphs, tables, and texts and their placement into briefings and reports.

An Example of a Computer System Connected to a Network

Figure A.2 is an example of a computer system (140) connected to a network (149). The computing system (140) can be used, for example, for implementing the methods described in the approach discussed in the main text and shown in Figures 2.2–2.4. With reference to Figures A.1 and A.2, computer 11, computer 12, and server 13 in Figure A.1 could be implemented, for example, by respective individual embodiments of computing system 140. Computing system 140 includes a bus (143) or other communication component for communicating information and a processor (141) coupled to the bus (143) for processing information. Computing system 140 also includes main memory (144), such as a random access memory (RAM) or other dynamic storage device, coupled to bus 143 for storing information and instructions to be executed by processor 141. Main memory 144 can also be used for storing position information, temporary variables, or other intermediate information during execution of instructions by processor 141. Computing system 140 may further include a read-only memory (ROM) (145) or other static storage device coupled to bus 143 for storing static information and instructions for processor 141. A storage device (146)—such as a nontransitory, solid-state device, magnetic disk, or optical disk—is coupled to bus 143 for persistently storing information and instructions.

Computing system 140 may be coupled through bus 143 to display 147, such as a liquid crystal display or active matrix display, for displaying information to a user. Input device 148, such as a keyboard, including alphanumeric and other keys, may be coupled to bus 143 for communicating information and command selections to processor 141. In another implementation, input device 148 includes a touch-screen display. Input device 148 can also include a cursor control, such as a mouse, a trackball, or cursor direction keys, for communicating direction information and command selections to processor 141 and for controlling cursor movement on display 147.

Figure A.2. An Example of a Computer System Connected to a Network



In some implementations, computing system 140 may include a communications adapter (142), such as a networking adapter. Communications adapter 142 may be coupled to bus 143 and may be configured to enable communications with computing or communications network 149 and/or other computing systems. In various illustrative implementations, any type of networking configuration may be achieved using communications adapter 142, such as wired (for example, through an Ethernet connection), wireless (for example, through Wi-Fi, Bluetooth, worldwide interoperability for microwave access [WiMAX], etc.), preconfigured, ad hoc, LAN, WAN, etc.

According to various implementations, various processes that bring about illustrative implementations can be achieved using computing systems, such as computing system 140. Processor 141 can execute an arrangement of instructions contained in main memory 144. Such instructions can be read into main memory 144 from another computer-readable medium, such as storage device 146. Execution of the arrangement of instructions contained in main memory 144 causes computing system 140 to perform processes. One or more processors in a multi-processing arrangement may also be used to execute the instructions contained in main memory 144. In alternative implementations, hard-wired circuitry may be used in place of, or in combination with, software instructions to implement illustrative implementations.

Although an example processing system has been described in Figure 2.2, implementations of the subject matter and the functional operations described in this specification can be carried out using other types of digital electronic circuitry; in computer software, firmware, or hardware, including the structures disclosed in this specification and their structural equivalents; or in combinations of one or more of them.

Various implementations can be implemented as one or more computer programs—that is, one or more modules of computer program instructions, encoded on one or more nontransitory computer readable storage mediums for execution by, or to control the operation of, data processing apparatuses. A computer-readable storage medium can be, or can be included in, a computer-readable storage device, a computer-readable storage substrate, a random or serial access memory array or device, or a combination of one or more of them. Moreover, while a computer-readable storage medium is not a propagated signal, a computer-readable storage medium can be a source or destination of computer program instructions encoded in an artificially generated propagated signal. The computer-readable storage medium can also be, or be included in, one or more separate components or media (for example, multiple CDs, disks, or other storage devices). Accordingly, a computer-readable storage medium is tangible.

Various operations described in this specification can be implemented as operations performed by a data processing apparatus on data stored on one or more computer-readable storage devices or received from other sources. The term “data processing apparatus” or “computing device” encompasses all kinds of apparatuses, devices, and machines for processing data, including by way of example a programmable processor, a computer, a system on a chip, or multiples or combinations of the foregoing. An apparatus can include special-purpose logic circuitry—for example, a field programmable gate array (FPGA) or an application-specific integrated circuit (ASIC). An apparatus can also include, in addition to hardware, code that creates an execution environment for a computer program—for example, code that constitutes processor firmware, a protocol stack, a database management system, an operating system, a cross-platform runtime environment, a virtual machine, or a combination of one or more of them. The apparatus and execution environment can realize various different computing model infrastructures, such as web services, distributed computing, and grid computing infrastructures.

A computer program (also known as a program, software, software application, script, or code) can be written in any form of programming language—including compiled or interpreted languages and declarative or procedural languages—and can be deployed in any form, including as a stand-alone program or as a module, component, subroutine, object, or other unit suitable for use in a computing environment. A computer program may, but need not, correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (for example, one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (for example, files that store one or more modules, sub-programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

The processes and logic flows described in this specification can be performed by programmable processors executing one or more computer programs to perform actions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatuses can also be implemented as, special-purpose logic circuitry—for example, an FPGA or an ASIC.

Processors suitable for executing a computer program include, by way of example, both general and special-purpose microprocessors and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read-only memory, a random-access memory, or both. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data (for example, magnetic, magneto-optical disks, or optical disks). However, a computer need not have such devices. Moreover, a computer can be embedded in another device—for example, a mobile telephone, a personal digital assistant (PDA), a mobile audio or video player, a game console, a Global Positioning System (GPS) receiver, or a portable storage device (for example, a universal serial bus [USB] flash drive), to name just a few. Devices suitable for storing computer program instructions and data include all forms of nonvolatile memory, media, and memory devices, including, by way of example, semiconductor memory devices (for example, erasable programmable read-only memory [EPROM], electrically erasable programmable read-only memory [EEPROM], and flash memory devices); magnetic disks (for example, internal hard disks or removable disks); magneto-optical disks; and

CD-ROMs and DVD-ROMs. A processor and a memory can be supplemented by, or incorporated in, special-purpose logic circuitry.

To provide for interaction with a user, various implementations can be carried out using a computer having a display device—for example, a cathode ray tube (CRT) or liquid crystal display (LCD) monitor—for displaying information to the user and a keyboard and a pointing device—such as a mouse or a trackball—by which the user can provide input to the computer. Other kinds of devices can be used to provide for interaction with a user as well; for example, feedback provided to the user can be any form of sensory feedback, e.g., visual feedback, auditory feedback, or tactile feedback; and input from the user can be received in any form, including acoustic, speech, or tactile input. In addition, a computer can interact with a user by sending documents to, and receiving documents from, a device that is used by the user, for example, by sending web pages to a web browser on a user's client device in response to requests received from the web browser.

Various implementations can be carried out using computing systems that include a back-end component (for example, as a data server), that include a middleware component (for example, an application server), that include a front-end component (for example, a client computer having a graphical user interface or a web browser), or any combination of one or more such back-end, middleware, or front-end components. The components of the system can be interconnected by any form or medium of digital data communication, such as a communication network. Examples of communication networks include a local LAN and a WAN, an inter-network (such as the Internet), and peer-to-peer networks (such as ad hoc peer-to-peer networks).

Computing systems can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other. In some implementations, a server transmits data (for example, an HTML page) to a client device (for example, to display data to and receive user input from a user interacting with the client device). Data generated at the client device (for example, a result of the user interaction) can be received from the client device at the server.

While this specification contains many specific implementation details, these should not be construed as limitations on the scope of any inventions or of what may be claimed, but rather as descriptions of features specific to particular implementations of particular inventions. Certain features that are described in this specification in the context of separate implementations can also be carried out in combination in a single implementation. Conversely, various features that are described in the context of a single implementation can also be carried out in multiple implementations separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a combination can in some cases be excised from the combination, and the combination may be directed to a subcombination or variation of a subcombination.

Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. In certain circumstances, multitasking and parallel processing may be advantageous. Moreover, the separation of various system components in the implementations described above should not be understood as requiring such separation in all implementations, and it should be understood that various described program components and systems can generally be integrated together in a single software product or packaged into multiple software products.

Particular implementations of the subject matter have been described. Other implementations are within the scope of the invention. In some cases, the actions recited can be performed in a different order and still achieve desirable results. In addition, the processes depicted in the accompanying figures do not necessarily require the particular order shown, or sequential order, to achieve desirable results. In certain implementations, multitasking and parallel processing may be advantageous.

Bibliography

Beale, E. M. L., "On Minimizing a Convex Function Subject to Linear Inequalities," *Journal of the Royal Statistical Society*, Vol. 17, No. 2, 1955, pp. 173–184.

Bellman, Richard, *Dynamic Programming*, Princeton University Press, 1957.

Bertsimas, Dimitris, Vineet Goyal, and Xu Andy Sun, "A Geometric Characterization of the Power of Finite Adaptability in Multistage Stochastic and Adaptive Optimization," *Mathematics of Operations Research*, Vol. 36, No. 1, February 2011, pp. 24–54.

Birge, John, and Francois Louveaux*, Introduction to Stochastic Programming*, Second Edition, Springer Series on Operations Research and Financial Engineering, 2010.

Black, Paul E., "Polynomial Time," in *Dictionary of Algorithms and Data Structures* (online), Paul E. Black, ed., U.S. National Institute of Standards and Technology. August 13, 2004. As of February 27, 2013:
http://www.nist.gov/dads/HTML/polynomialtm.html

Chen, Xin, Melvyn Sim, Peng Sun, and Jiawei Zhang, "A Linear Decision-Based Approximation Approach to Stochastic Programming," *Operations Research*, Vol. 56, No. 2, March–April 2008, pp. 344–357.

Chow, Brian, Richard Silberglitt, and Scott Hiromoto, *Toward Affordable Systems: Portfolio Analysis and Management for Army Science and Technology Programs*, Santa Monica, Calif.: RAND Corporation, MG-761-A, 2009. As of March 11, 2013:
http://www.rand.org/pubs/monographs/MG761.html

Chow, Brian, Richard Silberglitt, Scott Hiromoto, Caroline Reilly, and Christina Panis, *Toward Affordable Systems II: Portfolio Management for Army Science and Technology Programs Under Uncertainties*, Santa Monica, Calif.: RAND Corporation, MG-979-A, 2011. As of March 8, 2013:
http://www.rand.org/pubs/monographs/MG979.html

Chow, Brian, Richard Silberglitt, Caroline Reilly, Scott Hiromoto, and Christina Panis, *Choosing Defense Project Portfolios: A New Tool for Making Optimal Choices in a World of Constraint and Uncertainty*, Santa Monica, Calif.: RAND Corporation, RB-9678-A, 2012. As of March 8, 2013:
http://www.rand.org/pubs/research_briefs/RB9678.html

Chow, Brian G., Richard Silberglitt, Caroline Reilly, Scott Hiromoto, and Christina Panis, *Toward Affordable Systems III: Portfolio Management for Army Engineering and Manufacturing Development Programs*, Santa Monica, Calif.: RAND Corporation, MG-1187-A, 2012. As of March 8, 2013:
http://www.rand.org/pubs/monographs/MG1187.html

Dantzig, George, "Linear Programming Under Uncertainty," *Management Science*, Vol. 1, Nos. 3 and 4, April–July 1955, pp. 197–206.

Davis, Paul, *Lessons from RAND's Work on Planning Under Uncertainty for National Security*, Santa Monica, Calif.: RAND Corporation, TR-1249-OSD, 2012. As of March 8, 2013: http://www.rand.org/pubs/technical_reports/TR1249.html

Davis, Paul, and Paul Dreyer, *RAND's Portfolio Analysis Tool (PAT): Theory, Methods, and Reference Manual*, Santa Monica, Calif.: RAND Corporation, TR-756-OSD, 2009. As of March 8, 2013:
http://www.rand.org/pubs/technical_reports/TR756.html

Davis, Paul, Russell Shaver, Gaga Gvineria, and Justin Beck, *Finding Candidate Options for Investment Analysis: A Tool for Moving from Building Blocks to Composite Options (BCOT)*, Santa Monica, Calif.: RAND Corporation, TR-501-OSD, 2008. As of March 8, 2013: http://www.rand.org/pubs/technical_reports/TR501.html

Goldberg, David E., *Genetic Algorithms in Search Optimization and Machine Learning*, Addison Wesley, 1989.

Hillestad, Richard, and Paul Davis, *Resource Allocation for the New Defense Strategy: The Dynarank Decision Support System*, Santa Monica, Calif.: RAND Corporation, MR-996-OSD, 1998. As of March 8, 2013:
http://www.rand.org/pubs/monograph_reports/MR996.html

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, Vol. 220, No. 4598, 1983.

Landree, Eric, Richard Silberglitt, Brian G. Chow, Lance Sherry, and Michael S. Tseng, *A Delicate Balance: Portfolio Analysis and Management for Intelligence Information Dissemination Programs*, Santa Monica, Calif.: RAND Corporation, MG-939-NSA, 2009. As of March 11, 2013: http://www.rand.org/pubs/monographs/MG939.html

Lempert, Robert, David Groves, Steven Popper, and Steven Bankes, "A General, Analytic Method for Generating Robust Strategies and Narrative Scenarios," *Management Science*, Vol. 52, No. 4, April 2006, pp. 514–552.

Lempert, Robert, Steven Popper, and Steven Bankes, *Shaping the Next One Hundred Years: New Methods for Quantitative Long-Term Policy Analysis*, Santa Monica, Calif.: RAND Corporation, MR-1626-RPC, 2003. As of March 8, 2013:
http://www.rand.org/pubs/monograph_reports/MR1626.html

Luenberger, David, and Yinyu Ye, *Linear and Nonlinear Programming*, Third Edition, International Series in Operations Research and Management Science, Springer, 2008.

Pennanen, Teemu, "Convex Duality in Stochastic Optimization and Mathematical Finance," *Mathematics of Operations Research*, Vol. 36, No. 2, May 2011, pp. 340–362.

Sahinidis, Nikolaos, "Optimization Under Uncertainty: State-of-the-Art and Opportunities," *Computers and Chemical Engineering 28*, Elsevier Ltd., 2004, pp. 971–983.

So, Anthony Man-Cho, Jiawei Zhang, and Yinyu Ye, “Stochastic Combinatorial Optimization with Controllable Risk Aversion Level,” *Mathematics of Operations Research*, Vol. 34, No. 3, August 2009, pp. 522–537.

Zhang, Xinhui, Meenakshi Prajapati, and Eugene Peden, “A Stochastic Production Planning Model Under Uncertain Seasonal Demand and Market Growth,” *International Journal of Production Research*, Vol. 49, No. 7, April 1, 2011, pp. 1957–1975.

(This page is intentionally left blank.)