

OPTUM: Optimum Portfolio Tool for Utility Maximization Documentation and User's Guide

Decision and Information Sciences Division



About Argonne National Laboratory

Argonne is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC under contract DE-AC02-06CH11357. The Laboratory's main facility is outside Chicago, at 9700 South Cass Avenue, Argonne, Illinois 60439. For information about Argonne, see www.anl.gov.

Availability of This Report

This report is available, at no cost, at <http://www.osti.gov/bridge>. It is also available on paper to the U.S. Department of Energy and its contractors, for a processing fee, from:

U.S. Department of Energy

Office of Scientific and Technical Information

P.O. Box 62

Oak Ridge, TN 37831-0062

phone (865) 576-8401

fax (865) 576-5728

reports@adonis.osti.gov

Disclaimer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor UChicago Argonne, LLC, nor any of their employees or officers, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of document authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof, Argonne National Laboratory, or UChicago Argonne, LLC.

**OPTUM:
Optimum Portfolio Tool for Utility Maximization
Documentation and User's Guide**

by
J.C. VanKuiken, M.J. Jusko, and M.E. Samsa
Decision and Information Sciences Division, Argonne National Laboratory

September 2008



CONTENTS

ACKNOWLEDGMENTS	vi
ABSTRACT	1
SUMMARY	1
1 INTRODUCTION	3
2 APPROACH	5
2.1 Overall Description	5
2.2 Overview of Steps in the Portfolio Analysis Process	5
2.2.1 Establishing Objectives	6
2.2.2 Establishing the Relative Importance of the Objectives	6
2.1.3 Setting Scale Ranges for Rating Objectives	6
2.2.4 Identifying, Costing, and Categorizing Projects	7
2.2.5 Rating Projects According to Objective Scales	9
2.2.6 Defining Portfolio Constraints	9
2.2.7 Choosing Portfolio Selections and Performing Sensitivity Studies	9
3 USER ENTRIES AND VALUE/UTILITY DERIVATIONS	11
3.1 Notation for the Sample Problem	11
3.2 Objectives & Projects Tab	11
3.2.1 Entering Objective Names and Descriptions	12
3.2.2 Defining Objective Scales: Scale Types and Units of Measure	12
3.2.3 Defining Objective Scale Ranges	14
3.2.4 Assigning Relative Values to Objective Scale Ranges	15
3.2.5 Assigning Relative Weights to Each Objective	16
3.2.6 Identifying and Costing the Projects	16
3.2.7 Rating Projects According to the Objectives	16
3.3 Project Value and Utility Derivations for the Sample Problem	21
3.3.1 Overview of Sample Problem Derivations	21
3.3.2 Detailed OPTUM Derivations	21
4 CONSTRAINTS AND ANALYSIS	31
4.1 Constraints & Analysis Tab	31
4.1.1 Portfolio Optimization with Total Portfolio Constraints	32
4.1.2 Portfolio Optimization with Project-specific Constraints	34
4.1.3 Portfolio Optimization with More Complex Constraints	37
4.1.3.1 Simple Conditional Constraint Example	38
4.1.3.2 Example of Constraint Implementation for Sensitivity Analysis	38
4.2 Results, Graphs, and Sensitivity Analysis	40
4.2.1 Modification of Cost Constraints for Sensitivity Analysis	41

CONTENTS (Cont.)

4.2.2	Graphical Display of Optimal “Frontier” Solutions	41
4.2.3	Graphical Display of Alternative Optimal Solutions.....	43
4.3	Exporting Results.....	45
4.4	Advanced Analysis Capabilities	45
4.5	User Preferences	45
5	OPTIMIZATION ENGINE.....	48
5.1	Rationale for Using Optimization.....	48
5.2	Optimizer Selection Criteria	49
5.2.1	Cost and Licensing Requirements	49
5.2.2	Problem-solving Capabilities.....	50
5.2.3	Problem Size and Speed.....	51
5.2.4	Ease of Use and Implementation	51
5.2.5	Prior Testing and Verification	51
5.2.6	Software Documentation	51
5.2.7	Maintenance.....	51
5.3	Initial Optimizer Candidates.....	52
5.4	Selected Solver.....	52
5.5	Implementation	53
6	CONCLUSIONS	54
7	REFERENCES	55
	APPENDIX A: Conditional Constraint Formulations.....	57

FIGURES

2.1	Preview of Basic OPTUM User Input Screen	7
2.2	Preview of Sample OPTUM Constraints & Analysis Screen.....	8
2.3	Opening Screen for OPTUM	10
3.1	Objective Name and Description Fields	13
3.2	Objective Scale Types.....	13
3.3	Objective Scale Units of Measure, Scale Ranges, and Relative Values.....	14

FIGURES (Cont.)

3.4	Entering Project Names and Descriptions	17
3.5	Project Ratings: Distribution Types, Scales, and Relative Likelihoods	17
3.6	Distribution Types for Objectives with Continuous-scale Types	18
3.7	Manual Selection of Projects for a Portfolio.....	20
3.8	Summary Screen with Intermediate Calculation Results Displayed	20
4.1	Capabilities Available from Constraints & Analysis Screen.....	31
4.2	Optimized Results with Cost Constraints	33
4.3	Optimized Results with Modified Cost Constraints	34
4.4	Portfolio Optimization with Project-specific Categorization Constraints	35
4.5	Results with Project Categorization Constraints Included.....	36
4.6	Conditional Constraint Options	38
4.7	Optimal Portfolio Results	40
4.8	Setup Screen for Preparing Optimal Frontier Graph	42
4.9	Example of Optimal Frontier Graph.....	43
4.10	Menu for Constructing Alternate Solution Graph	44
4.11	Illustration of Alternate Solution Graph	44
4.12	User Preferences Menu	46
A.1	Conditional Constraint Options	57

TABLES

3.1	Sample Problem and Internal OPTUM Calculations.....	22
3.2	Evaluation and Optimization of Portfolio.....	24

ACKNOWLEDGMENTS

The authors would like to express their appreciation to the following individuals who contributed significantly to this report. William Buehring and Ronald Whitfield provided valuable guidance and feedback on the design and functionality of OPTUM. Their knowledge of decision analysis and their experience in developing previous software support tools provided an excellent foundation for the development of OPTUM. Stanko Dimitrov developed the logic rules for implementing the conditional constraints. Brian Craig provided valuable support and expertise on software design issues and problem resolution. Robbie Davidson constructed and annotated the screen images critical in this report for illustrating model features and operational procedures.

OPTIMUM PORTFOLIO TOOL FOR UTILITY MAXIMIZATION DOCUMENTATION AND USER'S GUIDE

by

J.C. VanKuiken, M.J. Jusko, and M.E. Samsa

ABSTRACT

The Optimum Portfolio Tool for Utility Maximization (OPTUM) is a versatile and powerful tool for selecting, optimizing, and analyzing portfolios. The software introduces a compact interface that facilitates problem definition, complex constraint specification, and portfolio analysis. The tool allows simple comparisons between user-preferred choices and optimized selections. OPTUM uses a portable, efficient, mixed-integer optimization engine (`lp_solve`) to derive the optimal mix of projects that satisfies the constraints and maximizes the total portfolio utility. OPTUM provides advanced features, such as convenient menus for specifying conditional constraints and specialized graphical displays of the optimal frontier and alternative solutions to assist in sensitivity visualization. OPTUM can be readily applied to other nonportfolio, resource-constrained optimization problems.

SUMMARY

The Optimum Portfolio Tool for Utility Maximization (OPTUM) is a powerful tool for selecting, optimizing, and analyzing portfolios. The software provides options for making user-preferred selections and comparing them with optimized project selections. A wide range of constraints can be introduced to accommodate various resource limitations or upper and lower limits on the numbers of projects. Complex conditional constraints can also be accommodated, whereby inclusion of one or more projects can be forced to depend on selections of other projects. The optimized solution will then reflect all of these constraints in the collection of projects selected for the optimal portfolio.

Graphs of the optimal “frontiers” and alternate solutions are included to help the user understand the outcomes in the context of other available possibilities. These graphic options provide useful tools for exploring sensitivities and near-optimal alternatives.

The optimization engine `lp_solve` (Linderoth and Ralphs 2005; Notebaert et al. 2008) was selected on the basis of a number of criteria that included portability, cost, licensing considerations, and problem size capability. It performed very well in all test cases and continues to be maintained, updated, and verified by a large user community.

OPTUM can be applied to other nonportfolio resource allocation problems. The framework is very flexible, allowing the user to customize the problem objectives and scales for rating each of the alternatives to be considered.

Significant attention was devoted to simplifying the problem construction process and streamlining the data input interfaces. This effort resulted in a single spreadsheet-like form for entering all the information related to defining the basic problem (objective definitions and project evaluations). Likewise, a single screen covers all the data entry needs for defining basic resource constraints and viewing outcomes. Another screen is dedicated to the construction of more complex conditional constraints. The end result is a convenient and compact interface that facilitates problem construction and analysis.

1 INTRODUCTION

The Optimum Portfolio Tool for Utility Maximization (OPTUM) provides a structured environment to help users select specific projects or tasks from a choice of candidates that may be competing for limited resources. The overall goal is for the software to identify optimal project selections in the context of user-defined objectives and constraints. Additional functions of the tool yield insights into alternative, nearly optimal sets of projects and the incremental benefits associated with constraint relaxations.

The core functionality of OPTUM was patterned after the Portfolio Analysis Support System (PASS), an earlier decision support tool (Jusko et al. 2006). OPTUM extended the decision-analytic logic of the PASS precursor by adding new operational and display features that not only added to its analytical capabilities but also made it easier to use and interpret the results.

A primary goal of the OPTUM software tool is to maximize the value of selected projects (or tasks) on the basis of a set of user-defined objectives and constraints. The objectives are defined to capture various aspects of project performance or impacts that affect the overall perceived value of each project. Some examples of objectives are those that specify a project's completion time, scope of benefits, technical difficulty, or any other criterion that is measurable and of value to the decision maker. Each of the objectives is characterized by the user in terms of its relative importance.

After the objectives are established, each project or task is evaluated in terms of how well it achieves each objective. From the specific project objective "scores," an overall project value is calculated. OPTUM determines a priority ranking of all projects on the basis of how well each one has achieved its objectives. If no other considerations were active, this priority ranking would represent the logical order for selecting projects. However, usually there are other constraints associated with constructing a portfolio of projects that alter the selection of projects. For example, budgetary constraints are common, and other objectives (e.g., spreading project investments over different categories of activities like research and development [R&D], new ventures, or overhead) might represent additional competing factors to be considered in selecting the optimal projects.

OPTUM recognizes simple constraints, such as maximum total funding limitations or minimum funding for specific projects. It also treats more complex constraints, such as project prerequisites or mutual exclusions. Considerations for these complicating factors can significantly affect the selection of optimal projects. OPTUM provides a tool that explicitly recognizes such constraints and captures their effects on the selection of projects for an optimal portfolio.

Although OPTUM was designed with a focus on project portfolio evaluations, it can be used to analyze virtually any set of alternatives that compete for limited resources, whether monetary, human, material, or others.

This report serves as a user guide and describes the fundamental decision-analytic problem representation. It is intended to describe how (1) a problem is formulated, (2) project values are derived, (3) the optimization engine is invoked, and (4) the user can apply visualization tools to analyze and probe the results for sensitivities.

Section 2 provides an overview of basic steps used in OPTUM to obtain an optimal portfolio. Sections 3 and 4 contain examples to illustrate the use of OPTUM and derivations to show how the fundamental project values are obtained and analyzed. Section 5 documents the optimization engine selection process, the criteria that were considered, the engines that were examined, and the final choice that was made. Section 6 summarizes the report. Appendix A describes the internal logic rules that were formulated for each of the conditional constraints.

2 APPROACH

2.1 OVERALL DESCRIPTION

OPTUM treats the primary goal of portfolio selection as a mathematical maximization problem — as a problem of how to maximize the achievement of a set of objectives. Each project contributes to the achievement of the objectives to a different degree, and assessing how well a project achieves the objectives determines that project's relative value or importance. OPTUM provides a structured framework for (1) identifying and characterizing the objectives, (2) identifying candidate projects and assessing how effectively each one achieves the objectives, (3) characterizing resource and other types of limitations, and (4) selecting from the list of candidate projects the optimal set of projects that would maximize the overall achievement of the objectives.

If there were no resource or other limits in effect, then simple priority rankings and assessments for each project would result in a logical order for selecting projects. However, there are usually other constraints associated with a portfolio of projects that alter the optimal selection of projects. Budgetary constraints are common, but other considerations (e.g., spreading resource investments over several critical programmatic areas) are also typical.

OPTUM provides the user with convenient mechanisms for implementing constraints and optimizing a portfolio when virtually any number of constraints are present. When constraints are included in the analysis, OPTUM identifies the set of projects that would maximize the portfolio's value in terms of meeting the objectives while satisfying the constraint requirements. If a problem becomes over-constrained (i.e., meaning there is no valid solution that would satisfy all of the constraints), a warning message is displayed so the user knows that adjustments are needed.

2.2 OVERVIEW OF STEPS IN THE PORTFOLIO ANALYSIS PROCESS

There are seven steps in the portfolio optimization process. OPTUM uses spreadsheet-like entry forms that are designed to help the user complete these steps. Constraints can be simple or complex, and multiple sets of constraints can be implemented. Unconstrained cases can also be evaluated. An overview of the steps follows:

1. Establish the objectives;
2. Assign relative weights to each objective;
3. Set scale ranges for rating each project according to the objectives;
4. Identify, cost, and categorize the projects;
5. Rate projects according to the objectives;

6. Define portfolio constraints; and
7. Optimize portfolio selections and perform sensitivity analyses.

These steps are briefly described below and are presented with examples in Sections 3 and 4 of this report.

2.2.1 Establishing Objectives

This first step is perhaps the most critical one in the OPTUM process. Three sources — Keeney (1992), Keeney and Raiffa (1993), and Hammond et al. (1998) — describe how the objectives and the scales to measure their achievement are established. Some examples of objectives are the project cost, scope of benefits, degree of risk, and completion time. For those unfamiliar with establishing objectives and scales, the three sources serve as useful tutorials for setting up a new problem. If the objectives are not established properly, then the portfolio analysis that follows may be misleading. Carefully setting the objectives will simplify and clarify the portfolio analysis process.

2.2.2 Establishing the Relative Importance (Weights) of the Objectives

Objectives typically vary in terms of their relative importance or value. The user must establish the relative importance of each of the objectives defined in Step 1. In OPTUM, the user selects a “Relative Weight of Objective” between 0 and 100 to represent the importance of each objective relative to other defined objectives. The least desirable value is 0, and the most desirable value is 100. Objective weights between 0 and 100 have intermediate values that are proportional to the value of the objective (i.e., an objective valued at 50 points is worth half of an objective valued at 100 points in terms of relative importance). Each objective should be assigned a value greater than zero, unless the user is conducting a sensitivity analysis to determine the effect that a given objective has on the portfolio choices. Only the *relative* values are important. This means that if all objectives are assigned 50 points, then all objectives are treated as being of equal value. OPTUM performs the necessary value normalizations after the user has specified relative weights.

2.1.3 Setting Scale Ranges for Rating Objectives

An objective can be satisfied by various projects at different levels, and the level of achievement has an associated importance or value to the user. OPTUM offers two general types of scale ranges for rating how projects satisfy objectives: continuous or discrete. Within the two categories of scale ranges, the program provides eight types of continuous distributions (e.g., triangular, normal, and lognormal are three of them) and an unlimited number of user-defined discrete rating scales. After determining the preferred choices of scale range types, the user can specify the parameters that define best and worst values in the scale ranges, as well as points between the best and worst outcomes.

Figure 2.1 shows a preview of the primary input screen, which is used to define the objectives as described above. The objectives are defined in the area between the “Objectives” and “Projects” headings (headings are highlighted in yellow). Green and blue boxes represent user inputs to define the portfolio selection problem. User inputs for objectives include names, descriptions, scale type, scale units (for continuous scales), scale range descriptors, relative value of scale range, and relative weight of objective. More comprehensive descriptions of these input parameters are provided in Section 3.

2.2.4 Identifying, Costing, and Categorizing Projects

The same spreadsheet-like form used for entering objective information (Figure 2.1) is also used for entering information about the candidate projects that are to be considered for possible inclusion in the optimal portfolio. The user enters abbreviated project titles (e.g., P1, P2, P3, ...) and can also include longer descriptions of the projects. The longer titles are optional, but the short identifying descriptors are necessary.

The screenshot shows the OPTUM software interface with a menu bar (File, Objectives, Scale Levels, Projects, Categories, Tools, Help) and a toolbar. The main window displays a spreadsheet with the following data:

Row	Column	Content														
1	A	Small Sample Case														
2	A	Small sample portfolio, for illustrative purposes only.														
5	Objectives															
6		Objective Name:	Completion Time	Scope of Benefits	Technical Difficulty											
7		Description:	Time to complete project.	Area of benefits achieved.	Measure of complexity of project.											
9		Scale Type (Continuous, Constructed):	Continuous Scale	Constructed Scale	Constructed Scale											
10		Scale Units:	Months													
11		Scale Range Descriptor:	Best	Worst	Global	National	Regional	Local	High	Medium	Low					
12		Relative Value of Scale Range:	0	24	70	100	20	10	0	100	30					
14		Relative Weight of Objective:	80		100			20								
18	Projects															
20	P1	Distribution Type (if Continuous):	Triangular													
21	Project 1	Distribution / Scale Descriptors:	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low	Cost	Utility	Selected?	
22		Relative Likelihood of Project Outcomes:	2	8	18	40	20	20	0	0	100	0	1000	0.66944444		
28	P2	Distribution Type (if Continuous):	Triangular													
30	Project 2	Distribution / Scale Descriptors:	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low				
31		Relative Likelihood of Project Outcomes:	4	6	24	60	100	20	20	20	100	80	900	0.64311111		
37	P3	Distribution Type (if Continuous):	Triangular													
38	Project 3	Distribution / Scale Descriptors:	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low				
40		Relative Likelihood of Project Outcomes:	5	6	24	60	100	20	20	20	100	80	800	0.63751805	✓	
46	P4	Distribution Type (if Continuous):	Triangular													
47	Project 4	Distribution / Scale Descriptors:	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low				
49		Relative Likelihood of Project Outcomes:	6	6	24	60	100	20	20	20	100	80	780	0.63199875		
55	P5	Distribution Type (if Continuous):	Triangular													
57	Project 5	Distribution / Scale Descriptors:	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low				
58		Relative Likelihood of Project Outcomes:	7	8	24	60	100	20	20	20	100	80	950	0.61531165		
64	P6	Distribution Type (if Continuous):	Triangular													
65	Project 6	Distribution / Scale Descriptors:	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low				
67		Relative Likelihood of Project Outcomes:	9	10	24	60	100	20	20	20	100	80	600	0.59309029	✓	
73	P7	Distribution Type (if Continuous):	Triangular													
75	Project 7	Distribution / Scale Descriptors:	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low				
76		Relative Likelihood of Project Outcomes:	18	20	22	10	20	20	50	100	40	10	400	0.27533333		
82													Total of All Projects:	5430	4.06580763	
83													Total of Selected (✓) Projects:	1400	1.23060834	2

FIGURE 2.1 Preview of Basic OPTUM User Input Screen (Objective and Project Definitions)

Data entry fields also include an opportunity to provide point estimates for the costs of each project (next-to-last column on right side of the screen). These cost data are used to generate useful rank-ordering metrics for projects (according to utility-per-unit-cost rankings), and they are also used in cost constraint calculations. In addition to point estimates for costs, uncertainty ranges for the cost estimates can also be explicitly captured through user-defined cost objectives. These can be characterized by probability distributions or discrete ranges.

OPTUM provides a mechanism for categorizing projects according to user-defined categories (e.g., as construction versus R&D versus overhead, or as military versus civilian, or by corporate division). A simple matrix with inclusion/exclusion buttons allows the user to define which projects belong to which categories. These categorizations provide convenient tools for implementing constraints (e.g., for specifying that at least 20% of the budget must be allocated to projects that qualify as R&D activities). Examples are given in Section 4 to illustrate how these categorizations and constraints can be used to represent relatively complex conditional inclusion requirements. An example matrix for project categorizations is shown in the upper right portion of Figure 2.2.

Project Name	Utility ¹	Cost ¹	Utilis/Unit Cost ¹	Portfolio Choices		Project Categories		
				Optimal	Selected	Cat 1	Cat 2	Cat 3
P1	0.66944444	1000	0.000669444			•	•	•
P2	0.64311111	900	0.000714568			•	•	•
P3	0.63751805	800	0.000796898		✓	•	•	•
P4	0.63199875	780	0.000810255			•	•	•
P5	0.61531165	950	0.000647696			•	•	•
P6	0.59309029	600	0.000988484		✓	•	•	•
P7	0.27533333	400	0.000688333			•	•	•
Total:		4.06580763	5430					
Average (excluding 0-cost projects):		0.58082966	775.714286	0.000759383				
Unconstrained Totals				Utility:	4.065807628	2.19760748	0.94477778	3.12102985
				Utilis / Unit Cost:	0.000759383	7.0059E-04	6.7889E-04	0.00079158
Constraints on Cost of Portfolio				Maximum Possible:	5430	3150	1400	4030
				Minimum:				
				Maximum:	1500			
Constraints on Number of Projects in Portfolio				Maximum Possible:	7	4	2	5
				Minimum:				
				Maximum:		2		
Optimal (★) Portfolio Summary				Cost:	0	0	0	0
				Number of Projects:	0	0	0	0
				Utility:	0	0	0	0
				Utilis / Unit Cost:		1.7515E-04	3.3944E-04	1.5832E-04
Your Selected (✓) Portfolio Summary ²				Cost:	1400	800	0	1400
				Number of Projects:	2	1	0	2
				Utility:	1.23060834	0.63751805	0	1.23060834
				Utilis / Unit Cost:	8.9269E-04	1.9441E-04	1.6972E-04	2.7767E-04

Symbols
★ These are projects in the optimal portfolio.
✓ These are projects you selected for your customized portfolio.

Notes
¹ Portfolio utilities and costs shown in italic are better than average.
² Values shown in red violate constraints.

FIGURE 2.2 Preview of Sample OPTUM Constraints & Analysis Screen

2.2.5 Rating Projects According to Objective Scales


Figure 2.1 illustrates the OPTUM screen that is used to rate or score each project in terms of how it is expected to fulfill each of the objectives. These ratings are entered next to the name and description information. For objectives with continuous-scale ranges, the user specifies distribution parameters (e.g., for triangular distributions: minimum, maximum, and mode parameters would be specified). For objectives with discrete-scale ranges, the user estimates the relative likelihood of the project falling in each of the scale ranges. To simplify the entry process, the ratings (0–100) are interpreted as relative, and the program performs the necessary normalizations to complete the calculations.

2.2.6 Defining Portfolio Constraints


Portfolio constraints can be easily entered in the screen shown in Figure 2.2. For the total portfolio, constraints can be implemented as upper or lower bounds on costs or total numbers of projects to be selected. In addition, for each of the user-defined project categorizations (e.g., construction, R&D, or overhead), the user can specify upper and lower bounds for the total number or cost of projects selected from a given category. The constraints and categorization tools can be used creatively to represent conditional selection criteria, such as project prerequisites (e.g., the portfolio *must* include Project A *if* Project B is to be included). Section 4 provides examples of conditional constraint implementations.

2.2.7 Choosing Portfolio Selections and Performing Sensitivity Studies

At this stage (after Steps 1 through 6 are completed), OPTUM has all the information it needs to run the portfolio analysis and provide the user with results. The user can choose to manually select projects for the portfolio or use the optimization button “★” (or “☆” if constraints are defined and active) to invoke the integrated optimization engine. OPTUM provides the user with options for which types of constraints to apply for a given optimization trial (see User Preferences menu described in Section 4.5). If the “optimize” option is invoked, the optimized portfolio is displayed side by side with the user’s preferred selections, revealing any improvements that would result from the optimization and allowing for an easy comparison of project selections.

In addition, there are charting features, available via the “” button, that plot “optimal frontier” curves. These are curves that show the best solution for any given level of funding/investment. Two frontiers are charted. One is for the “unconstrained” optimal combinations of project selections that maximize total utility at any given funding level (total cost). The other frontier shows the constrained optimal solutions that are feasible within the envelope of user constraints (including not only total cost limits but also any category-specific cost limits and other user-defined constraints, such as the maximum or minimum numbers of projects or conditional constraints). These two charts of optimal frontiers provide valuable insights into the range of possibilities, the impact of constraints on optimal portfolio value, and possible improvements that can be obtained by relaxing the constraints. Additional sensitivity

analyses are readily performed by modifying the initial problem specifications and reoptimizing the portfolio.

Sections 3 and 4 contain examples to illustrate the use of OPTUM. The internal derivations of project values are also documented, so the user can better understand how the input parameters affect the outcomes. In addition, for the screen depicted in Figure 2.1, the user can activate the rows expand button “









The opening screen for OPTUM (Figure 2.3) provides a quick-start guide. The User Preferences button “
 or  button) b. Edit or add objective and project characterizations [Objectives & Projects Tab] - enter data in green-highlighted cells - right-click mouse in blue-highlighted cells c. Edit or add constraints [Constraints & Analysis Tab] d. Perform Analysis - Select Projects Manually ( button) - Optimize ( or  buttons) - Graph Results ( button) (3) Export results to Excel for custom formatting and printing ( button)" data-bbox="125 370 865 789"/>

FIGURE 2.3 Opening Screen for OPTUM

3 USER ENTRIES AND VALUE/UTILITY DERIVATIONS


This section describes the data entries and calculations that ultimately drive the overall portfolio optimization process. OPTUM provides a basic framework for defining and weighting the relative values of objectives. Then each project is evaluated in the context of those objectives to determine relative project values and utilities. In this context, “utility” represents a single measure of merit that combines all of the estimated outcomes and relative importance factors as assessed by the user. The OPTUM framework is designed to be flexible to accommodate a comprehensive range of objectives and project characterization metrics.

As indicated in Section 2, the preparation and analysis steps consist of the following:


1. Establish the objectives;
2. Assign relative weights to each objective;
3. Set scale ranges for rating the projects according to the objectives;
4. Identify, cost, and categorize the projects;
5. Rate projects according to the objectives;
6. Define portfolio constraints; and
7. Optimize portfolio selections and perform sensitivity analyses.

The procedures and assumptions for quantifying inputs are described below, along with an example problem that shows sample inputs for solving the problem with OPTUM.

3.1 NOTATION FOR THE SAMPLE PROBLEM

Mathematical notation is introduced in this report for the reader’s convenience in tracking how the data entries are used in value/utility calculations. This notation is not displayed on the OPTUM screens, but it is possible for the user to view intermediate results of the calculations by pressing the rows expand button “” when viewing the screen shown in Figure 2.1. Section 3.3 describes the detailed internal OPTUM calculations.

3.2 OBJECTIVES & PROJECTS TAB

The user can start by using the open folder button “” to open an existing sample portfolio case. Selecting a predefined case, such as “sample-case-1.xml,” can help the reader follow the instructions for constructing a new problem, as described in this section. The sample case can serve as a template that can be edited and customized to fit any new problem.

For those unfamiliar with establishing objectives and scales, Keeney (1992), Keeney and Raiffa (1993), and Hammond et al. (1998) serve as useful tutorials for setting up a new problem. The following example illustrates the mechanics of representing the objectives in OPTUM.

3.2.1 Entering Objective Names and Descriptions

This example uses project completion time, scope of benefits, and technical difficulty as the three objectives to be optimized. (Some other examples of objectives are costs, technical or financial uncertainties, expansion of production capability, uniqueness, or likelihood of attracting funding from outside sources.) It is important to recognize that costs can be entered and treated in two ways in OPTUM. One way is to include costs explicitly as an objective and trade them off against other objectives. This approach is recommended for problems in which cost is a valid and important consideration. In addition, to facilitate the construction of simple project rank-order listings in OPTUM screens, the expected costs for each project can be entered in the “cost” data field (shown later in this section). Even if cost estimates are entered explicitly as an objective, with the associated option of using probability distributions to capture potential uncertainties, the user is encouraged to also specify *expected* costs as an estimate in the cost data field.

Characterizing costs through objective definitions allows costs to be treated explicitly in the optimization process. In contrast, user entries in the simple cost data field are only partially recognized in the optimization process. They are used in summary screens for reference and sorting purposes and are recognized in cost *constraints* during optimization. However, the point estimates for costs do not contribute to the overall *objective function* to be maximized. As a guideline for choosing whether or not to create a separate objective to treat costs, the user should consider cases where two projects are otherwise equally valued according to the objectives. If the preferred portfolio choices favor the lower-cost project over the higher-cost project, the user should explicitly include costs in the objectives (in addition to entering the expected cost data). And if the preference for the lower-cost option is only slight, the relative weight for costs should be small in the objective function.

Figure 3.1 shows the objective names under the “Objectives” heading. The green background indicates which of the table fields are to be filled in by the user (either alphanumeric or numeric entries). Fields that have a blue background indicate that the user needs to make a multiple-choice selection. In addition, the screen provides a field for more lengthy descriptions of each objective. This extended description can be very useful for documenting the interpretation of objective definitions that are not obvious.

3.2.2 Defining Objective Scales: Scale Types and Units of Measure

For any objective, the user has a choice of using a continuous scale of measurement or a discrete “constructed” scale. The choice appears when the user right-clicks on the blue-highlighted scale-type field (Figure 3.2).

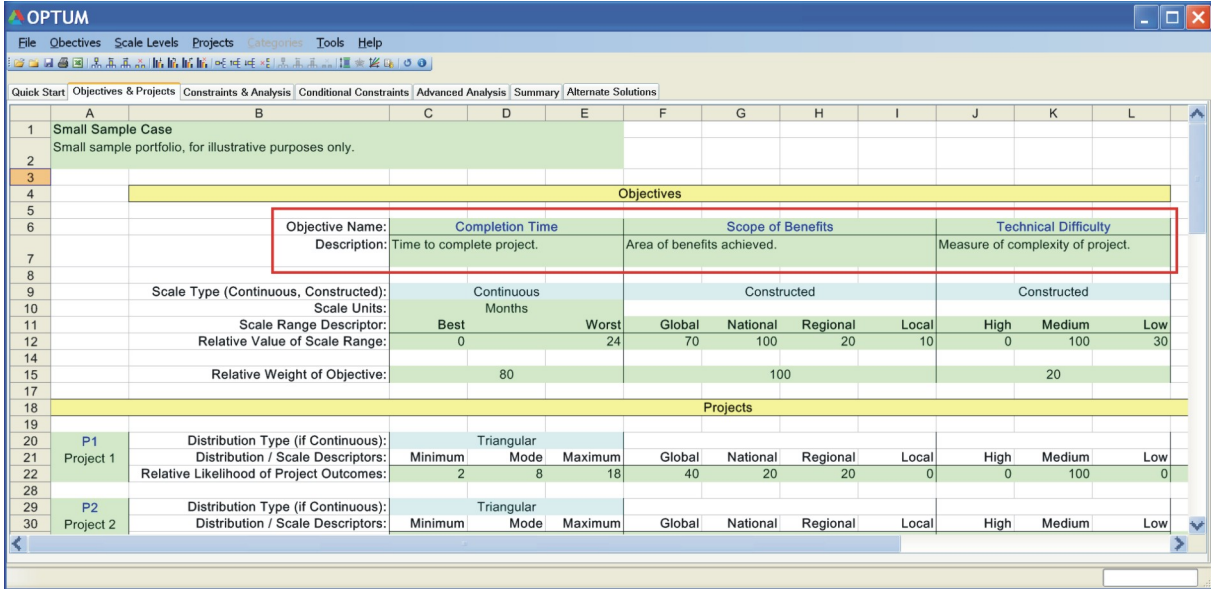


FIGURE 3.1 Objective Name and Description Fields

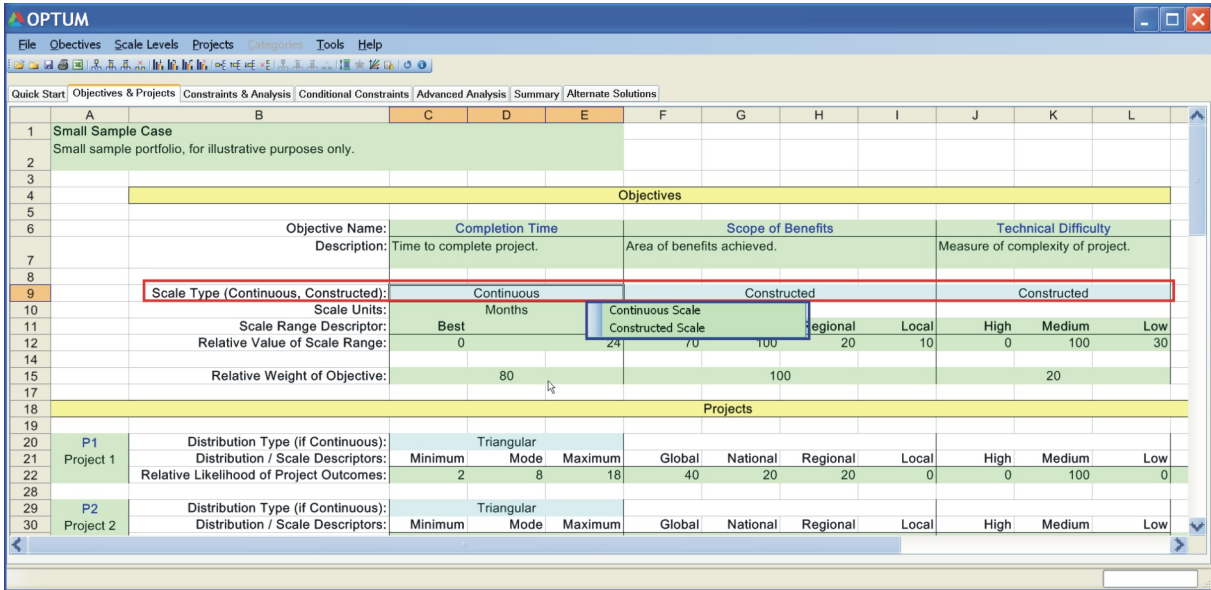


FIGURE 3.2 Objective Scale Types

An example of a continuous scale — as applied to the first objective — would be time expressed in some unit of measure, such as days or months. For this example, the time scale of months is used, and the range for that scale is 0–24 months. Alternatively, the completion time could have been represented in discrete “bins” (constructed scales). For example, four bins of 6 months each could cover the same time frame as that defined in the 0–24 month continuous scale. And when the constructed scale was used, the scale range descriptors would be changed to 0–6 months, 6–12 months, 12–18 months, and 18–24 months.

The choice between continuous and constructed scales is largely determined by user preference and how the objective is to be measured. If the user chooses a continuous scale for rating a particular objective, then a continuous probability distribution format will automatically be displayed under the corresponding project rating fields, and the user can then choose among eight different types of distributions. If the user chooses to use a constructed scale for rating an objective, then the proper number of bins will automatically be displayed under the corresponding project rating fields for that objective.

3.2.3 Defining Objective Scale Ranges

Once the scale type (and units of measure for continuous scales) is determined, as highlighted in Figure 3.2, the user enters the scale range descriptors (Figure 3.3). This entry is simply a label that serves as a reminder of what units of measure have been adopted. For continuous scales, OPTUM inserts this descriptor as the orientation for which end of the continuous range is considered best and which end of the range is considered worst. For constructed scales, the descriptors convey the range of each scale bin. For the example cited above for time (measured in months and separated into four discrete bins), the scale range

Row	Column	Content									
1	A	Small Sample Case									
2	A	Small sample portfolio, for illustrative purposes only.									
4	Objectives										
6	B	Objective Name: Completion Time									
6	D	Scope of Benefits									
6	J	Technical Difficulty									
7	B	Description: Time to complete project.									
7	D	Area of benefits achieved.									
7	J	Measure of complexity of project.									
9	B	Scale Type (Continuous, Constructed): Continuous									
9	D	Constructed									
9	J	Constructed									
10	B	Scale Units: Months									
11	B	Scale Range Descriptor: Best									
11	D	Worst									
11	F	Global									
11	G	National									
11	H	Regional									
11	I	Local									
11	K	High									
11	L	Medium									
11	M	Low									
12	B	Relative Value of Scale Range: 0									
12	D	24									
12	F	70									
12	G	100									
12	H	20									
12	I	10									
12	K	0									
12	L	100									
12	M	30									
15	B	Relative Weight of Objective: 80									
15	D	100									
15	J	20									
18	Projects										
20	A	P1									
20	B	Distribution Type (if Continuous): Triangular									
21	B	Minimum									
21	D	Mode									
21	F	Maximum									
21	G	Global									
21	H	National									
21	I	Regional									
21	J	Local									
21	L	High									
21	M	Medium									
21	N	Low									
22	B	Relative Likelihood of Project Outcomes: 2									
22	D	8									
22	F	18									
22	G	40									
22	H	20									
22	I	20									
22	J	0									
22	L	0									
22	M	100									
22	N	0									
29	A	P2									
29	B	Distribution Type (if Continuous): Triangular									
30	B	Minimum									
30	D	Mode									
30	F	Maximum									
30	G	Global									
30	H	National									
30	I	Regional									
30	J	Local									
30	L	High									
30	M	Medium									
30	N	Low									

FIGURE 3.3 Objective Scale Units of Measure, Scale Ranges, and Relative Values

descriptors would be 0–6 months, 6–12 months, 12–18 months, and 18–24 months. Or, as shown in Figure 3.3, the scale descriptors for the objective defined as “Scope of Benefits” are “Global,” “National,” “Regional,” and “Local.” A constructed scale can be divided into any number of bins.

3.2.4 Assigning Relative Values to Objective Scale Ranges

The final step for defining objective scales is to assign the relative values for each scale range. For continuous scales, the relative values are simply the endpoints of the distributions. In general, relative values are typically assumed to be linear and continuous over the range of the objective scale. Internally, OPTUM substitutes values in the range of 0–100 to complete the required normalizations and value calculations.

For continuous scales, OPTUM provides a mechanism to represent nonlinear risk preferences as they apply to interior points within the objective scale ranges. The derivations in Section 3.3.2 describe the use of an “intrinsic risk parameter” (r), which can be used to capture risk-averse and risk-tolerant preferences. Positive values of r (greater than zero) correspond to risk-averse attitudes, and negative values of r (less than zero) correspond to risk-tolerant attitudes. In practice, the value of r is assessed from how a decision maker responds to choices between a sure outcome and a specially constructed lottery. Their risk preferences are revealed through a sequence of alternative “fixed-outcome” versus “lottery” choices, and the value of r is derived from the responses. Once the value of r is determined, the user can input that value in the User Preferences menu (Section 4.5).

For constructed scales, the user must determine the values of each scale range *relative to the other scale ranges*. For example, under the “Scope of Benefits” objective, the user needs to assign relative values to each category (global, national, regional, or local). In the example, the categories have been “assessed” and reflect the highest value for national benefits (100%), second-highest value for global benefits (70%), third-highest value for regional benefits (20%), and lowest value for local benefits (10%). If the user has elected to apply an intrinsic risk parameter as mentioned above, that risk parameter does not apply to constructed scales because the decision maker’s risk preferences are captured directly in the value assessments for each scale range.

As shown by this example, the value ratings do not need to total 100%. And while any value (between 0 and 100%) can be entered for any of the constructed scales, *it is highly recommended that at least one of the scale values be defined by the user as 100%* (for the scale range that represents the best outcome). OPTUM performs the necessary normalizations internally before the final utility assessments or optimization is made (see Section 3.3). For convenience, a great deal of flexibility in assigning the relative values of scale ranges is allowed, but to avoid potential confusion, it is recommended that the user assign the value of 100% to the highest-valued scale range. This convention improves the intuition for assigning relative values to the other scale ranges. Also, as shown in Figure 3.3 under the objective category of “Technical Difficulty,” relative values of zero may be assigned to cases in which those outcomes are assessed to contribute no value to the overall project utility.

3.2.5 Assigning Relative Weights to Each Objective

Once the objective scales have been defined (in terms of types, units, ranges, and values), the user is able to assign relative weights for each of the objectives (outlined with blue box in Figure 3.3). The user can assign any number between 0 and 100 for any of the objectives, and the entries do not need to total 100%. However, *it is highly recommended that at least one of the objectives (the most valued) should be assigned a value of 100%*. OPTUM performs the necessary normalizations for later calculations. The convention of assigning 100% to the most-valued objective (as it did in case for assigning scale range values) improves the intuition for assigning relative weights for the other objectives.

The data entries described above are the final user inputs needed to characterize the problem objectives. The final stage of problem preparation involves rating each of the projects according to each of the objective scales. Those steps are described below in Sections 3.2.6 and 3.2.7.

3.2.6 Identifying and Costing the Projects





Figure 3.4 illustrates where the user enters a brief title/label and a description for each project. The title should be short enough (e.g., eight characters or fewer) to allow the compact notation to be readable without expanding the data cell and occupying a large portion of the viewable screen. The description field can be as long as needed but will collapse to a few lines under normal use. The purpose of these fields is to provide unique identifiers for each project and critical reference information for occasional reviews and reminders. The add button “”, move up button “”, move down button “”, and remove button “” are used to insert, move, or delete projects from the list of candidates.

Figure 3.4 also highlights the area where a user is allowed to enter estimated cost information for each project. As noted in Section 3.1, this cost entry is a point estimate and used for summary and constraint purposes. It is used in summary screens for sorting and is applied to total and project-categorization cost constraints but does not contribute to the objective function. A separate cost objective can be defined to explicitly treat and weight cost preferences in the objectives.

3.2.7 Rating Projects According to the Objectives

Figure 3.5 highlights the data entry fields used to evaluate each project with respect to the objectives. The three categories of inputs are (1) distribution type (if the objective has been defined by a continuous measurement scale), (2) distribution or scale descriptor, and (3) relative likelihood of outcomes for this project. These inputs provide all the information needed to complete the calculation of project utility, which is then used to optimize the selection of projects for a given portfolio.

Objectives

Objective Name:	Completion Time	Scope of Benefits	Technical Difficulty
Description:	Time to complete project.	Area of benefits achieved.	Measure of complexity of project.
Scale Type (Continuous, Constructed):	Continuous Scale	Constructed Scale	Constructed Scale
Scale Units:	Months		
Scale Range Descriptor:	Best	Worst	Global National Regional Local High Medium Low
Relative Value of Scale Range:	0	24	70 100 20 10 0 100 30
Relative Weight of Objective:	80		100 20

Projects

Project	Distribution Type (if Continuous):	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low	Cost	Utility	Selected?
P1 Project 1	Triangular	2	8	18	40	20	20	0	0	100	0	1000	0.66944444	
P2 Project 2	Triangular	4	6	24	60	100	20	20	20	100	80	900	0.64311111	
P3 Project 3	Triangular	5	6	24	60	100	20	20	20	100	80	800	0.63751805	✓
P4 Project 4	Triangular	6	6	24	60	100	20	20	20	100	80	780	0.63199875	
P5 Project 5	Triangular	7	8	24	60	100	20	20	20	100	80	950	0.61531165	
P6 Project 6	Triangular	9	10	24	60	100	20	20	20	100	80	600	0.59309029	✓
P7 Project 7	Triangular	18	20	22	10	20	20	50	100	40	10	400	0.27533333	

Total of All Projects: 5430 4.06580763
Total of Selected (✓) Projects: 1400 1.23060834 2

FIGURE 3.4 Entering Project Names and Descriptions

Projects

Project	Distribution Type (if Continuous):	Minimum	Mode	Maximum	Global	National	Regional	Local	High	Medium	Low
P1 Project 1	Triangular	2	8	18	40	20	20	0	0	100	0
P2 Project 2	Triangular	4	6	24	60	100	20	20	20	100	80

FIGURE 3.5 Project Ratings: Distribution Types, Scales, and Relative Likelihoods

The distribution type is selected from a dropdown menu of eight options (Figure 3.6): triangular, normal, lognormal, uniform, single value (fixed), geometric, exponential, and binomial distributions. Once the user selects one of the distribution options for a continuous scale objective, the OPTUM program prompts for the essential parameters that define the distribution. For example, as shown in Figure 3.5, after selecting the triangular distribution to represent “Completion Time” for Project 1, the user would be prompted for three parameters: minimum, mode, and maximum. The OPTUM program automatically inserts the headings (labels) for each parameter. For this example, the user would enter the number 2 for minimum, 8 for mode, and 18 for maximum. And the interpretation for these entries would be that the completion time for Project 1 would be expected to fall between 2 and 18 months, with a most likely outcome of 8 months.

The type of distribution can be different for each project, even for the same objective. For example, the user can choose to characterize the completion time for Project 1 with a triangular distribution, for Project 2 with a uniform distribution, and for Project 3 with a binomial distribution. This degree of freedom can help accommodate different types of information that might be available for different projects. Similarly, different objectives can be characterized for a given project with different types of distributions.

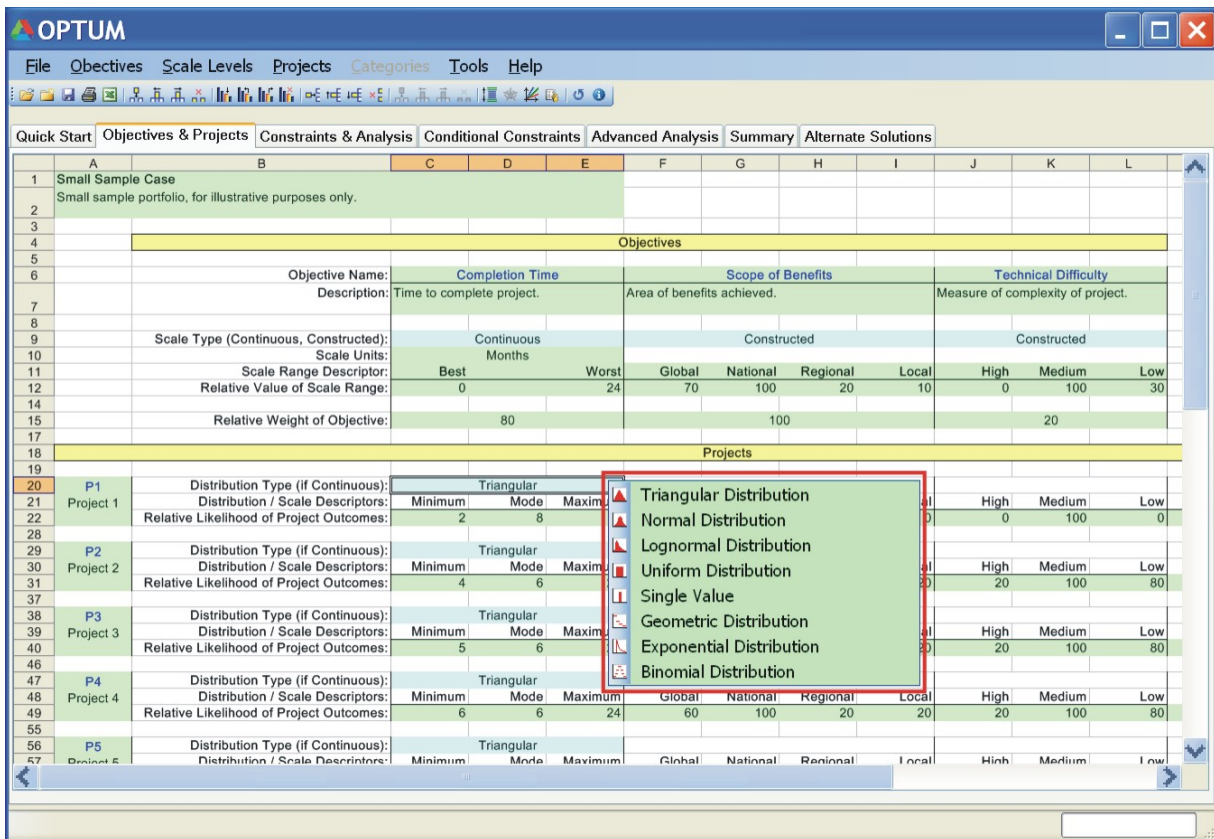







FIGURE 3.6 Distribution Types for Objectives with Continuous-scale Types

For objectives rated with constructed (discrete) scales, the data entries are flexible and straightforward. The user first determines how many scale levels are needed to characterize the objective. For example, the “Scope of Benefits” objective has been characterized by four scale levels (bins) corresponding to global, national, regional, and local benefits. If the existing template has too few columns to accommodate the number of scale levels needed, the user can apply the add column button “” to add columns. Alternatively, if there are too many columns, the user can apply the remove column button “” to remove columns. The move column left button “” and move column right button “” can be used to move existing scale levels to alternate locations.

Once the desired number of columns is arranged, the user provides labels to describe the scale level. So for “Scope of Benefits,” the labels are “Global,” “National,” “Regional,” and “Local.” And for “Technical Difficulty,” the labels are “High,” “Medium,” and “Low.” To rate each project according to these objective scales, the user provides numerical inputs that characterize the anticipated outcomes in the “Relative Likelihood of Project Outcomes” row. The relative likelihoods of project outcomes can be input as any number between 0 and 100, and they do not need to total 100. The OPTUM program provides normalizations for the relative likelihood inputs, so they can be properly weighted in the utility calculations.

Figure 3.7 shows where the user has an opportunity to manually select specific projects to be included in a portfolio (outlined with red box). As projects are selected, the sum of overall utility (as derived in Section 3.3) and costs are displayed at the bottom of the screen (outlined with blue box). As noted earlier, utility represents a single measure of merit that combines all of the estimated outcomes and relative importance factors as assessed by the user.

The total utility and costs for *all* projects are also displayed for reference. More sophisticated manipulations and displays of utilities, costs, and constraints are shown on the next tab (Constraints & Analysis) and described in Section 4. But for a simple one-page review and summary of the candidate projects, the screen shown in Figure 2.1 provides a useful display of characteristic information. As noted earlier, more detailed summary information can be viewed on the basic input screen by pressing the reveal button “” to reveal the intermediate parameter calculations and final utility results (Figure 3.8).

In terms of internal calculations, Section 3.3.2 describes the derivations used to determine overall project values and utilities. For the average user, these internal calculations can generally be ignored, and the analysis may proceed without user intervention. But for advanced users, an option is presented in the User Preferences menu (Section 4.5) to choose Monte Carlo sampling in place of analytical solutions for calculating project values over the ranges of user-specified scale ranges (including objectives with either continuous scales or constructed scales).

Whereas default analytical solutions for continuous scales are obtained through piece-wise trapezoidal approximations to the interior function values, the user may elect to use Monte Carlo samplings (from the User Preferences menu) to override the derivations of objective-level project values. For Monte Carlo trial solutions, any number of trials may be specified from the User Preferences menu. For default analytical solutions, the number of piece-wise segments is

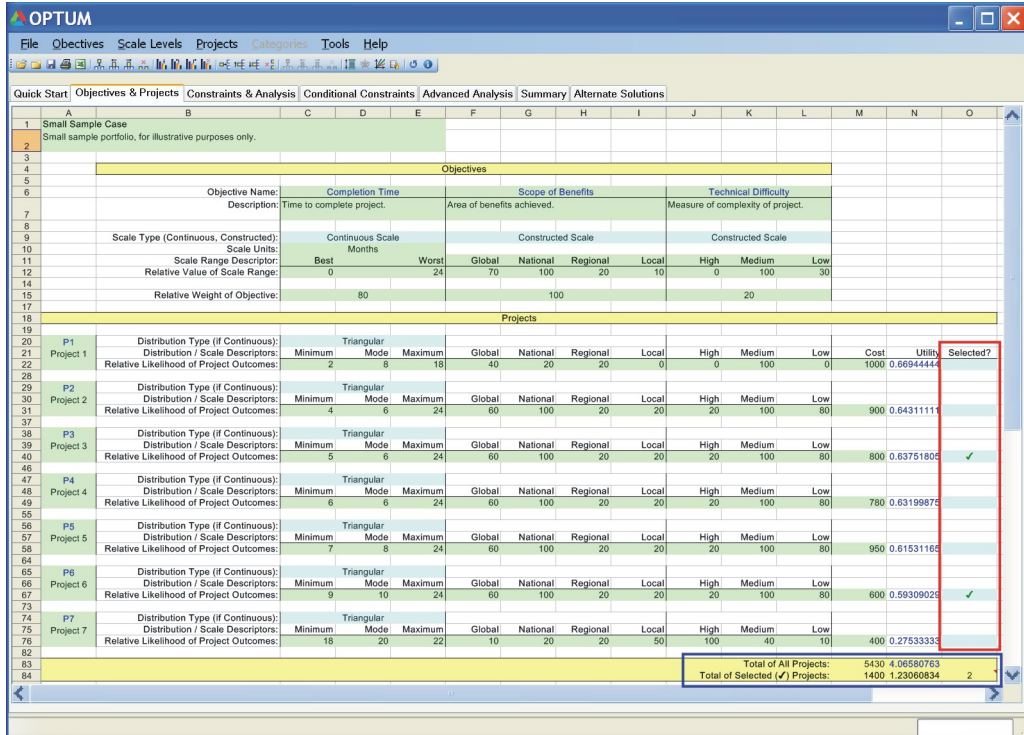


FIGURE 3.7 Manual Selection of Projects for a Portfolio

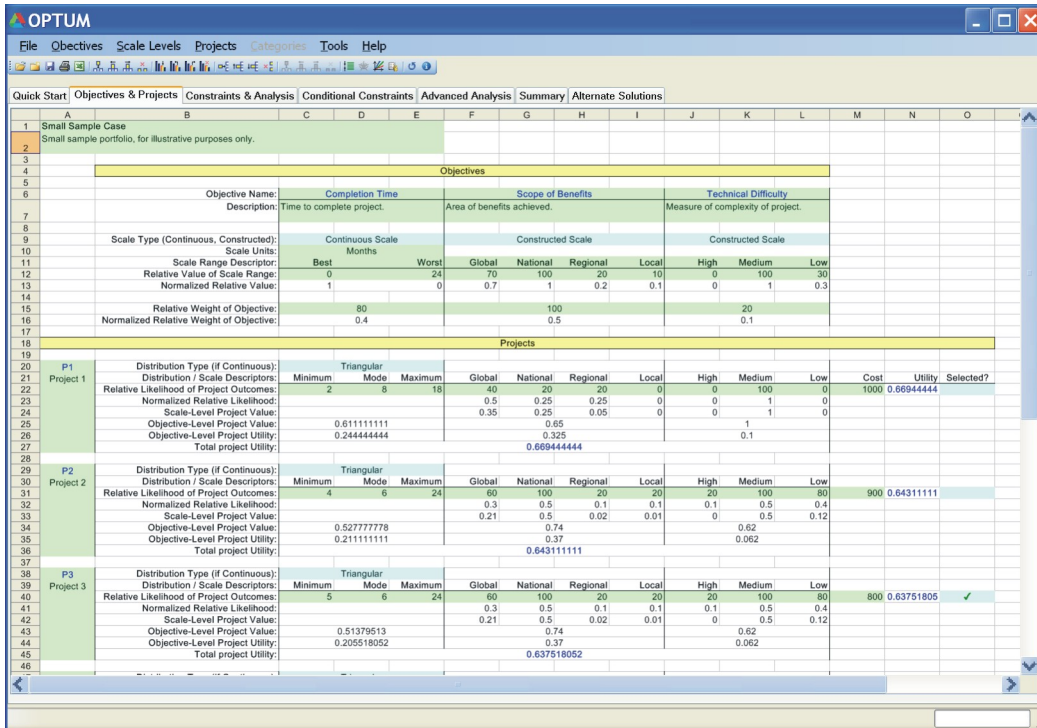


FIGURE 3.8 Summary Screen with Intermediate Calculation Results Displayed

fixed at 200 points for accurate and efficient calculations (only used in derivations for continuous scales).

3.3 PROJECT VALUE AND UTILITY DERIVATIONS FOR THE SAMPLE PROBLEM

This section shows an overview of how project values and utilities are derived from user inputs, and it also provides the detailed mathematical derivations for these values (Section 3.3.2). In Table 3.1, the column farthest to the left contains a label used for reference purposes. That same label is used for shorthand notation in the column farthest to the right, showing the source of derivations and intermediate values.

3.3.1 Overview of Sample Problem Derivations

Table 3.1 is visually different from the screen display shown in Figure 3.8, but the two are parallel in terms of the example portrayed and the user-defined inputs. Table 3.1 can be used to trace the intermediate calculations used to derive project and portfolio utility values.

For example, to derive the values used in row X6 (Normalized Relative Value of Scale Range: \overline{RVSR}_{ij}), Table 3.1 shows that those values equal the entries in row X5 (Relative Value of Scale Range: $RVSR_{ij}$) divided by 100. And to derive the values in row X8, each entry in row X7 is divided by the sum of all entries in row X7 ($X8 = X7 \div \sum X7$).

All of the entries in Table 3.1 are similarly derived, and the end result is a translation of user inputs, first expressed and input as *estimated outcomes for each project* (i.e., scale-level project values), into *total project utilities*. In this context, utility represents a single measure of merit that combines all of the estimated outcomes and relative importance factors as assessed by the user. The total project utilities are used to compare, rank, and select projects on a uniform and comparable basis.

Table 3.1 includes entries for three hypothetical projects, and the derivation of results can be followed to the completion of total project utility results (X14) for each of the projects. Table 3.2 summarizes the results from Table 3.1, and the entries in this table serve to initiate the portfolio optimization process. Alternatively, these values can be used to rank and guide manual selections to construct a user-preferred portfolio.

3.3.2 Detailed OPTUM Derivations

The mathematical derivations of the results for Tables 3.1 and 3.2 are shown here in greater detail. For this section, the following notation and conventions are used: N = number of projects, n = project number index, J = number of objectives, j = objective number index, I_j = number of objective scale ranges for Objective j ($j = 1, 2, \dots, J$), and i = scale range index.

TABLE 3.1 Sample Problem and Internal OPTUM Calculations

Label	Variable	Values										Source			
A. Define/Name Objectives (example below with three objectives [J=3])															
X1	Objective Number and Name (j); (ON _j)	1. Completion Time			2. Scope of Benefits				...	3(=J). Technical Difficulty			User		
B. Define Scale Ranges and Types (for Measuring Objective "j") and Assess Relative Value of Each Scale Range															
X2	Scale Type	Continuous/"Natural"			Discrete/"Constructed"				...	Discrete/"Constructed"			User		
X3	Scale Range Index (i)	1	2 (=I ₁)		1	2	3	4 (=I ₂)		...	1	2	3 (=I _j)		User
X4	Scale Range Descriptor (SRD _{ij})	0 months (SRmin ₁)		24 months (SRmax ₁)		Global	National	Regional	Local	...	high	med	low		User
X5	Relative Value of Scale Range (RVSR _{ij})	100 (best)		0 (worst)		70	100 (best)	20	10 (worst)	...	0 (worst)	100 (best)	30		User (≥0 & ≤100 & must include 100)
X6	Normalized Relative Value of Scale Range (RVSR _{ij})	1.0		0.0		0.700	1.000	0.200	0.100	...	0.000	1.000	0.300		Eq. 1 (≥0 & ≤1.0) X5÷100
C. Assign Relative Weights for Each Objective															
X7	Relative Weight of Objective (RWO _j)	80			100				...	20			User (≥0 & ≤100 & should include 100)		
X8	Normalized Relative Weight of Objective (RWO _j)	0.400			0.500				...	0.100			Eq. 2 sum to 1 X7÷(∑X7)		
D.1 Assess Relative Likelihood of Outcomes (for Project "1")															
X9	Relative Likelihood of Project Outcomes (RLPO _{nij})	2 mo. [min.]	8 mo. [mode]	18 mo. [max.]	40	20	20	0	...	0	100	0		User (≥0 & ≤100)	
X10	Normalized Relative Likelihood of Project Outcomes (RLPO _{nij})	0.000	0.125	0.000	0.500	0.250	0.250	0.000	...	0.000	1.000	0.000		Eq. 3a/b sum to 1 for discrete X9÷(∑X9)	
		(triangular probability distribution)			(discrete probability distribution)						(discrete probability distribution)				
E.1 Determine Values and Utilities for Project "1" (Scale-level, Objective-level, and Total)															
X11	Scale-level Project Value (SLPV _{nij})	N/A	N/A	N/A	0.350	0.250	0.050	0.000	...	0.000	1.000	0.000		Eq. 4 X6×X10 for Discrete	
X12	Objective-level Project Value (OLPV _{nj})	0.611 (area under (X6 • X10))			0.650 (∑ (X10))				...	1.000 (∑ (X10))			Eq. 5		
X13	Objective-level Project Utility (OLPU _{nj})	0.244			0.325				...	0.100			Eq. 6 X8•X12		
X14	Total Project Utility (TPU _n)	0.669										Eq. 7 ∑ _j (X13)			

TABLE 3.1 (Cont.)

Label	Variable	Values											Source
D.2 Assess Relative Likelihood of Outcomes (for Project “2”)													
X9	Relative Likelihood of Project Outcomes (RLPO _{nij})	4 mo. [min.]	6 mo. [mode]	24 mo. [max.]	60	100	20	20	...	20	100	80	User (≥0 & ≤100)
X10	Normalized Relative Likelihood of Project Outcomes (RLPO _{nij})	0.000	0.100	0.000	0.300	0.500	0.100	0.100	...	0.100	0.500	0.400	Eq. 3a/b sum to 1 for discrete X9+(ΣX9)
		(triangular probability distribution)			(discrete probability distribution)					(discrete probability distribution)			
E.2 Determine Values and Utilities for Project “2” (Scale-level, Objective-level, and Total)													
X11	Scale-level Project Value (SLPV _{nij})	N/A	N/A	N/A	0.210	0.500	0.020	0.010	...	0.000	0.500	0.120	Eq. 4 X6-X10 for discrete
X12	Objective-level Project Value (OLPV _{nj})	0.528 (area under (X6 • X10))			0.740 (Σ _i (X10))				...	0.620 (Σ _i (X10))			Eq. 5
X13	Objective-level Project Utility (OLPU _{nj})	0.211			0.370				...	0.062			Eq. 6 X8-X12
X14	Total Project Utility (TPU _n)	0.643											Eq. 7 Σ _i (X13)
... (assessments for other projects 3, 4, ... [N - 1]) ...													
D.N Assess Relative Likelihood of Outcomes (for Project “N”)													
X9	Relative Likelihood of Project Outcomes (RLPO _{nij})	18 mo. [min.]	20 mo. [mode]	22 mo. [max.]	10	20	20	50	...	100	40	10	User (≥0 & ≤100)
X10	Normalized Relative Likelihood of Project Outcomes (RLPO _{nij})	0.000	0.500	0.000	0.100	0.200	0.200	0.500	...	0.667	0.267	0.067	Eq. 3a/b sum to 1 for discrete X9+(ΣX9)
		(triangular probability distribution)			(discrete probability distribution)					(discrete probability distribution)			
E.N Determine Values and Utilities for Project “N” (Scale-level, Objective-level, and Total)													
X11	Scale-level Project Value (SLPV _{nij})	N/A	N/A	N/A	0.070	0.200	0.040	0.050	...	0.000	0.267	0.020	Eq. 4 X6-X10 for discrete
X12	Objective-level Project Value (OLPV _{nj})	0.167 (area under (X6 • X10))			0.360 (Σ _i (X10))				...	0.287 (Σ _i (X10))			Eq. 5
X13	Objective-level Project Utility (OLPU _{nj})	0.067			0.180				...	0.029			Eq. 6 X8-X12
X14	Total Project Utility (TPU _n)	0.276											Eq. 7 Σ _i (X13)

TABLE 3.2 Evaluation and Optimization of Portfolio

Project No.	Project Name	Risk Neutral ($r = 0.0$)	Source
1	ABC	0.669	Eq. 7 (X14)
2	DEF	0.643	Eq. 7 (X14)
NSP ^a	XYZ	0.276	Eq. 7 (X14)
Total utility (TU) of portfolio	(X15)	1.588	Eq. 8 \sum_n (X14)

^a Number of selected projects (selected manually or through optimization for inclusion in portfolio). Note: NSP = N if all projects are selected, and NSP = 3 in this simple example.

Row X1

J = number of objectives (user-defined),

j = objective number index, and

ON_j = objective name for Objective j (user-defined).

Row X2

ST_j = scale type for Objective j ($j = 1, 2, \dots, J$) (continuous or discrete) (user-defined).

Definition: This is the type of scale used for evaluating each project with respect to Objective j . Continuous scales are typically used for natural scales, such as costs (e.g., \$) or time (e.g., days/months). Discrete scales are typically applied to constructed scales, such as a technical challenge (e.g., high/medium/low) or a benefit (high/medium-high/average/medium-low/low).

Row X3

I_j = number of scale ranges for Objective j ($j = 1, 2, \dots, J$) (user-defined) and

i = scale-range index.

Row X4

SRD_{ij} = scale-range descriptor for Scale Range i of Objective j ($j = 1, 2, \dots, J$; $i = 1, 2, \dots, I_j$) (user-defined).

For continuous/natural scales:

$SRmin_j$ = scale range minimum for Objective j ($j = 1, 2, \dots, J$) (user-defined) and

$SRmax_j$ = scale range maximum for Objective j ($j = 1, 2, \dots, J$) (user-defined).

Row X5

$RVSR_{ij}$ (user assessment)

X5a — For discrete/constructed scales:

$RVSR_{ij}$ = relative value of Scale Range i (0–100) for Objective j ($j = 1, 2, \dots, J$; $i = 1, 2, \dots, I_j$).

Definition: For Objective j , user-assessed relative value of Scale Range i outcome. At least one of the scale ranges should be assigned a relative value of 100 (best outcome).

X5b — For continuous/natural scales:

$RVSR_{ij}$ = relative value of outcomes at scale range endpoints $SRmin_j$ and $SRmax_j$ (0 or 100) for Objective j ($j = 1, 2, \dots, J$; $i = 1$ for $SRmin_j$ and 2 for $SRmax_j$).

Definition: For Objective j , user assesses the best (100) and worst (0) outcomes. $RVSR_{1j}$ should be assigned a relative value of 0 or 100, corresponding to whether the value associated with outcome $SRmin_j$ is worst or best; and $RVSR_{2j}$ should be assigned the opposite relative value (100 or 0), corresponding to whether the value associated with $SRmax_j$ is best or worst.

Row X6

\overline{RVSR}_{ij} = normalized relative value of scale range (0–1.0) for Scale Range i of Objective j ($j = 1, 2, \dots, J$; $i = 1, 2, \dots, I_j$).

Definition: For each Objective j , $RVSR_{ij}$ values scaled/normalized to span range of 0.0–1.0 for each Scale Range i ($i = 1, 2, \dots, I_j$).

$\overline{RVSR}_{ij} = RVSR_{ij}/100$ for each Objective j ($j = 1, 2, \dots, J$; $i = 1, 2, \dots, I_j$). (Eq. 1)

Note: For cases where the user has specified a nonzero “intrinsic risk parameter” (in the User Preferences menu), the interior values of \overline{RVSR}_{ij} , between endpoints of 0.0 and 1.0, are adjusted to be nonlinear, reflecting risk-averse or risk-tolerant preferences. Assigning a positive value (greater than zero) to the intrinsic risk parameter (r) corresponds to risk-averse preferences, and assigning a negative value (less than zero) to the parameter corresponds to risk-tolerant preferences. An r value of 0.0 represents risk-neutral preferences, and the function reverts to a standard linear interpolation between the lower and upper bounds. The interior points z' are derived as follows:

$$\begin{aligned} \overline{RVSR}_{ij}(z') &= [1 - \exp(-rz')] / [1 - \exp(-r)] && \text{(when } \overline{RVSR}_{1j} < \overline{RVSR}_{2j} \text{) [desirable]} \\ \overline{RVSR}_{ij}(z') &= [1 - \exp(-r\{1-z'\})] / [1 - \exp(-r)] && \text{(when } \overline{RVSR}_{1j} > \overline{RVSR}_{2j} \text{) [undesirable]} \end{aligned}$$

where z is a continuous variable $SR_{minj} \leq z \leq SR_{maxj}$,
and $z' = (z - SR_{minj}) / (SR_{maxj} - SR_{minj})$;
therefore z' is a normalized continuous variable $0 \leq z' \leq 1$.

Row X7

RWO_j = relative weight of objective (0–100) for Objective j ($j = 1, 2, \dots, J$)
(user assessment).

Definition: User-assessed relative weight for each Objective j as it contributes to overall value of portfolio of projects. One of the relative weights for each objective should be assigned a relative value of 100 (most important objective).

Row X8

\overline{RWO}_j = normalized relative weight of objective for Objective j ($j = 1, 2, \dots, J$).

Definition: For all objectives, RWO_j scaled to sum to 1.0 over j ($j = 1, 2, \dots, J$).

$$\overline{RWO}_j = RWO_j / (\sum RWO_j \text{ [summed over } j = 1, 2, \dots, J]) \text{ for each } j \text{ (} j = 1, 2, \dots, J \text{).} \quad (\text{Eq. 2})$$

Row X9

$RLPO_{nij}$ (user assessment)

X9a — For discrete/constructed scales:

$RLPO_{nij}$ = relative likelihood of project outcomes (0–100) for Project n , Scale Range i , of Objective j ($j = 1, 2, \dots, J$; $i = 1, 2, \dots, I_j$; $n = 1, 2, \dots, N$).

Definition: User-assessed relative likelihood of Project n resulting in Scale Level i outcome for Objective j. Assessed values do not need to sum to 100 (or any other number). OPTUM will normalize assessments in \overline{RLPO}_{nij} to sum to 1.0.

X9b — For continuous/natural scales (this example is for triangular distribution):

$RLPO_{nij}$ = relative likelihood distribution (minimum, mode, and maximum) of project outcomes (min, mode, max) for Project n, Objective j ($j = 1, 2, \dots, J$; $n = 1, 2, \dots, N$; $i = 1 \{\text{min}\}, 2 \{\text{mode}\}, 3 \{\text{max}\}$), where min, mode, and max are located within the range of ($SR_{minj} - SR_{maxj}$):

Minimum = $RLPO_{n1j} \geq SR_{minj}$;

Mode = $RLPO_{n2j}$, and $RLPO_{n1j} \geq RLPO_{n2j} \leq RLPO_{n3j}$;

Maximum = $RLPO_{n3j} \leq SR_{maxj}$.

Definition: User-assessed distribution (triangular in this example) of likely outcomes for Project n, Objective j, where $i = 1 \{\text{min}\}, i = 2 \{\text{mode}\}, i = 3 \{\text{max}\}$.

Row X10

\overline{RLPO}_{nij}

X10a — For discrete/constructed scales:

\overline{RLPO}_{nij} = normalized relative likelihood of project outcomes (0–1.0) for Project n, Scale Range i, of Objective j ($j = 1, 2, \dots, J$; $i = 1, 2, \dots, I_j$; $n = 1, 2, \dots, N$).

Definition: For each Project n and Objective j, $RLPO_{nij}$ values scaled to sum to 1.0 over ($i = 1, 2, \dots, I_j$).

$\overline{RLPO}_{nij} = RLPO_{nij} / (\sum RLPO_{nij} [\text{summed over } i = 1, 2, \dots, I_j])$ for each $j = 1, 2, \dots, J$;
 $n = 1, 2, \dots, N$. (Eq. 3a)

X10b — For continuous/natural scales (triangular distribution in this example):

\overline{RLPO}_{nij} = normalized relative likelihood distribution of project outcomes for Project n, Objective j ($j = 1, 2, \dots, J$; $n = 1, 2, \dots, N$).

Definition: For each Project n and Objective j, \overline{RLPO}_{n2j} value (triangular distribution height at $RLPO_{n2j}$) calculated so area under triangular distribution between $RLPO_{n1j}$ and $RLPO_{n3j} = 1.0$. (OPTUM determines curve height at mode, so area under triangular distribution equals 1.0.)

$\overline{RLPO}_{nij} = \overline{RLPO}_{n1j} = 0$.

$$\overline{RLPO}_{nij} = \overline{RLPO}_{n2j} = 2/(\overline{RLPO}_{n3j} - \overline{RLPO}_{n1j}) \text{ for each } j = 1, 2, \dots, J; \\ n = 1, 2, \dots, N. \quad (\text{Eq. 3b})$$

$$\overline{RLPO}_{nij} = \overline{RLPO}_{n3j} = 0.$$

Note: In this example, linear segments fill in triangular distribution between \overline{RLPO}_{n1j} , \overline{RLPO}_{n2j} , and \overline{RLPO}_{n3j} . For continuous scale distributions, analytic functions define the curve points needed in the derivations for Row X12.

Row X11

$$SLPV_{nij}$$

X11a — For discrete/constructed scales:

$$SLPV_{nij} = \text{scale-level project value for Project } n, \text{ Scale Range } i, \text{ of Objective } j \\ (j = 1, 2, \dots, J; i = 1, 2, \dots, I_j; n = 1, 2, \dots, N).$$

Definition: For Project n, the estimated value contribution for each Scale Range i of each Objective j, derived by weighting the normalized relative likelihood of project outcomes (assessed by scale range) by the normalized relative values of each scale range.

$$SLPV_{nij} = (\overline{RLPO}_{nij}) (\overline{RVSR}_{ij}) \text{ for } j = 1, 2, \dots, J; i = 1, 2, \dots, I_j; n = 1, 2, \dots, N). \quad (\text{Eq. 4a})$$

(X11b) — For continuous/natural scales:

It is not meaningful to display distribution results at scale level. Values are calculated internally in OPTUM for entire range of outcome distributions by multiplying \overline{RLPO}_{nij} distribution by \overline{RVSR}_{ij} distribution. These intermediate results are used to derive the subsequent values of $OLPV_{nj}$ as noted below.

$$SLPV_{nij} = (\overline{RLPO}_{nij}) (\overline{RVSR}_{ij}) \text{ for } j = 1, 2, \dots, J; n = 1, 2, \dots, N. \quad (\text{Eq. 4b})$$

Row X12

$$OLPV_{nj}$$

X12a — For discrete/constructed scales:

$$OLPV_{nj} = \text{objective-level project value for Project } n, \text{ Objective } j (j = 1, 2, \dots, J; \\ n = 1, 2, \dots, N).$$

Definition: For each Project n, the estimated value contribution for each Objective j, derived by summing scale-level project values for all scale ranges in the objective.

$$OLPV_{nj} = \sum_{n=1, 2, \dots, N} (SLPV_{njj} \text{ [summed over } i = 1, 2, \dots, Ij]) \text{ for } j = 1, 2, \dots, J; \quad (\text{Eq. 5a})$$

X12b — For continuous/natural scales:

$$OLPV_{nj} = \text{objective-level project value for Project } n, \text{ Objective } j \text{ (} j = 1, 2, \dots, J; n = 1, 2, \dots, N).$$

Definition: For each Project n , the estimated value contribution for each Objective j , derived by integrating scale-level project values over entire scale range of the objective.

$$OLPV_{nj} = \int (SLPV_{njj} \text{ [integrated over } RLPO_{n1j} \text{ to } RLPO_{n3j}]) \text{ for } j = 1, 2, \dots, J; n = 1, 2, \dots, N. \quad (\text{Eq. 5b})$$

Note: For continuous scales, these integrations are made numerically, with piece-wise trapezoidal approximations for the integrals of the continuous $SLPV_{njj}$ functions (not needed for constructed scale derivations). The number of piece-wise segments is fixed at 200.

In Monte Carlo mode, random draws are made over the distribution ranges to derive estimates for $OLPV_{nj}$ (for both continuous and constructed scales). The User Preferences menu allows the user to specify any number of Monte Carlo samplings to be made when the Monte Carlo mode is being used.

Row X13

$$OLPU_{nj} = \text{objective-level project utility for Project } n, \text{ Objective } j \text{ (} j = 1, 2, \dots, J; n = 1, 2, \dots, N).$$

Definition: For Project n , the estimated utility contribution of each Objective j , derived by multiplying objective-level project values by the normalized relative weight of each Objective j .

$$OLPU_{nj} = (OLPV_{nj}) (\overline{RWO}_j) \text{ for Objective } j \text{ (} j = 1, 2, \dots, J; n = 1, 2, \dots, N). \quad (\text{Eq. 6})$$

Row X14

$$TPU_n = \text{total project utility for Project } n \text{ (} n = 1, 2, \dots, N).$$

Definition: For Project n , the total estimated utility contributions from all J objectives, derived by summing objective-level project utilities over all objectives.

$$TPU_n = \sum OLPU_{nj} \text{ [summed over } j = 1, 2, \dots, J] \text{ for } n = 1, 2, \dots, N. \quad (\text{Eq. 7})$$

NSP = number of selected projects (manually or through optimization) to be included in portfolio.

Row X15

TU = total utility for all projects, or all “selected” projects, in portfolio.

Definition: The total estimated utility contributions from all N projects in a portfolio or all NSP-selected projects in a portfolio, derived by summing the total project utilities over all or selected projects.

$$TU = \sum TPU_n \text{ [summed over } n = 1, 2, \dots N \text{ or } n = 1, 2, \dots \text{NSP]}. \quad (\text{Eq. 8})$$

4 CONSTRAINTS AND ANALYSIS

This section describes the capabilities and options for introducing user-defined constraints to address portfolio optimization problems. It also provides an overview of how to graph and interpret the results.

4.1 CONSTRAINTS & ANALYSIS TAB

Once all the objectives and projects are defined, the Constraints & Analysis Tab can be used to proceed with in-depth analysis and optimization. The user can manually select projects for a portfolio and perform limited analysis from the Objectives & Projects Tab, but the Constraints & Analysis Tab opens up many more possibilities for the analysis. Figure 4.1 shows the basic screen layout for the Constraints & Analysis Tab. The portion of this screen outlined in red shows a basic list of the projects, including summaries for total utility, cost, and

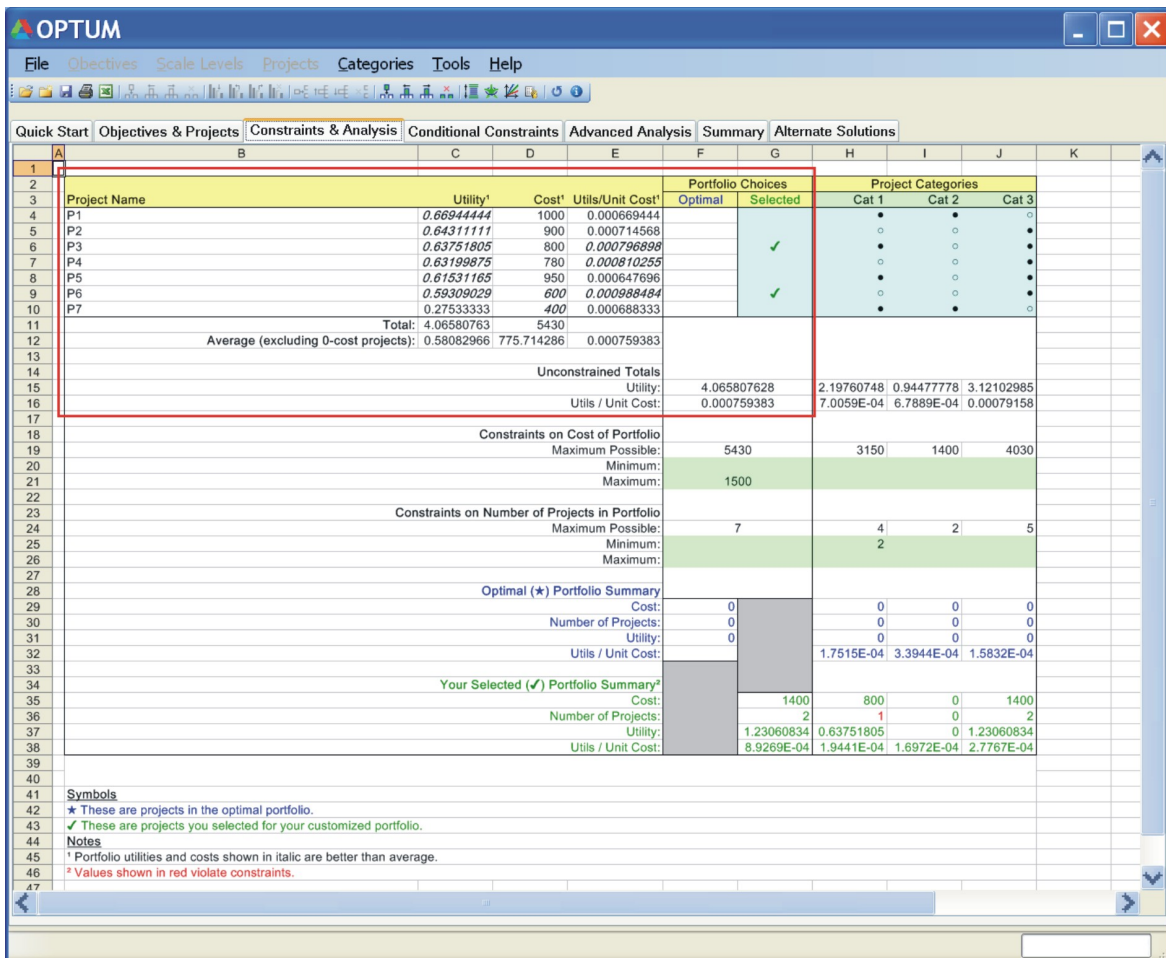


FIGURE 4.1 Capabilities Available from Constraints & Analysis Screen

utility per unit-cost (utils/unit-cost). The relative value of different projects can be assessed from the utils/unit-cost parameter, without consideration of resource or cost constraints or of inclusion or exclusion requirements. This metric provides a general indicator of which projects should be selected first to maximize the total expected value of a portfolio. However, when user-defined constraints are introduced, this simple approach to rank-ordering projects typically fails to produce the constrained optimal solution. Section 4.1.2 describes how the project categorizations can be used to implement user-defined constraints.

The blue-background cells near the top of Figure 4.1, under the heading “Portfolio Choices/Selected,” show where the user can click to select (✓) or unselect () projects to be included in a portfolio. As projects are included or excluded (selected or unselected), the totals under each column at the bottom of the screen (displayed in green lettering) change to reflect new costs and utilities.

4.1.1 Portfolio Optimization with Total Portfolio Constraints

Figure 4.2 highlights parameter fields that can be used effectively to optimize portfolio selections by using total-portfolio constraints (non-project-specific constraints). The highlighted columns show:

- Cost constraints (maximum and/or minimum),
- Number of projects constraints (maximum and/or minimum),
- Results for optimized portfolio (cost, number of projects, utility, and utils/unit-cost), and
- Results for manually selected portfolio (cost, number of projects, utility, and utils/unit-cost).

The green-background cells, shown in the featured areas of Figure 4.2, indicate where the user has opportunities to define constraints (on total costs and/or numbers of projects to be selected). For each of these data entry fields, the potential maximums are shown for reference; they are based on the projects that have been defined. If the user enters limits that are outside the range of [0 – potential maximum], OPTUM will reset the entry to the maximum or minimum possible value. For example, if the user enters \$6000K (i.e., \$6 million) for the maximum cost constraint, OPTUM will reset the entry to \$5430K, which corresponds to the total cost of all possible projects that have been defined.

After setting the cost and number of projects constraints, the user can activate the optimization function by pressing the optimization button “★”. Note that the “★” icon replaces the “★” icon as an indication that constraints are to be observed during the optimization process. To ignore any constraints that have been specified without deleting them from the data entry cells, the user can bring up the User Preferences menu (☰) and check or uncheck the appropriate boxes for constraints to be observed or ignored (Section 4.5). The example in

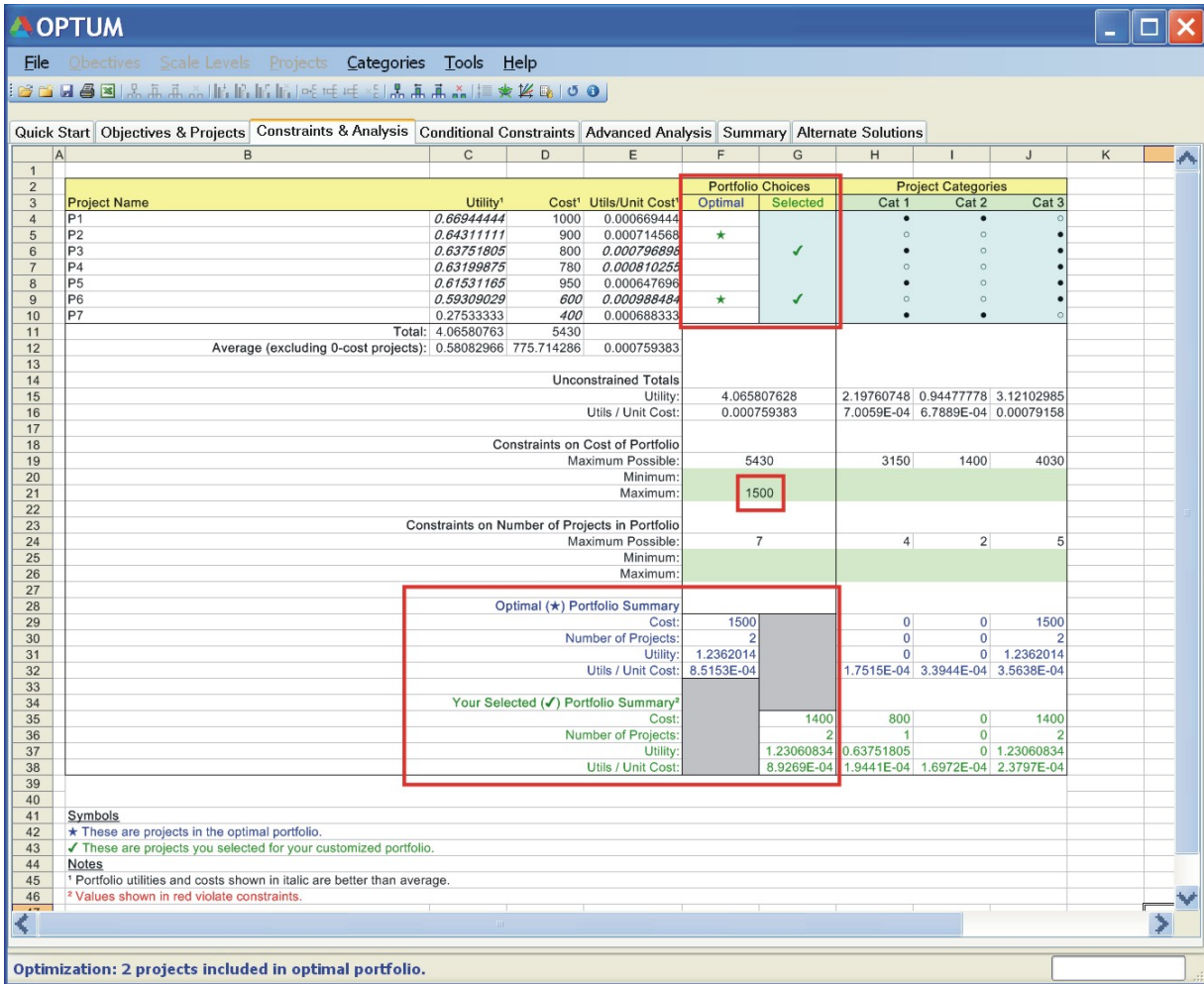


FIGURE 4.2 Optimized Results with Cost Constraints

Figure 4.2 shows a case where the maximum cost has been set at \$1500K (out of a possible maximum of \$5430K).

The final optimized results are displayed in Figure 4.2. The portfolio consists of two projects, P2 + P6, with a total utility value of 1.236 and total cost of \$1500K. These results are shown side by side with the user-selected combination of projects P3 + P6, which yielded a lower utility value of 1.231 and totaled \$1400K in cost.

In this simple example, the optimal value may not be difficult to find manually, but if the upper bound on costs is higher, the task becomes more difficult. With a maximum cost limit of \$3000K, a user might select a combination of P1 + P5 + P6 + P7, with a total utility of 2.153 and cost of \$2950K. Or the user might locate a better solution, with higher utility and lower cost, by selecting P2 + P3 + P4 + P7, resulting in a utility of 2.188 and cost of \$2880K. Still, the solution can be further improved as shown with the optimized solution of P1 + P3 + P4 + P7, with a total utility of 2.214 and cost of \$2980K (Figure 4.3). Larger problems with more complex constraints

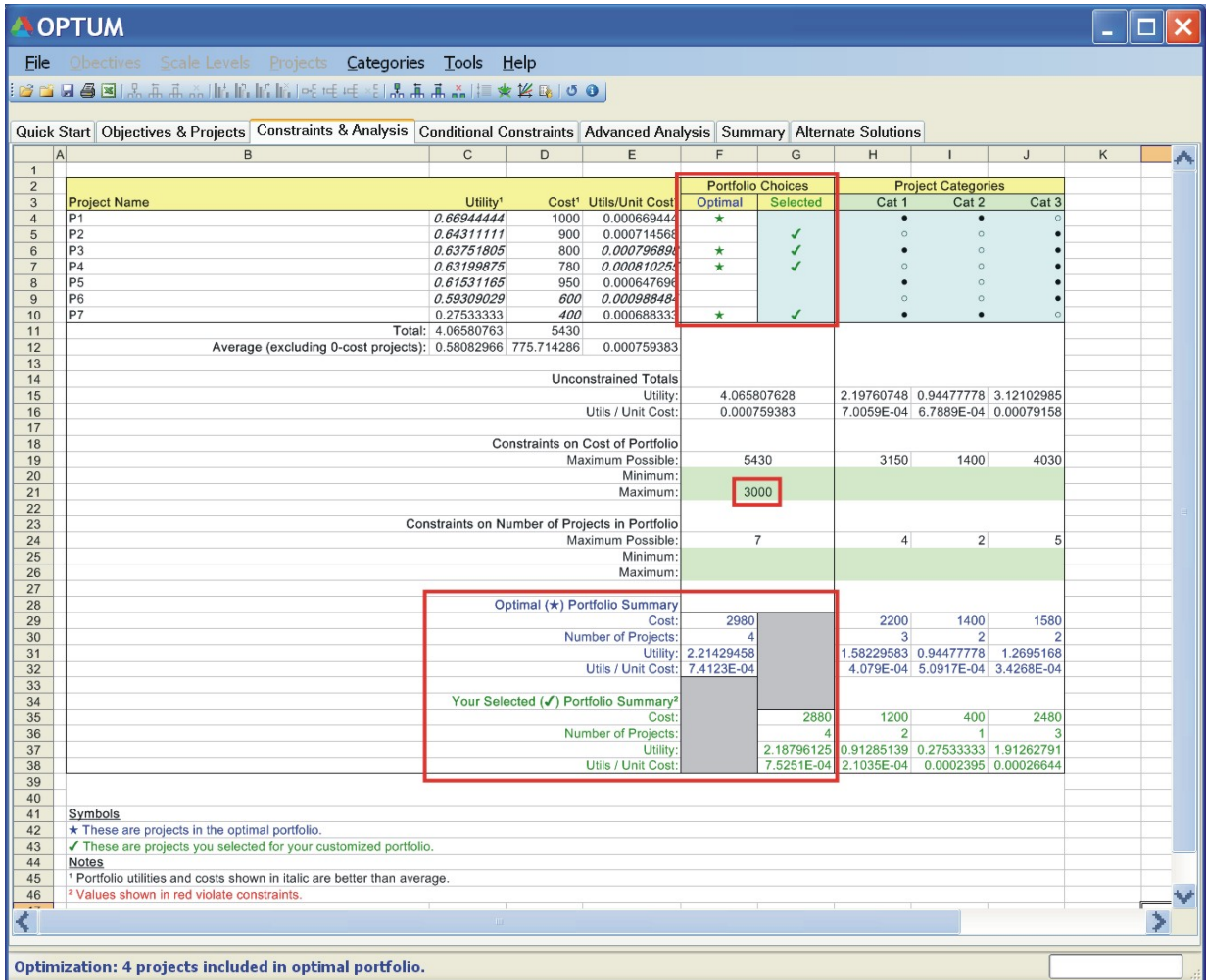



FIGURE 4.3 Optimized Results with Modified Cost Constraints

can make it virtually impossible to find the optimal solution by manual inspection, so the optimization capability becomes more essential as the problem grows in size and complexity.

4.1.2 Portfolio Optimization with Project-specific Constraints

In addition to constraints that apply to the total portfolio, OPTUM provides the user with options for implementing project-specific constraints. Figure 4.4 illustrates the screen area where the user is allowed to define project categorizations and then assign each project to any of the predefined categories. These categorizations provide a great deal of control for the user to introduce new limitations and conditional constraints for the portfolio optimization process.

New categories are added by pressing the add button “” and entering a name in the green-background cell near the top of the screen. The order of categories can be rearranged after

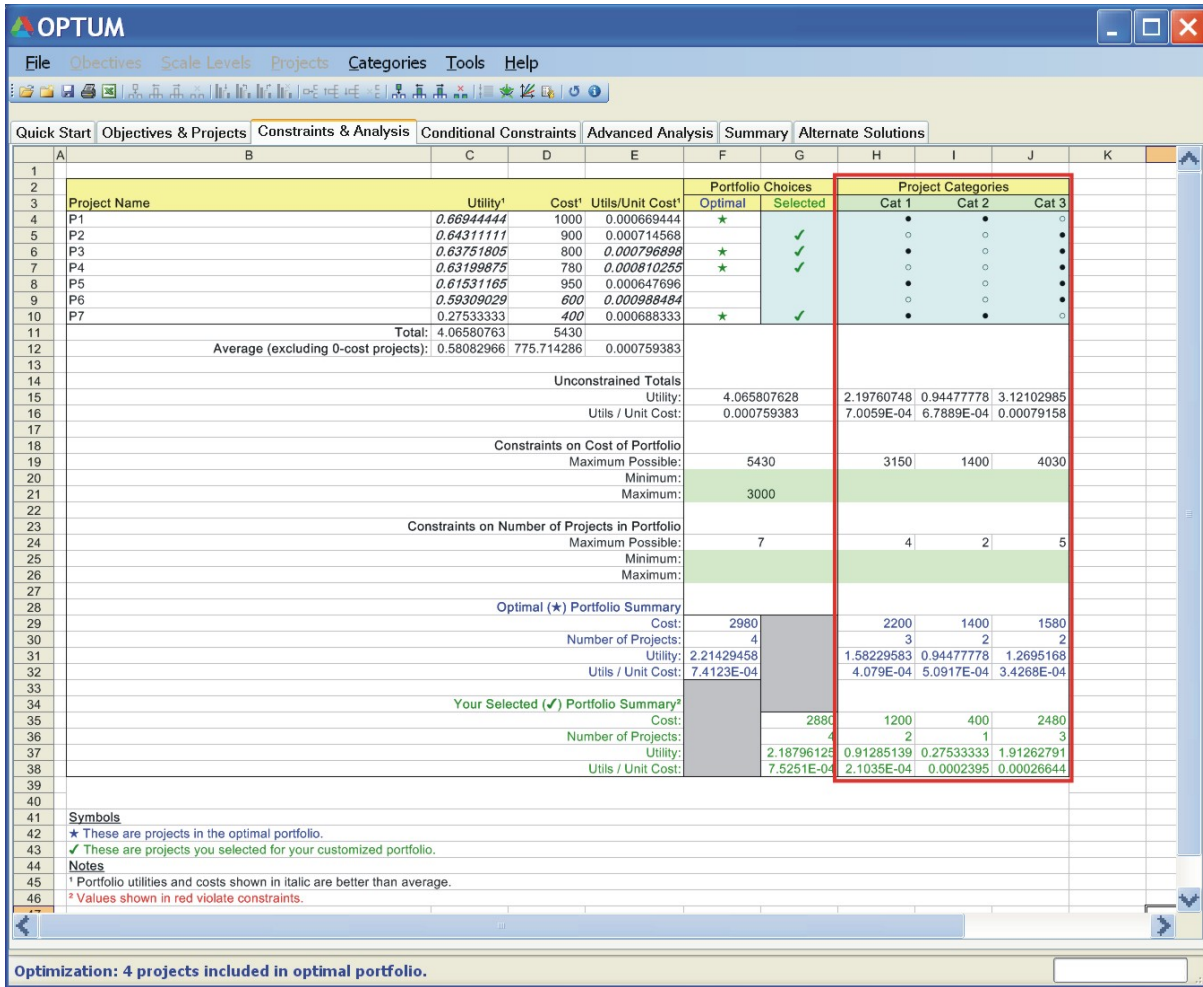





FIGURE 4.4 Portfolio Optimization with Project-specific Categorization Constraints

the categories are defined by using the move left button “” or move right button “”. Categories can be deleted by using the delete category button “”.

Once the categories are defined, the user can introduce many associations between the individual projects and the different categorizations. For the example shown in Figure 4.4, three project categories have been defined: Cat 1, Cat 2, and Cat 3. The icon “●” indicates that a project is associated with a given category, while the icon “○” indicates that a given project is not associated with that category. So in this example, Projects P1 and P7 are associated with categories Cat 1 and 2; Projects P2, P4, and P6 are associated with Cat 3; and Projects P3 and P5 are associated with Cat 1 and 3.

As a hypothetical interpretation, Cat 1 projects might correspond to R&D-related efforts, Cat 2 projects could correspond to new ventures, and Cat 3 could represent maintenance-related projects. So in this example, if the user wanted to ensure that at least one project was selected from each of the general categories, a set of minimum constraints (each set at the value of 1) could be introduced in the green-background data entry cells labeled “Constraints on Number of

Projects in Portfolio,” across from the “Minimum” heading and under each category (Figure 4.5). And if the user also wanted to ensure that no more than two projects were selected from any of the categories, similar entries set at the value of 2 could be entered across from the “Maximum” heading for “Constraints on Number of Projects in Portfolio” data entry fields, under each project category. These entries are also shown in Figure 4.5.

After these new constraints have been defined, the problem can be reoptimized, and the results as shown in Figure 4.5 will demonstrate that the selection of projects has changed. Now projects P1, P2, P6, and P7 represent the optimal collection of projects to satisfy all of the constraints. The total utility of this solution has dropped from 2.214 (in the case with only one constraint — a total cost constraint) to 2.181 (in a new case with a total cost constraint and also the project minimum and maximum constraints).

An additional observation that can be made from Figure 4.5 is that one of the summary fields for the user-defined portfolio has changed to the color red under the Cat 3 column. This indicates that the user selections violated at least one constraint for numbers of projects in that

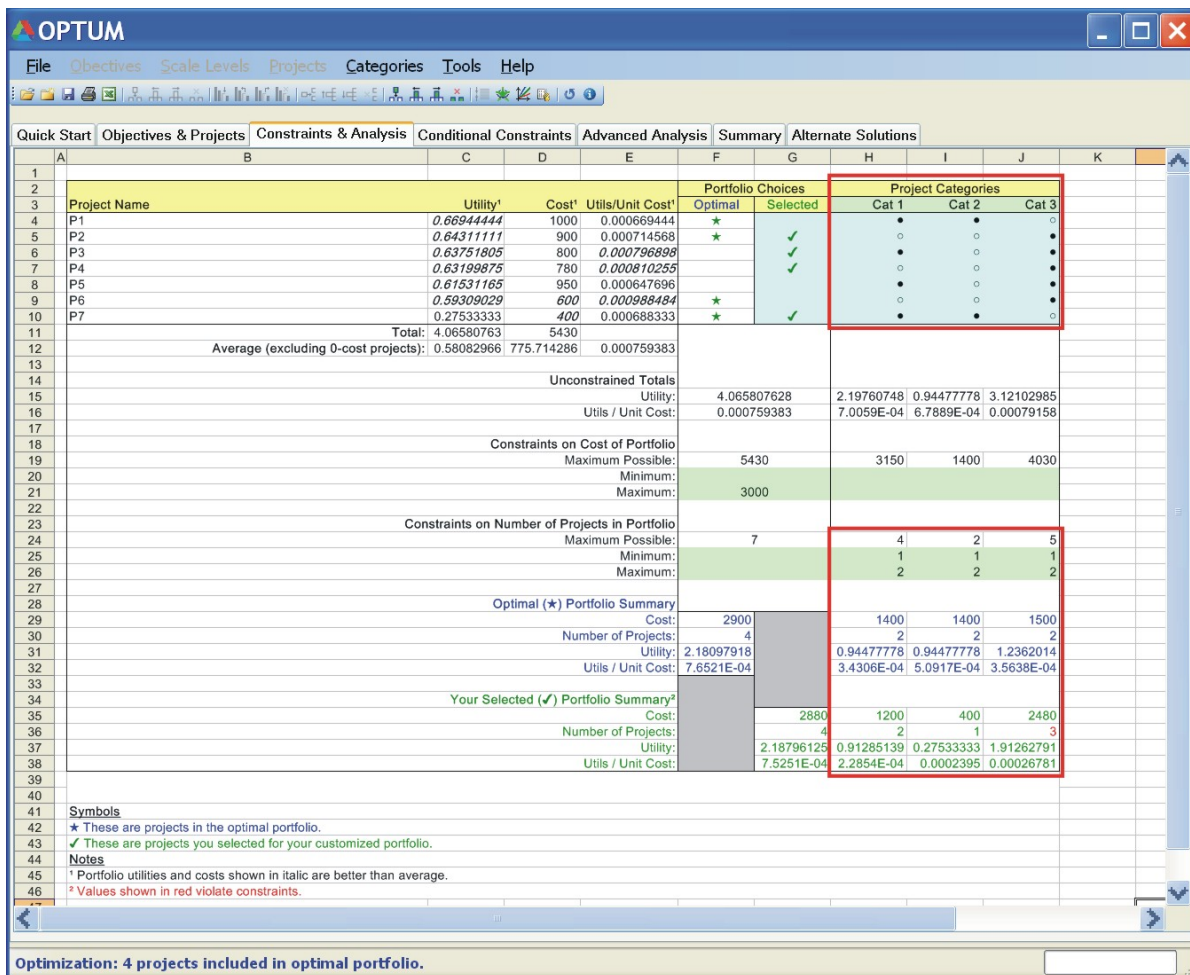


FIGURE 4.5 Results with Project Categorization Constraints Included

category. In this case, the user-selected portfolio included three projects from Cat 3, and this choice violated the requirement to include no more than two projects from any of the three categories. The red highlights in OPTUM are there to help the user recognize when manually selected portfolios do not meet the prescribed limitations.

In some cases — especially when many constraints have been defined — the problem may not have a valid solution (i.e., it may be over-constrained). In these cases, OPTUM will print an “Optimization: Optimal Portfolio Could Not Be Found” message at the bottom of the screen to warn the user that an optimal solution was not found and that the constraints need to be relaxed.

The preceding example uses constraints on the *numbers* of projects to illustrate this use of project-specific limitations. OPTUM also allows similar limitations to be placed on the *costs* of projects in each category. Implementation is parallel for costs and numbers of projects.

4.1.3 Portfolio Optimization with More Complex Constraints

In addition to simple upper and lower bounds on groups of projects, the category constraints can also be used to construct sophisticated conditional constraints on various projects. For example, suppose that a user would like the final portfolio to include one of two projects but not both of them. This can be accomplished by creating a new category, identifying both of the projects as belonging to this category, and then constraining the number of projects with a minimum of “1” and a maximum of “1.” The result is that exactly one project is selected in the optimal portfolio, and OPTUM will select the best of those two choices on the basis of the project utilities and any other constraints that are defined for the problem.

While more sophisticated constraints can be similarly constructed by using the “category” constraint mechanism, once the complexity increases beyond simple inclusion/exclusion rules, that method becomes challenging and susceptible to potential user errors. So OPTUM has been designed with special capabilities to address “conditional constraints.”

Figure 4.6 shows the OPTUM screen that manages the construction of conditional constraints. The menu-driven framework provides great flexibility and control over implementation of sophisticated conditional constraints.

To construct a new constraint, the user selects a specific option from general categories of (a) Include, (b) Exclude, (c) If/Then, (d) If-Not/Then, (e) If/Then-Not, or (f) If-Not/Then-Not. A total of 24 options are available within these categories, and these options provide different ways of specifying which projects are to be constrained, via modifiers such as “all,” “at least,” “none,” etc.

To implement a conditional constraint, the user selects a radio button for the basic type of constraint and then fills in the lists of projects from the candidates. Either one or two columns will appear when a specific radio button is selected. The user can then drag project names from

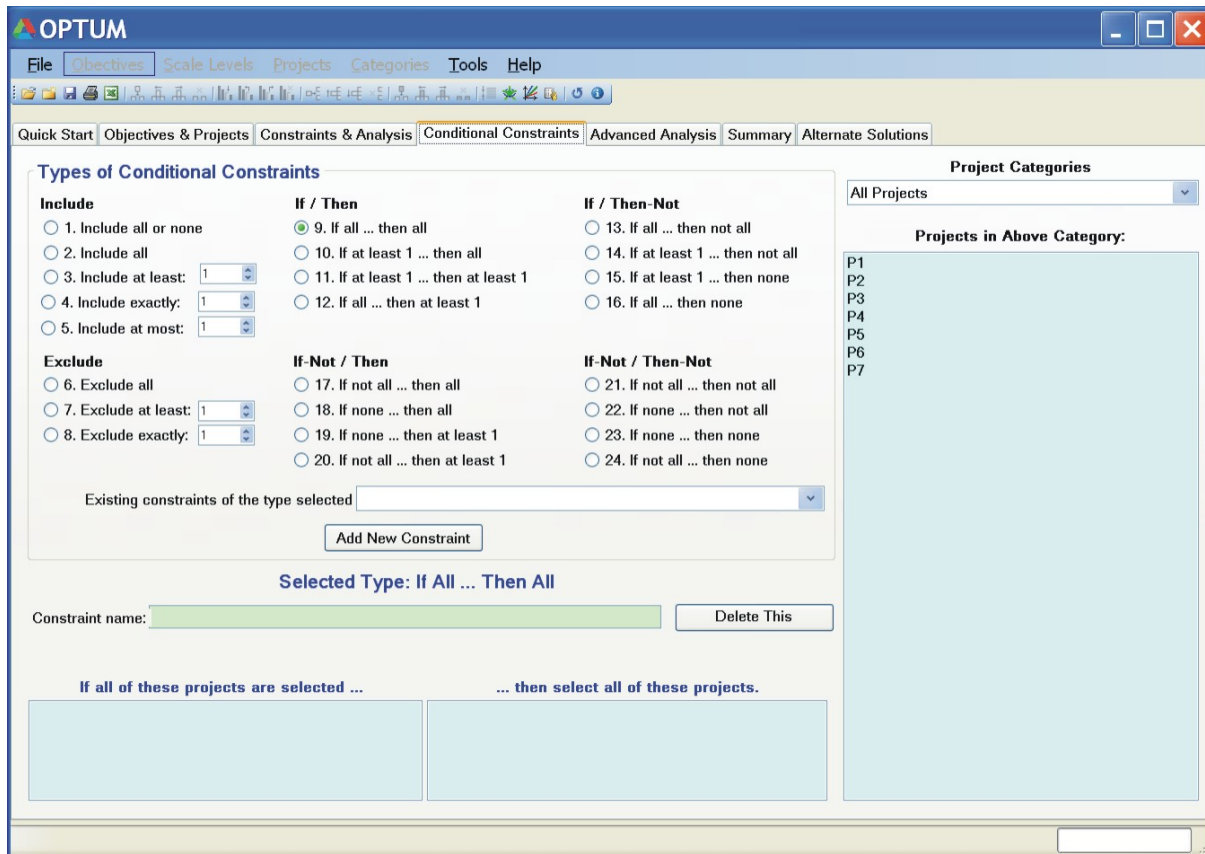


FIGURE 4.6 Conditional Constraint Options

the master list into each column to complete the constraint specification. As many constraints as needed may be constructed, but the user is cautioned that adding large numbers of constraints may render the problem infeasible if no valid solutions exist that satisfy all of the constraints.

4.1.3.1 Simple Conditional Constraint Example

As an illustration, to implement the example described at the beginning of Section 4.1.3 (include one, but not both, of two projects [e.g., P3 and P7]), the user would select Conditional Constraint Option 4 (“Include exactly [n]”). Then, under the set of projects to be affected, the user would specify *both* projects P3 and P7, and “n” would be set at “1.” The model will then ensure that the optimal solution contains one but not both of those projects.

4.1.3.2 Example of Constraint Implementation for Sensitivity Analysis

Several of the conditional constraints deserve special attention because they can assist in sensitivity tests. Conditional Constraint Options 6, 7, and 8 can be used to explore “next best” alternatives after an initial optimal solution has been located. How the user wants to explore or

define “next best alternatives” will determine which of the constraints to apply. The following examples illustrate possible uses of the constraints.

- *Conditional Constraint Option 6.* One approach to sensitivity testing is contained in Constraint Option 6, which requires that *all* of the projects from a designated solution *are to be excluded* from a new optimum. So if the user enters P2, P3, and P4 in the constraint definition, Constraint Option 6 would ensure that new solutions completely exclude *any and all* of the elements P2, P3, and/or P4.
- *Conditional Constraint Option 7.* A second option for sensitivity tests, Option 7, requires that *at least “n”* of a designated set *is/are to be excluded* in any new optimum solutions. So for Constraint Option 7, if the user enters P2, P3, and P4 in the constraint definition and sets “n” to the value of 1, then (in contrast to Constraint Option 6) new solutions such as {P2, P3}, {P3, P4, P7}, and {P2, P7, P8} *are all valid* under this definition, as well as solutions such as {P5, P6, P7}. Candidate solution sets such as {P2, P3, P4, P7} or {P2, P3, P4, P6, P7} are *invalid* options because they include *all* of the original projects P2, P3, and P4.
- *Conditional Constraint Option 8.* A third option for sensitivity tests, Option 8, requires that *exactly “n”* of a designated set *is/are to be excluded* in a new optimum. So, for Constraint Option 8, if the user enters P2, P3, and P4 in the constraint definition and sets “n” to the value of 1, then (in contrast to Constraint Option 7) new solutions {P2, P7, P8}, and {P5, P6, P7} *are not valid* under this definition. These solutions eliminate 2 and 3 previous set elements, respectively, and thus do not satisfy the requirement of “exactly 1” element being excluded. Candidate solution sets such as {P2, P3} or {P3, P4, P6, P7} would be *valid* options because they exclude *exactly 1* of the original projects (P2, P3, and P4).

The choice of exclusion criteria is driven by the goals and preferences of the user in conducting the sensitivity analysis. The three choices outlined above are included for illustration and show some of the versatility in OPTUM for exploring alternate optimums. Additional tools designed to assist in sensitivity analysis are discussed in Section 4.2.

The other conditional constraint options offered in Figure 4.6 are intended to provide a reasonably comprehensive set of options for implementing a broad range of inclusion and exclusion conditions. In some cases, it may be necessary to apply more than one of these constraints to achieve more complex limitations on the outcomes. Appendix A describes the specific logic formulations used to implement each of the conditional constraint options.

4.2 RESULTS, GRAPHS, AND SENSITIVITY ANALYSIS

The optimal portfolio selections are displayed in tabular form on the “Constraints and Analysis” screen (Figure 4.7). The listings highlighted in *blue letters and numbers* (not the blue background fields) correspond to the optimized selections. Figure 4.7 shows that these summary listings include optimized project selections (blue or green stars), total cost, total number of projects selected, total utility, and average utility per unit-cost (utls/unit-cost). Across the page from these total portfolio summary values are the category-specific summaries.

The summary results shown in *green letters and numbers* correspond to the user-specified portfolio selections. Entries in *red numbers* in the user-specified area indicate values that violate one or more of the constraints.

Examples in Section 3 and Section 4 illustrate that the results can (and might be expected to) change in response to user-defined constraints. However, if new constraints are not

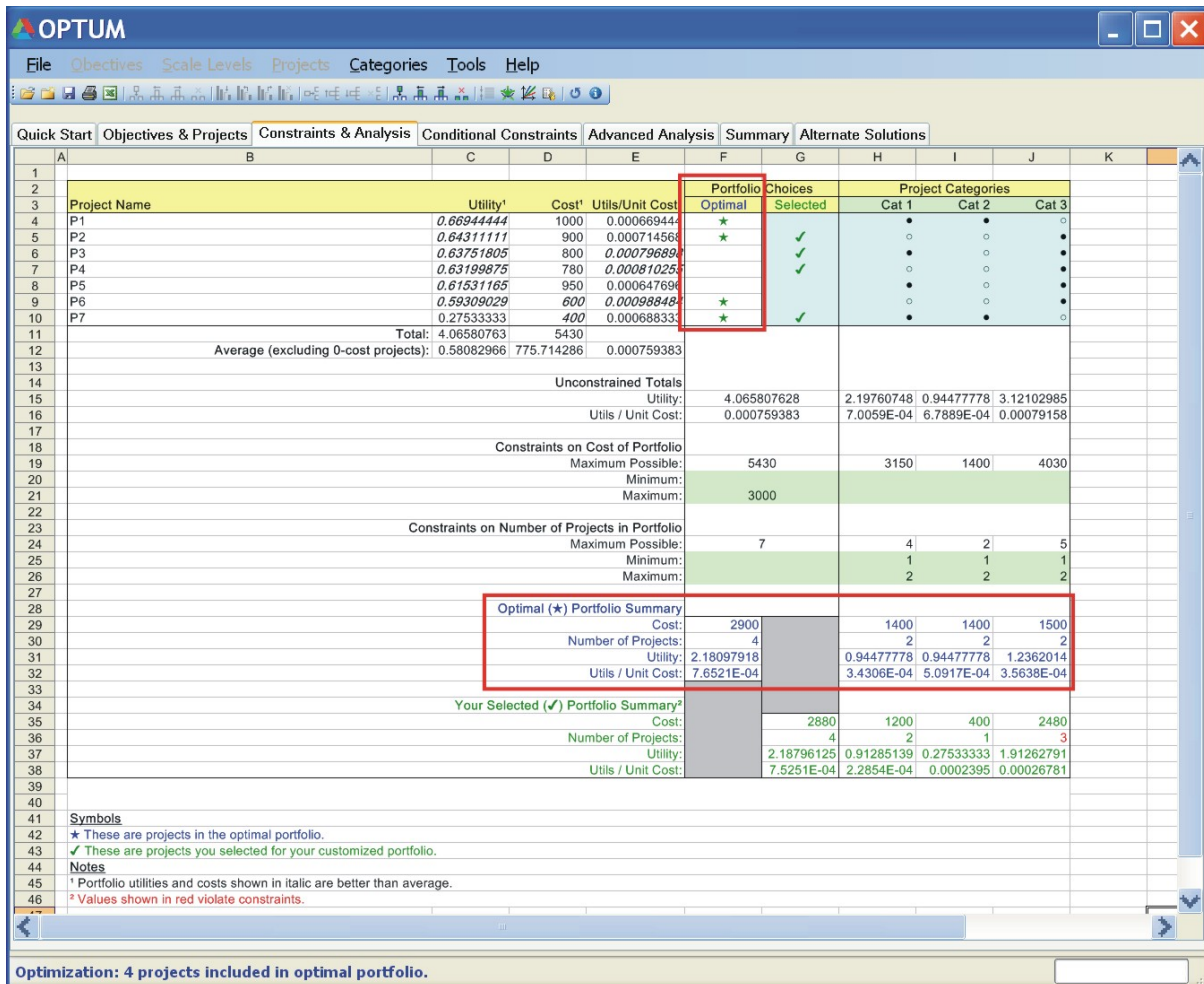


FIGURE 4.7 Optimal Portfolio Results


binding (i.e., if they are already met by an existing optimal solution), the optimal solution may not change from a previous solution.

The examples described in Section 4.1.3.2 provide guidance for basic sensitivity investigations. By selecting one of several conditional constraints (inclusions or exclusions), the user can explore alternative optimum solutions.

4.2.1 Modification of Cost Constraints for Sensitivity Analysis

Another approach to examining alternative optimal solutions is to modify the maximum or minimum cost constraints to ensure that the previous optimum is not feasible for a subsequent solution. For the example in Figure 4.7, which shows an optimal solution with a total cost of \$2900K, the user could change the *maximum cost* to \$2899K, and this would force an alternative solution to be found. Similarly, the user might set the *minimum cost* constraint at \$2901K, which would also force another solution to be located. The directions for following these sensitivity tests (higher or lower cost constraints) would depend on the user's interests and the nature of the problem being examined. However, the user should be aware that in using this approach, multiple solutions that have identical costs might be overlooked.

4.2.2 Graphical Display of Optimal “Frontier” Solutions

In addition to the tabulated results, OPTUM also provides a graphical depiction of the optimum solution shown in the context of other nonoptimal alternatives that are the best solutions at other cost levels. This curve is sometimes referred to as the “frontier” of optimal solutions for different cost levels. Figure 4.8 shows a setup screen that opens when the user selects the “” button. This setup screen leads the user to the graph shown in Figure 4.9. To display the graph shown in Figure 4.9, the user presses the “Display Solutions” button in the setup screen. The graph shows a sample case, with the utility value plotted on the vertical axis and total cost plotted along the horizontal axis. The user can select other parameters for the axis choices — including cost, utility, utls/unit-cost, or any of the objectives.

The results in Figure 4.9 provide useful insights into the sensitivity of a given problem to constraints and the availability of other feasible alternatives in the neighborhood of the current optimal solution. In this chart, the *red* outline traces the frontier of *constrained* optimal solutions at any given cost level within user-specified bounds of cost constraints. These are solutions that satisfy all of the user-defined constraints. In this graph, the optimal solution is the point of highest utility value on the Y-axis that falls within the boundaries of the red outline (typically the point that is furthest to the right on the red curve).

The *blue* outline traces the *unconstrained* solution frontier over the entire range of cost possibilities. The blue solutions do not (necessarily) satisfy the constraints, except where the blue outline coincides exactly with the red outline. In general, there are other suboptimal combinations of projects not plotted that would be positioned *under* the red or blue frontier lines. These are not included in the graph because they represent outcomes that can be improved with

solutions that are positioned *on* the red outlines (satisfying constraints) or *on* the blue outlines (not satisfying constraints). Solutions positioned on the outlines represent the outcomes with the highest utility for a given cost.

If the user wishes to explore lower-cost portfolios with nearly the same value of the existing optimum, the chart in Figure 4.9 shows that several feasible alternatives (that satisfy the constraints) exist to the left of the existing optimum (the point furthest to the right on the red graph located at cost = \$2900K). To determine the composition of these alternative solutions (which projects are included), the user can adjust the maximum cost constraint to the lower values and rerun the problem. Each of the lower-cost options can be found through this iterative approach. Section 4.2.3 describes yet another method of exploring alternative optimal solutions, and that method does not require iterations, which are needed when the optimal frontier curves are used.

Conversely, if the user is interested in higher-cost options that increase the total utility of the portfolio, the chart in Figure 4.9 reveals that solutions are potentially available to the right-hand side of the current optimum solution (to the right of \$2900K). Depending on the problem, the identification of such solutions may be a simple matter of adjusting the maximum cost constraint, or it may require a more extensive look at other constraint, such as project categorization limits or conditional constraints. The blue outline serves as a guide on the maximum utility possible when constraints are relaxed or completely abandoned.

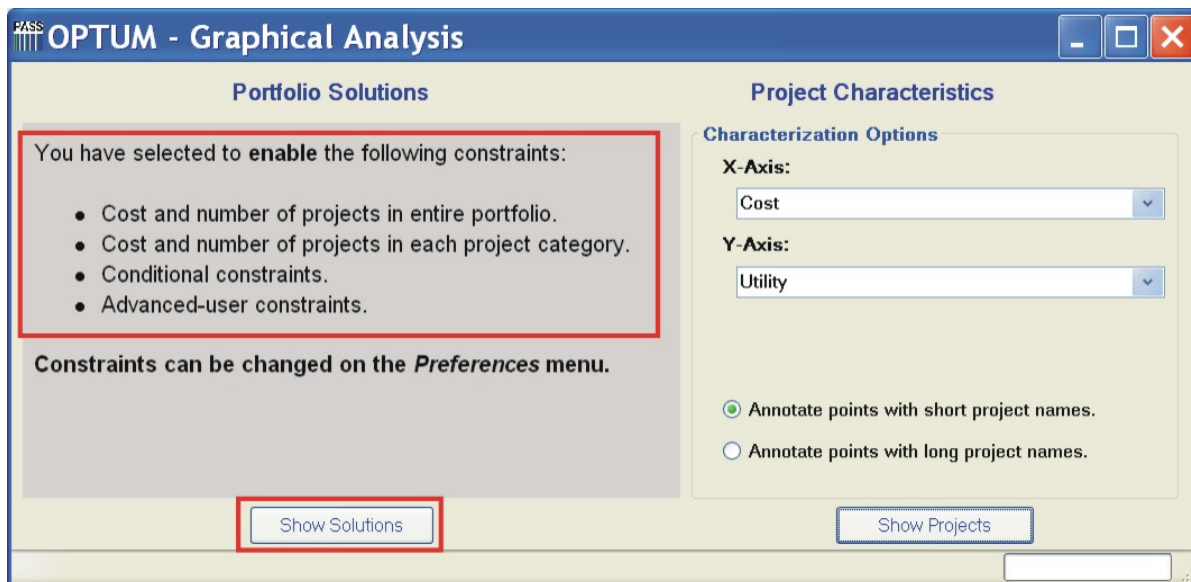


FIGURE 4.8 Setup Screen for Preparing Optimal Frontier Graph

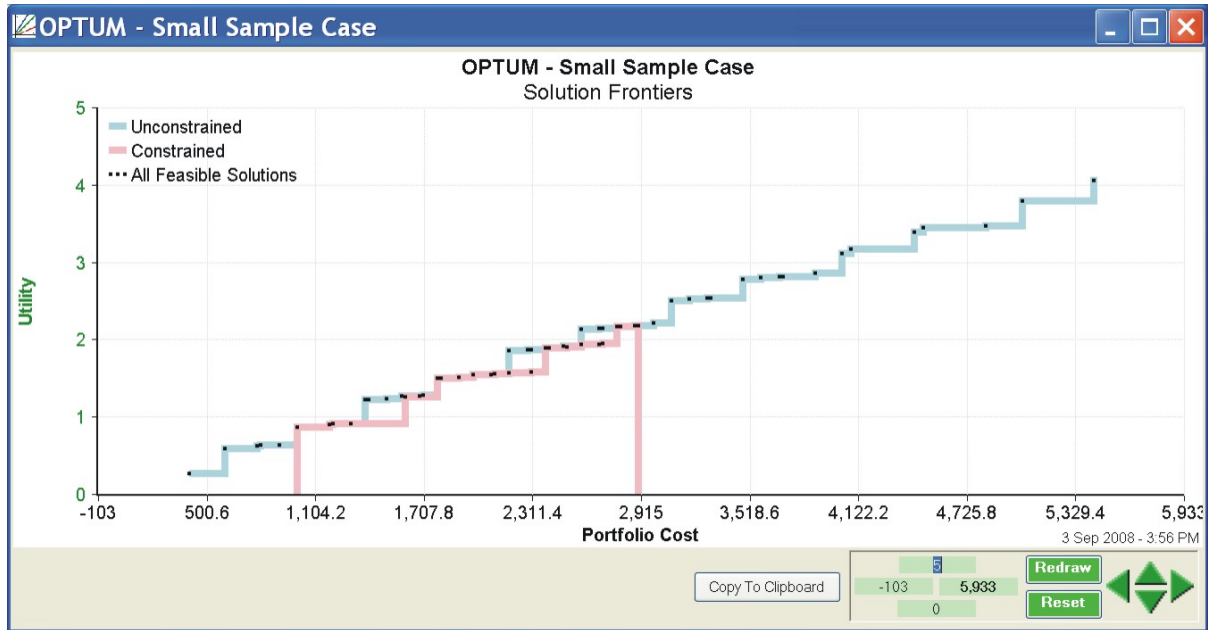


FIGURE 4.9 Example of Optimal Frontier Graph

4.2.3 Graphical Display of Alternative Optimal Solutions

OPTUM offers one additional and very powerful graphical display for evaluating results and alternative solutions. This feature is available through the “Alternative Solutions” tab. Selecting this option first brings up a screen similar to that shown in Figure 4.10, but initially all the cells are blank. Once the user presses the “Create Alternative Solutions” button, the screen shown in Figure 4.11 will appear (and the entries in Figure 4.10 will be populated in the cells).

This feature allows the user to dynamically view the entire spectrum of optimal solutions for a given set of constraints. The user can slide the red line across the range of costs, and at each cost increment, the optimum solution is displayed under the red line and equivalently by the projects that are bolded on the left margin of the chart. So in Figure 4.11, the results are showing that at a cost of \$1500K, the optimal solution is {P2, P6} (same as shown in the example of Figure 4.2). Figure 4.11 lends itself to another worthwhile interpretation, by letting the user observe which projects enter into a majority of the optimal solutions and which ones are included in only a small number of optimal results. The user gets a visual indication of which projects are most valuable and least sensitive to budget limitations, and which ones are most sensitive to different levels of total expenditures.

Here are a couple of cautionary notes on using the Alternative Solution option. First, the time required to generate the solutions (shown in Figure 4.11) increases as the number of projects increases. For small problems (i.e., 1–50 projects), the solution may appear almost instantaneously or within tens of seconds. For somewhat larger problems (i.e., 75–100 projects), solution times may approach a minute (depending on CPU speed). And of course, larger problems will require even longer solution times. OPTUM builds the Alternate Solution chart by

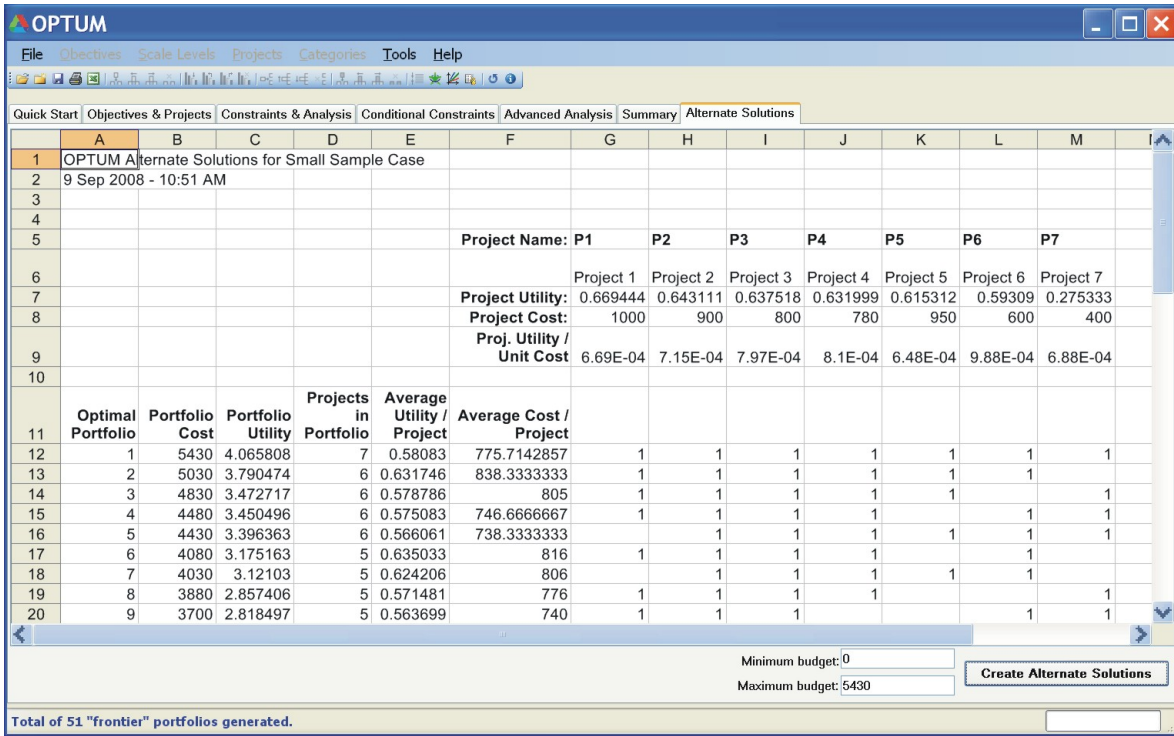


FIGURE 4.10 Menu for Constructing Alternate Solution Graph

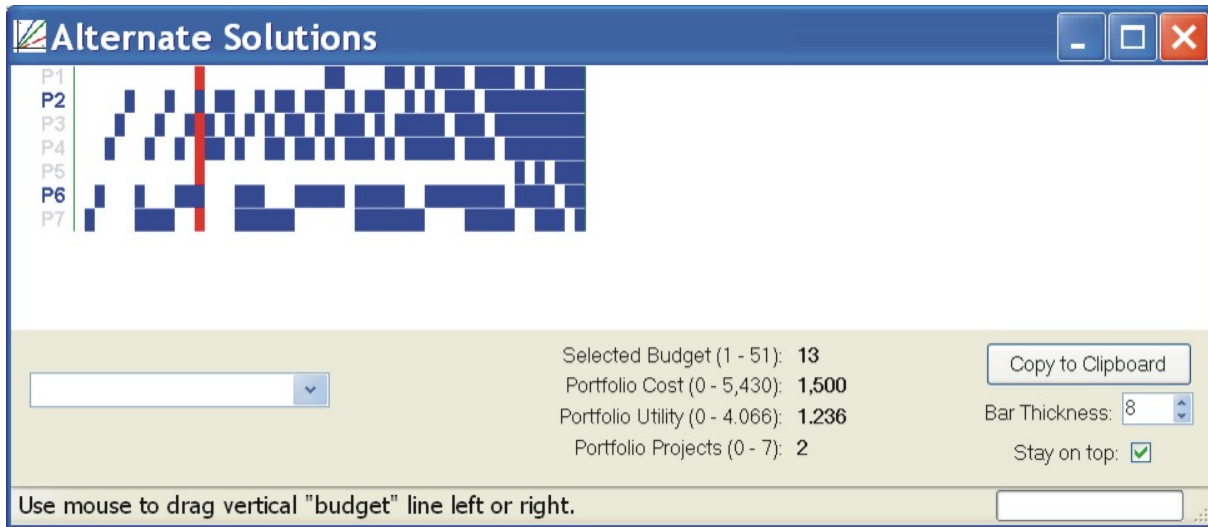




FIGURE 4.11 Illustration of Alternate Solution Graph


solving for the highest possible cost, imposing a cost constraint that is lower in cost by subtracting the lowest common denominator of project costs, and then re-solving for a new optimum. The process iterates until all of the potential optimum results are obtained.

A second note of caution is to observe that the X-axis scale in Figure 4.11 is not uniform. The optimum solutions are simply stacked side by side for convenience in viewing and displaying the results. So as the user moves the red line across the cost range, equal movements of the cursor are not necessarily paralleled by equal increments in cost. The user can view how much the cost is changing with cursor movements by observing the displayed values for “Portfolio Cost” tabulated near the bottom of Figure 4.11.


4.3 EXPORTING RESULTS

Results from OPTUM can be exported for further analysis or reporting. One method is to prepare a summary report by using the report button “”. This feature prepares a Microsoft Word document that includes definitions and assessments of projects, objectives, constraints, and solutions. The other method is to use the spreadsheet button “”, which saves an “.xls” file for later use as a spreadsheet in Microsoft Excel. The user should be aware that saved spreadsheets do not contain the equations or optimization functionality embedded in OPTUM; thus, saved spreadsheets would be intended primarily for review and formatting or for post-optimization calculations. The user could add customized calculations into one of these post-optimization spreadsheets to derive additional problem analysis metrics or compare alternative solutions.

4.4 ADVANCED ANALYSIS CAPABILITIES

In addition to the usual constrained optimization features already described, OPTUM offers an advanced formulation feature. Under the “Advanced Analysis” tab, the user can customize any type of constraint that the optimization engine can accept. To use this feature, the user is encouraged to become familiar with lp_solve capabilities and syntax (Notebaert et al. 2008). Very complex constraints can be constructed to accommodate special needs, and the user can also modify the objective function specifications. The downside of using this feature is that the constraint construction is not menu-driven and requires greater attention from the user to avoid errors or improper constraint representations. It is also important for the user to clear (delete) any customized modifications when he/she is finished using the advanced features, so that the problem formulation matches the data input menus (as contained in the “Objectives & Projects” and “Constraints & Analysis” tabs). The User Preferences menu “” offers a simple toggle menu to turn the advanced analysis constraints on and off (Section 4.5).

4.5 USER PREFERENCES

The User Preferences button “” is included for customizing how the OPTUM display appears and how the screens respond to scrolling. This button brings up the window shown in Figure 4.12.

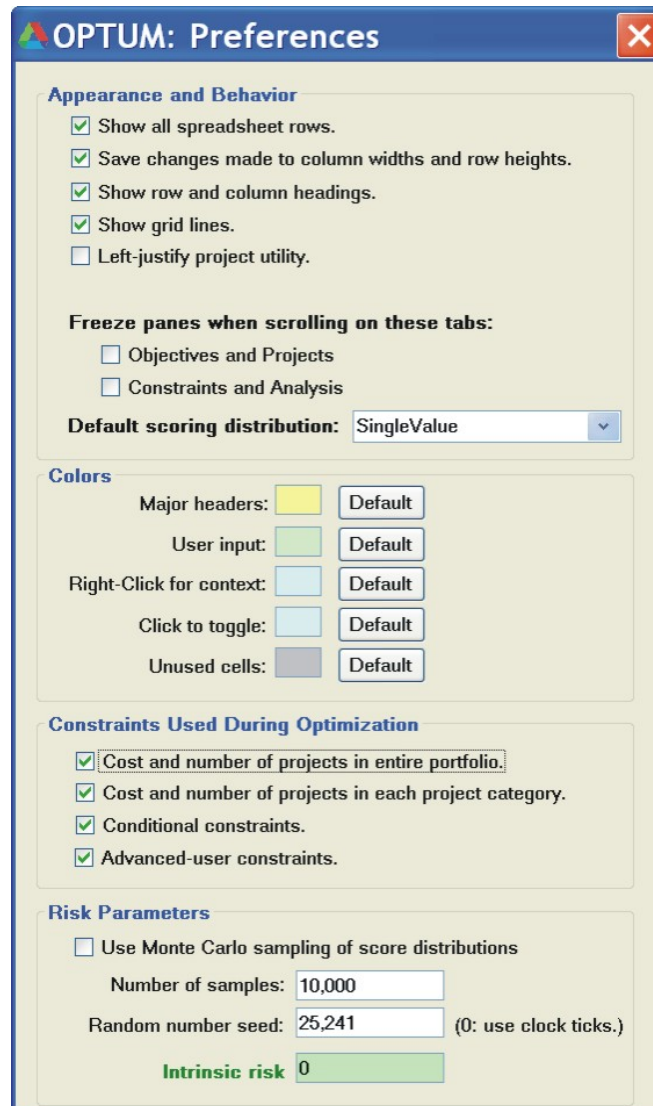


FIGURE 4.12 User Preferences Menu

As shown, the preferences window provides options for these default behaviors:

- Displaying or not displaying all rows (i.e., expanded or hidden);
- Saving the changes that a user has made to column widths and row heights;
- Showing row and column headings (numbers and letters), as in conventional spreadsheet format;
- Showing grid lines for rows and columns;

- Applying left justification to the total project utility (so screen is always in view without the user having to pan across wide pages);
- Freezing header panes (to keep row and column labels in view while user is panning and scrolling);
- Selecting the default type of project scoring distribution to be used when new projects are being added;
- Choosing colors;
- Selecting which (if any) constraints are to be applied during optimization;
- Specifying Monte Carlo sampling parameters, including sample size and seed number for constructing the project scoring distributions; and
- Setting the intrinsic risk parameter for recognizing risk-averse and risk-tolerant preferences (i.e., nonlinear) over the range of continuous objective scales.

When implementing the freeze pane options, the user will notice a double image of the row and column headers until the first panning or scrolling has been performed. Thereafter, the screen will show a single image of the headers, and they will stay in view while the rest of the table entries are panned or scrolled for viewing.

5 OPTIMIZATION ENGINE

OPTUM uses an optimization engine to derive the optimal mix of projects that satisfies the constraints and maximizes the total portfolio utility value. The optimization engine is invoked by pressing the optimization button “★” (or “✦” if constraints are defined and active), and the results are automatically shown within the screen depicted in Figure 2.2. The following section describes the motivation for using optimization, the criteria used to select the optimization engine, and the optimizer capabilities.

5.1 RATIONALE FOR USING OPTIMIZATION

In the context of portfolio management, the potential analytical methods and strategies vary over a broad range that includes manual techniques, heuristics, and optimization. To help users make decisions and selections, a wide range of metrics (measures of effectiveness) can be calculated that provide indicators for using resources efficiently or maximizing objectives (e.g., expected return per dollar invested, person-weeks per net-project-impact).

Manual methods rely on the analyst to choose among selected options (e.g., investments or projects) while recognizing limitations in resources (e.g., monetary, human, equipment, time, material resources). The challenge in using manual methods is that as the number of possible activities, resource options, and/or resource constraints grows in size, the number of feasible combinations also grows, and it grows very rapidly. Therefore, manual selections are likely to produce significantly suboptimal results. Although these results may be feasible, it is likely that better solutions that would make much better use of resources would be available.

Heuristic methods can often improve upon manual selections, because metrics are systematically used as a guide for the selection process. By employing fast computational methods, heuristics can examine very large numbers of combinations of project selections and then select the choices that provide good measures of effectiveness (efficient use of resources to satisfy overall objectives). While heuristics routinely improve upon manual selection methods, they do not ensure the optimal use of resources.

Finding the true optimal solution is the focus of more formal optimization methods. As the number of candidate projects grows larger than a few, and as the number of constraints or restrictions grows beyond a small number, the ability to find the best solution by using manual or heuristic methods becomes exponentially more difficult. Through their evolution over time, many “solvers” have reached a point where they are effective and stable in solving large-scale problems containing hundreds or thousands (and even tens of thousands) of variables and constraints. Such large-scale problems far exceed the capability of manual methods and generally challenge heuristic methods when it comes to locating the best solutions.

To summarize, the benefits of using a formal optimization engine are to:

- Identify the true optimal (instead of merely reasonable or good) solutions,

- Maximize the use of resources (human, monetary, material, equipment, etc.), and
- Methodically examine alternate near-optimal solutions in sensitivity tests.

The optimization engine candidates and selection process for OPTUM are described below.

5.2 OPTIMIZER SELECTION CRITERIA

The following issues were considered during the optimizer selection process:

1. Cost and licensing requirements (expense and portability),
2. Problem-solving capabilities:
 - Zero-one LP and
 - Mixed-integer LP,
3. Problem size and speed
4. Ease of use and implementation:
 - Hands-off operations (robust feasible solution capability) and
 - Application interfaces (APIs),
5. Reliability (prior testing and verification),
6. Software documentation, and
7. Maintenance.

These seven categories formed the general basis for selecting the optimization engine. The selection criteria were applied informally and were intended to *guide* the process of selecting a new solver without requiring rigorous comparative measurements. Each consideration is briefly discussed in the following sections.

5.2.1 Cost and Licensing Requirements

Cost-of-purchase and licensing issues were a significant consideration in software selection, not so much from the standpoint of the one-time purchase expense but more from the standpoint of reoccurring costs and licensing restrictions for each copy of OPTUM to be distributed. One-time purchase fees were considered acceptable as long as the expense was within reasonable limits. However, a primary goal was to develop an end-product that could be shared openly with multiple users across organizational boundaries. This consideration meant

that once the solver was obtained or licensed, it needed to be free of licensing restrictions for additional copies.

A fixed dollar amount (for one-time purchase) was not quantified during the review and screening process because solvers that required a purchase fee generally also required licensing fees for each add-on user or installation. Because the extent of distribution for OPTUM could not be anticipated at the time of development, the licensing criterion was treated as a need for unlimited distribution rights.

These considerations significantly limited the field of choices for selection. However, a number of public-domain solvers did meet other selection criteria. Commercial software was not automatically eliminated from consideration, since it was thought that some licensing arrangements might permit a large number of copies to be distributed for a predetermined cost.

5.2.2 Problem-solving Capabilities

For portfolio optimization problems, it is important that a solver be able to explicitly treat projects as discrete, indivisible entities. Simple linear programming (LP) methods are not adequate because, although they can solve conventional problems, they cannot treat decision variables (x_i) as being discrete (i.e., only able to take on integer values 0, 1, 2, 3, ...). For portfolio optimization, the user generally needs to treat projects as all-or-nothing candidates; each project is either chosen for inclusion or not included in the final selection (equates mathematically to a “0–1” choice). Other integer multiple representations (0, 1, 2, 3, ...) can occur in conjunction with nondivisible resources, such as the number of discrete machines available to complete different tasks. These portfolio optimization problems contrast with conventional LP problems, which potentially allow *portions* of projects (fractional parts, such as 0.35 of Project A and 0.77 of Project B) to be selected.

So although simple problems can sometimes be solved with the generalized LP formulation of the type:

$$\text{Maximize } f(x_i) \text{ where } f(x_i) = ax_1 + bx_2 + cx_3 + \dots + kx_n,$$

the OPTUM application requires a solver capable of the following additional requirement:

and where some x_i are integers (0, 1, 2, 3, ...) or binary (0, 1).

In general, OPTUM uses the binary restriction, but a solver that also offers integer or real value solutions was selected. Although the additional real and integer capabilities were expected to have limited use, the developers added them because they did not want to unnecessarily limit the modeling tool in case applications that required the additional flexibility arose. Solvers that can accommodate the integer and binary stipulations are typically referred to as mixed-integer LP solvers (i.e., MINT-LPs or MILPs).

5.2.3 Problem Size and Speed

The size and speed criteria were considered critical for practical reasons. It was anticipated that problems that might include dozens of objectives, hundreds of projects, and hundreds of constraints could occur. These could lead to formulations requiring thousands of variables. And, as noted above, the solver needed to handle integer and zero-one variables.

Some solvers, such as Excel's standard optimizer (spreadsheet implementation), satisfy the cost and licensing requirement but cannot handle sizeable problems and constraints or become very slow when they are looking for the solution to such problems.

5.2.4 Ease of Use and Implementation

Because OPTUM was developed for a mixed audience of potential users, the optimization engine needed to have a "hands-off" robustness and stability. Some optimization engines have sensitive internal parameter settings (e.g., search directions or step size) that require attention and monitoring in order to function efficiently and ensure optimal solutions. Although those types of solvers may be customized to solve specific types or sizes of problems faster than other more general algorithms, their requirement of constant attention to optimization parameters was considered unacceptable for this application.

In addition, some optimizers inherently interface more easily with other modeling components and software modules. Including various APIs in the OPTUM design was considered a significant plus to ensure that its development cost and time would be reasonable.

5.2.5 Prior Testing and Verification

It was considered highly important that users be confident in the functionality of the optimization engine for OPTUM. The candidate solvers considered included those that had been thoroughly tested, debugged, and verified against standard and nonstandard test problems.

5.2.6 Software Documentation

Up-to-date documentation of the optimizer software was considered an important criterion that would help in its implementation. Some solvers that might otherwise have satisfied the functional requirements do not have the detailed or useful documentation that would ensure that their integration could be completed successfully.

5.2.7 Maintenance

As a final consideration, optimizers were examined with regard to the amount of ongoing attention that is paid to maintenance issues and user feedback. Nearly all computer software can

be expected to encounter unanticipated bugs or odd behavior when it is subjected to rigorous and extensive testing — testing beyond the amount that might be feasible during the development process. Finding optimization engines that are routinely maintained for any newly discovered bugs or exceptions represents a very positive component of the selection process.

5.3 INITIAL OPTIMIZER CANDIDATES

In the initial screening process, a large number of optimizers were considered and reviewed for possible use in OPTUM. The list included:

- LINDO/LINGO (Schrage 1991),
- GAMS (Brooke et al. 1992),
- Excel embedded spreadsheet solver (Microsoft 2003),
- lp_solve (Linderoth and Ralphs 2005; Notebaert et al. 2008),
- COIN solvers/components (Linderoth and Ralphs 2005),
- CBC (Linderoth and Ralphs 2005), and
- Others (e.g., ABACUS, BCP, bonsaiG, GLPK, MINTO, and SYMPHONY) (Linderoth and Ralphs 2005).

The original candidates were assembled from in-house software packages, literature reviews, and Web searches.

5.4 SELECTED SOLVER

After careful consideration, one optimization engine emerged as the best candidate for OPTUM: lp_solve. This MILP solver has been well tested, can efficiently solve the types and sizes of problems anticipated for portfolio optimization, and can be easily licensed as an integrated product within other software packages at no cost for each installation. lp_solve is flexible in terms of its integration with various programming environments, such as C++, Excel, and Java, among many others. Furthermore, lp_solve has been thoroughly documented and is currently maintained with regard to user-reported issues, fixes, and improvements.

Some options that had already been licensed and installed at Argonne and proven to be functionally viable (e.g., LINDO/LINGO and GAMS) ultimately had to be eliminated from consideration because of other issues, such as cost and licensing. To ensure portability to other organizations without the need for special licensing, these candidates were treated as being less favorable.

The Excel solver option, because it is a standard plug-in for anyone who already has a licensed copy of Excel, satisfied the licensing requirement. Excel could also be a feasible candidate because of its widespread use. However, it is limited in terms of the size of the problems it can tackle, and it has a number of functional parameters that require attention and adjustment to ensure optimality. Moreover, other issues, such as its poor speed, eliminated Excel from the final selection.

Collections of solvers, such as COIN and others in the original candidate list, are well-respected in the optimization community but have serious implementation issues for a hands-off application like OPTUM. Some of these solvers are modular, and orchestrating the various modules, setting proper values for internal parameters, and making adjustments to address the various types of problems require a significant amount of attention.

5.5 IMPLEMENTATION

lp_solve was selected for implementation, and it was successfully integrated into OPTUM with very few difficulties. Once installed, the optimizer allowed OPTUM designers to expand some of its planned features and capabilities. Included in its expanded capabilities are:

- *Access to problem reformulations* (manual edits to constraints and objective function). This feature gives an experienced user a limitless range of problem-customization options.
- *Access to diagnostics*. If an optimal solution to a problem cannot be located or a set of problem conditions that is infeasible to solve is encountered, this feature provides possible leads for rectifying the problem.
- *Ability to construct conditional constraints* (e.g., prerequisites or other inclusion and exclusion criteria).

Interactions with the lp_solve developers have been positive exchanges. When OPTUM was implemented, a subtle (nonfatal) program bug was discovered. It was addressed by the lp_solve maintenance team/community.

6 CONCLUSIONS

OPTUM provides a powerful and versatile tool for selecting, optimizing, and analyzing portfolios. The software facilitates the entire process of portfolio assessment, including constructing the fundamental evaluation objectives, evaluating individual projects, and analyzing the results. The software also facilitates comparing user-preferred selections with optimal project selections.

A wide range of constraints can be applied in OPTUM to account for various resource limitations and upper and lower limits on the number of projects to be included in the portfolio. Conditional constraints can also be introduced, so that the inclusion or exclusion of one or more projects can force other projects to either be excluded or become mandatory. The optimized solution will then reflect these additional constraints in the sets of projects selected for the optimal portfolio.

OPTUM allows the user to introduce an “intrinsic risk parameter” that explicitly recognizes risk-averse or risk-tolerant preferences exhibited by various decision makers. This parameter provides a convenient mechanism for capturing potential nonlinear preferences over the ranges of (continuous) objective scales.

Several types of graphs of alternative optimum solutions are included to assist the user in understanding the outcomes in the context of other available solutions. These provide very useful tools for exploring sensitivities and compositions of near-optimal alternatives.

On the basis of criteria that included portability, cost, licensing considerations, and problem size capability, `lp_solve` was chosen as the optimization engine. It performed very well in all test cases and continues to be maintained, updated, and verified by a large user community. It has the speed to solve large-scale problems and the analytical features to handle complex constraint specifications. It can be freely distributed as long as it is appropriately acknowledged in the application software.

The OPTUM software can be readily applied to other non-portfolio resource allocation problems. Its framework is very flexible, allowing the user to define problem objectives and a rating scale for each alternative (which is not necessarily a project) to be considered. With such versatility, the OPTUM software is a powerful tool applicable to a broad range of resource-constrained optimization problems.

During the tool design process, significant attention was devoted to simplifying data input and problem definition procedures. The simplification effort resulted in a single spreadsheet-like form for entering all of the critical problem definition information (objective definitions and project evaluations) and project characterizations. A second screen shows all the requirements for defining basic constraints and viewing outcomes. A third screen provides a streamlined environment for implementing complex conditional constraints. The end result is a compact interface that facilitates problem construction and analysis.

7 REFERENCES

- Brooke, A., et al., 1992, *GAMS: A User's Guide*, The Scientific Press, South San Francisco, Calif.
- Hammond, J.S., et al., 1998, *Smart Choices: A Practical Guide to Making Better Decisions*, Harvard Business School Press, Sept.
- Jusko, M., et al., 2006, unpublished information, Argonne National Laboratory, Argonne, Ill., Apr.
- Keeney, R.L., 1992, *Value-focused Thinking: A Path to Creative Decisionmaking*, Harvard University Press, Cambridge, Mass.
- Keeney, R.L., and H. Raiffa, 1993, *Decisions with Multiple Objectives — Preferences and Value Tradeoffs*, Cambridge University Press, Cambridge and New York.
- Linderoth, J.T., and T.K. Ralphs, 2005, *Noncommercial Software for Mixed-Integer Linear Programming*, Jan. revision.
- Microsoft Corporation, 2003, *Microsoft Office Excel*, part of *Microsoft Office Professional Edition 2003*.
- Notebaert, P., et al., 2008, *Introduction to lp_solve 5.5.0.12*, <http://lpsolve.sourceforge.net/5.5/>.
- Schrage, L., 1991, *LINDO: An Optimization Modeling System*, The Scientific Press, South San Francisco, Calif.

APPENDIX A: CONDITIONAL CONSTRAINT FORMULATIONS

The “Conditional Constraints” tab contains radio buttons that allow the user to force the inclusion or exclusion of projects and to establish prerequisites and other conditional rules for project selection. The types of constraints are displayed in Figure A.1, and the equations used to implement each constraint are described below.

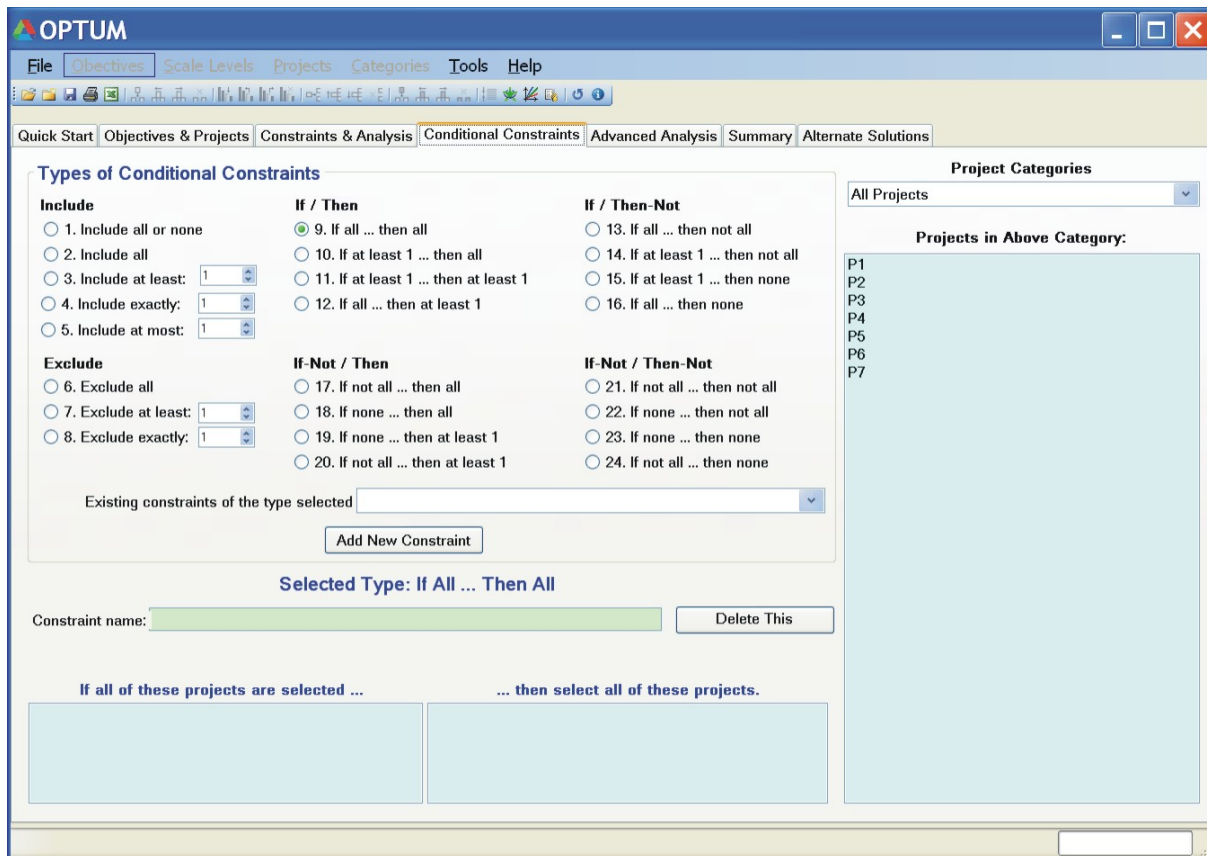


FIGURE A.1 Conditional Constraint Options

When one of the conditional constraint types is selected, one or two columns (e.g., the IF column and the THEN column) are displayed. After pressing the “Add New Constraint” button, the user can drag projects from the master list into each column. Equations for each type of constraint are displayed below to document the formulations that are encoded for LP_solve, the MILP optimization program used by OPTUM.

One important observation regarding notation for the constraint formulations is that variables such as “notA” or “notB” represent independent variables used for LP_solve inputs.

These variables *do not represent logical operators or operands* but are instead *simple programming variables that are defined by the equations*. Conditional constraints such as Types 13–24 make use of these types of variables.

Type 1: Include All or None

Logic Rule: Include either all projects or no projects in Column A.

Use this constraint type if you want to force PASS to either include a specific combination of projects (such as A, B, and C) in your optimal portfolio, or completely exclude all those projects, from your optimal portfolio.

Formulation: $A = B$
 $B = C$

Type 2: Include All

Logic Rule: Include all projects in Column A.

Use this constraint type if you want to force PASS to include a specific combination of projects (such as A, B, and C) in your optimal portfolio.

Formulation: $A = 1$
 $B = 1$
 $C = 1$

Type 3: Include at Least “n”

Logic Rule: Include at least “n” of the projects in Column A.

Use this constraint type if you want to force PASS to include “n” or more projects from a specific list of projects (such as A, B, and C).

Formulation: $A + B + C \geq n$

Type 4: Include Exactly “n”

Logic Rule: Include exactly “n” of the projects in Column A.

Use this constraint type if you want to force PASS to include exactly “n” projects from a specific list of projects (such as A, B, and C).

Formulation: $A + B + C = n$

Type 5: Include at Most “n”

Logic Rule: Include no more than “n” of the projects listed in Column A.

Use this constraint type if you want to force PASS to include no more than “n” projects from a specific list of projects (such as A, B, and C).

Formulation: $A + B + C \leq n$

Type 6: Exclude All

Logic Rule: Exclude all projects in Column A.

Use this constraint type if you want to force PASS to exclude a specific set of projects (such as A, B, and C).

Formulation: $A = 0$
 $B = 0$
 $C = 0$

Type 7: Exclude at Least “n”

Logic Rule: Exclude at least “n” projects in Column A.

Use this constraint type if you want to force PASS to exclude at least “n” projects from a specific set of projects (such as A, B, and C).

Formulation: $A + B + C \leq k - n$ (where k is the number of projects in exclusion list)

Type 8: Exclude Exactly “n”

Logic Rule: Exclude exactly “n” projects in Column A.

Use this constraint type if you want to force PASS to exclude exactly “n” projects from a specific set of projects (such as A, B, and C).

Formulation: $A + B + C = k - n$ (where k is the number of projects in exclusion list)

Type 9: If All ... Then All

Logic Rule: If all projects in Column A are selected, then select all projects in Column B.

Use this constraint type if one set of projects (such as A, B, and C) is the prerequisite of another set (full complement) of projects (such as D, E, F, and G). The “Conditional Constraints” columns for this example should look like this:

A If all of these projects are selected ...	B ... then select all of these projects.
D	A
E	B
F	C
G	

Formulation: $D+E+F+G-3 \leq A$
 $D+E+F+G-3 \leq B$
 $D+E+F+G-3 \leq C$

where 3 = (number of projects in Column A) – 1

Type 10: If at Least One ... Then All

Logic Rule: If at least one project in Column A is selected, then select all projects in Column B.

Use this constraint type if one set of projects (such as A, B, and C) is the prerequisite for any project in another set (such as D, E, F, and G). The “Conditional Constraints” columns for this example should look like this:

A If at least one of these projects is selected ...	B ... then select all of these projects.
D	A
E	B
F	C
G	

Formulation: $D+E+F+G \leq 4A$
 $D+E+F+G \leq 4B$
 $D+E+F+G \leq 4C$

where 4 = number of projects in Column A

Type 11: If at Least One ... Then at Least One

Logic Rule: If at least one project in Column A is selected, then select at least one project in Column B.

Use this constraint type if one set of projects (such as A, B, and C, of which at least one is mandatory) is the prerequisite of another set of projects (such as D, E, F, and G).

A If at least one of these projects is selected ...	B ... then select at least one of these projects.
D	A
E	B
F	C
G	

Formulation: $D+E+F+G \leq 4(A+B+C)$

where 4 = number of projects in Column A

Type 12: If All ... Then at Least One

Logic Rule: If all projects in Column A are selected, then select at least one from Column B.

Use this constraint type if one set of projects (such as A, B, and C, of which at least one is mandatory) is the prerequisite for another complete set of projects (such as D, E, F, and G).

A If all of these projects are selected ...	B ... then select at least one of these projects.
D	A
E	B
F	C
G	

Formulation: $D+E+F+G - 3 \leq A+B+C$

where $3 = (\text{number of projects in Column A}) - 1$

Type 13: If All ... Then Not All

Logic Rule: If all projects in Column A are selected, then do not select all (select some or none) projects in Column B.

A If all of these projects are selected ...	B ... then select some or none, but not all, of these projects.
D	A
E	B
F	C
G	

Formulation: $\text{not}A+A = 1$
 $\text{not}B+B = 1$
 $\text{not}C+C = 1$
 $D+E+F+G - 3 \leq \text{not}A+\text{not}B+\text{not}C$

where $3 = (\text{number of projects in Column A}) - 1$

Type 14: If at Least One ... Then Not All

Logic Rule: If at least one project in Column A is selected, then select some or none, but not all, projects in Column B.

A If at least one of these projects is selected ...	B ... then select some or none, but not all, of these projects.
D	A
E	B
F	C
G	

Formulation: $\text{not}A+A = 1$
 $\text{not}B+B = 1$
 $\text{not}C+C = 1$
 $D+E+F+G \leq 4 \text{ (not}A+\text{not}B+\text{not}C)$

where 4 = number of projects in Column A

Type 15: If at Least One ... Then None

Logic Rule: If at least one project in Column A is selected, then no projects from Column B will be selected.

A If at least one of these projects is selected ...	B ... then none of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}A+A = 1$
 $\text{not}B+B = 1$
 $\text{not}C+C = 1$
 $D+E+F+G \leq 4 \text{ not}A$
 $D+E+F+G \leq 4 \text{ not}B$
 $D+E+F+G \leq 4 \text{ not}C$

where 4 = number of projects in Column A

Type 16: If All ... Then None

Logic Rule: If all projects in Column A are selected, then no projects from Column B will be selected.

A If all of these projects are selected ...	B ... then none of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}A+A = 1$
 $\text{not}B+B = 1$
 $\text{not}C+C = 1$
 $D+E+F+G-3 \leq \text{not}A$
 $D+E+F+G-3 \leq \text{not}B$
 $D+E+F+G-3 \leq \text{not}C$

where 3 = number of projects in Column A – 1

Type 17: If Not All ... Then All

Logic Rule: If not all projects in Column A are selected, then all projects from Column B will be selected.

A If not all of these projects are selected ...	B ... then all of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}D+D = 1$
 $\text{not}E+E = 1$
 $\text{not}F+F = 1$
 $\text{not}G+G = 1$
 $\text{not}D+\text{not}E+\text{not}F+\text{not}G \leq 4A$
 $\text{not}D+\text{not}E+\text{not}F+\text{not}G \leq 4B$
 $\text{not}D+\text{not}E+\text{not}F+\text{not}G \leq 4C$

where 4 = number of projects in Column A

Type 18: If None ... Then All

Logic Rule: If no projects in Column A are selected, then all projects from Column B will be selected.

A If none of these projects are selected ...	B ... then all of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}D+D = 1$
 $\text{not}E+E = 1$
 $\text{not}F+F = 1$
 $\text{not}G+G = 1$

$$\begin{aligned} \text{not}D+\text{not}E+\text{not}F+\text{not}G-3 &\leq A \\ \text{not}D+\text{not}E+\text{not}F+\text{not}G-3 &\leq B \\ \text{not}D+\text{not}E+\text{not}F+\text{not}G-3 &\leq C \end{aligned}$$

where $3 = (\text{number of projects in Column A}) - 1$

Type 19: If None ... Then at Least One

Logic Rule: If no projects in Column A are selected, then at least 1 project from Column B will be selected.

A If none of these projects are selected ...	B ... then at least one of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}D+D = 1$
 $\text{not}E+E = 1$
 $\text{not}F+F = 1$
 $\text{not}G+G = 1$
 $\text{not}D+\text{not}E+\text{not}F+\text{not}G - 3 \leq A+B+C$

where $3 = (\text{number of projects in Column A}) - 1$

Type 20: If Not All ... Then at Least One

Logic Rule: If not all projects in Column A are selected, then at least 1 project from Column B will be selected.

A If not all of these projects are selected ...	B ... then at least one of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}D+D = 1$
 $\text{not}E+E = 1$
 $\text{not}F+F = 1$
 $\text{not}G+G = 1$
 $\text{not}D+\text{not}E+\text{not}F+\text{not}G \leq 4(A+B+C)$

where 4 = number of projects in Column A

Type 21: If Not All ... Then Not All

Logic Rule: If not all projects in Column A are selected, then not all projects from Column B will be selected.

A If not all of these projects are selected ...	B ... then some or none, but not all, of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}A+A = 1$
 $\text{not}B+B = 1$
 $\text{not}C+C = 1$
 $\text{not}D+D = 1$
 $\text{not}E+E = 1$
 $\text{not}F+F = 1$
 $\text{not}G+G = 1$
 $\text{not}D+\text{not}E+\text{not}F+\text{not}G \leq 4(\text{not}A+\text{not}B+\text{not}C)$

where 4 = number of projects in Column A

Type 22: If None ... Then Not All

Logic Rule: If no projects in Column A are selected, then not all projects from Column B will be selected.

A If none of these projects are selected ...	B ... then some or none, but not all, of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}A+A = 1$
 $\text{not}B+B = 1$
 $\text{not}C+C = 1$
 $\text{not}D+D = 1$
 $\text{not}E+E = 1$
 $\text{not}F+F = 1$
 $\text{not}G+G = 1$
 $\text{not}D+\text{not}E+\text{not}F+\text{not}G - 3 \leq \text{not}A+\text{not}B+\text{not}C$

where $3 = (\text{number of projects in Column A}) - 1$

Type 23: If None ... Then None

Logic Rule: If no projects in Column A are selected, then no projects from Column B will be selected.

A If none of these projects are selected ...	B ... then none of these projects will be selected.
D	A
E	B
F	C
G	

Formulation: $\text{not}A+A = 1$
 $\text{not}B+B = 1$
 $\text{not}C+C = 1$
 $\text{not}D+D = 1$
 $\text{not}E+E = 1$
 $\text{not}F+F = 1$
 $\text{not}G+G = 1$

$$\begin{aligned} \text{notD} + \text{notE} + \text{notF} + \text{notG} - 3 &\leq \text{notA} \\ \text{notD} + \text{notE} + \text{notF} + \text{notG} - 3 &\leq \text{notB} \\ \text{notD} + \text{notE} + \text{notF} + \text{notG} - 3 &\leq \text{notC} \end{aligned}$$

where $3 = (\text{number of projects in Column A}) - 1$

Type 24: If Not All ... Then None

Logic Rule: If not all projects in Column A are selected, then no projects from Column B will be selected.

A If not all of these projects are selected ...	B ... then none of these projects will be selected.
D	A
E	B
F	C
G	

Formulation:

$$\begin{aligned} \text{notA} + A &= 1 \\ \text{notB} + B &= 1 \\ \text{notC} + C &= 1 \\ \text{notD} + D &= 1 \\ \text{notE} + E &= 1 \\ \text{notF} + F &= 1 \\ \text{notG} + G &= 1 \\ \text{notD} + \text{notE} + \text{notF} + \text{notG} &\leq 4 \text{ notA} \\ \text{notD} + \text{notE} + \text{notF} + \text{notG} &\leq 4 \text{ notB} \\ \text{notD} + \text{notE} + \text{notF} + \text{notG} &\leq 4 \text{ notC} \end{aligned}$$

where $4 = \text{number of projects in Column A}$

ADDITIONAL SUPPORT MATERIAL FOR CONDITIONAL CONSTRAINT FORMULATIONS

The following material provides examples of verifications of conditional constraint formulations. These examples are shown for selected constraints to illustrate confirmation of the logic for several of the less obvious formulations.

Type 9: i and j are prerequisites for k and l .

$$l \wedge k \Rightarrow i \wedge j$$

Claim: The following equations capture the relationship above.

Let L be a binary set to 1 if l is selected, and 0 otherwise. Similarly K, I , and J are defined.

$$L + K - 1 \leq I$$

$$L + K - 1 \leq J$$

Proof: Only if $I = J = 1$ will the left-hand side of the inequalities be equal to 1, all other times the left-hand side will less than or equal to zero. Therefore the claim holds.

The idea above can be extended to n number of such constraints. Say that $i \wedge j$ are prerequisites for the conjunction n projects (p_1, \dots, p_n) . The following would be the inequality capturing this relationship:

$$\sum_{i=1}^n P_i - (n - 1) \leq I$$

$$\sum_{i=1}^n P_i - (n - 1) \leq J$$

Similarly if the conjunction of n projects (p_1, \dots, p_n) is the prerequisite for l and k , then the constraints are:

$$L + K - 1 \leq P_i \quad \forall i$$

Examples: $l \wedge k \Rightarrow i \wedge j$

$$L + K - 1 \leq I$$

$$L + K - 1 \leq J$$

$l \wedge k \wedge x \Rightarrow i \wedge j \wedge y$

$$L + K + X - 2 \leq I$$

$$L + K + X - 2 \leq J$$

$$L + K + X - 2 \leq Y$$

$l \wedge k \wedge x \wedge a \Rightarrow i \wedge j \wedge y$

$$L + K + X + A - 3 \leq I$$

$$L + K + X + A - 3 \leq J$$

$$L + K + X + A - 3 \leq Y$$

Type 10: i and j are prerequisites for k or l .

$$l \vee k \Rightarrow i \wedge j$$

Claim: The following equations capture the relationship above.

Let L be a binary set to 1 if l is selected, and 0 otherwise. Similarly $K, I,$ and J are defined.

$$L + K \leq 2I$$

$$L + K \leq 2J$$

Proof: We know that $I = J = 1$ if either $L = 1$ or $K = 1$ or both. This property is captured by the inequalities above. We see that $L = 1 \Rightarrow I = 1$ and $J = 1$, similarly $K = 1 \Rightarrow I = 1$ and $J = 1$, finally $L = 1$ and $K = 1 \Rightarrow I = 1$ and $J = 1$ as $I, J, K,$ and \bar{L} are binary variables.

The idea above can be extended to n number of such constraints. Say that $i \wedge j$ are prerequisites for the disjunction of n projects (p_1, \dots, p_n) . The following would be the inequality capturing this relationship:

$$\sum_{i=1}^n P_i \leq nI$$

$$\sum_{i=1}^n P_i \leq nJ$$

Similarly if the conjunction of n projects (p_1, \dots, p_n) is the prerequisite for k or l then the constraints are:

$$K + L \leq 2P_i \quad \forall i \in \{1, \dots, n\}$$

Examples: $l \vee k \vee x \Rightarrow i \wedge j \wedge y$

$$L + K + X \leq 3I$$

$$L + K + X \leq 3J$$

$$L + K + X \leq 3Y$$

$l \vee k \vee x \vee a \Rightarrow i \wedge j \wedge y$

$$L + K + X + A \leq 4I$$

$$L + K + X + A \leq 4J$$

$$L + K + X + A \leq 4Y$$

Type 11: i or j are prerequisites for k or l .

$$l \vee k \Rightarrow i \vee j$$

Claim: The following equations capture the relationship above.

Let L be a binary set to 1 if l is selected, and 0 otherwise. Similarly $K, I,$ and J are defined.

$$L + K \leq 2(I + J)$$

Proof: As the Boolean statement above suggests, if either L or K are 1 then either I or J or both must be 1. This relationship is captured by the inequality above. Note that by the inequality either I or J must be 1 if either L or K are 1.

The idea above can be extended to n number of such constraints. Say that $i \vee j,$ are prerequisites for the disjunction of n projects (p_1, \dots, p_n) . The following would be the inequality capturing this relationship:

$$\sum_{i=1}^n P_i \leq n(I + J)$$

Similarly if the disjunction of n projects (p_1, \dots, p_n) is the prerequisite for k or l then the constraints are:

$$K + L \leq 2 \sum_{i=1}^n P_i$$

Examples: $l \vee k \Rightarrow i \vee j$
 $L + K \leq 2(I + J)$

$$l \vee k \vee x \Rightarrow i \vee j \vee y$$

$$L + K + X \leq 3(I + J + Y)$$

$$l \vee k \vee x \vee a \Rightarrow i \vee j \vee y$$

$$L + K + X + A \leq 4(I + J + Y)$$

Type 12: i or j are prerequisites for k and l .

$$l \wedge k \Rightarrow i \vee j$$

Claim: The following equations capture the relationship above.

Let L be a binary set to 1 if l is selected, and 0 otherwise. Similarly $K, I,$ and J are defined.

$$L + K - 1 \leq I + J$$

Proof: Only if $I = J = 1$ will the left-hand side of the inequalities be equal to 1; all other times the left-hand side will less than or equal to zero. Therefore the claim holds.

The idea above can be extended to n number of such constraints. Say that $l \wedge k$ are prerequisites for the conjunction of n projects (p_1, \dots, p_n) . The following would be the inequality capturing this relationship:

$$\sum_{i=1}^n P_i - (n-1) \leq I + J$$

Similarly if the disjunction of n projects (p_1, \dots, p_n) is prerequisite for k and l , then the constraints are:

$$L + K - 1 \leq \sum_{i=1}^n P_i$$

Examples: $l \wedge k \Rightarrow i \vee j$
 $L + K - 1 \leq I + J$

$$l \wedge k \wedge x \Rightarrow i \vee j \vee y$$

$$L + K + X - 2 \leq I + J + Y$$

$$l \wedge k \wedge x \wedge a \Rightarrow i \vee j \vee y$$

$$L + K + X + A - 3 \leq I + J + Y$$

With respect to negation: Distribute the negation into the clause and define the artificial variables. Proceed with the concatenation procedures outlined above.

Example: This:

$$\neg(a \wedge b \wedge c) \Rightarrow i \wedge j \wedge k$$

is equivalent to:

$$\neg a \vee \neg b \vee \neg c \Rightarrow i \wedge j \wedge k$$

Define $\neg A$ such that $\neg A + A = 1$, and carry out the appropriate procedure above.

Negation of a conjunction:

$$\neg(a \wedge b \wedge c) \equiv \neg a \vee \neg b \vee \neg c$$

Negation of a disjunction:

$$\neg(a \vee b \vee c) \equiv \neg a \wedge \neg b \wedge \neg c$$



Decision and Information Sciences Division

Argonne National Laboratory
9700 South Cass Avenue, Bldg. 900
Argonne, IL 60439-4867

www.anl.gov



UChicago ▶
Argonne_{LLC}

A U.S. Department of Energy laboratory
managed by UChicago Argonne, LLC