3-1996

Optimization Analysis for Design and Planning of Multi-Project Programs

Victor D. Wiley

AFIT/GOR/ENS/96M-18

OPTIMIZATION ANALYSIS FOR DESIGN AND PLANNING OF MULTI-PROJECT PROGRAMS

THESIS

Victor D. Wiley, Captain, USAF

AFIT/GOR/ENS/96M-18

AFIT/GOR/ENS/96M-18

OPTIMIZATION ANALYSIS FOR DESIGN AND PLANNING OF MULTI-PROJECT PROGRAMS

THESIS

Presented to the Faculty of the School of Engineering
Air Education and Training Command
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Operations Research

Victor D. Wiley, B.S., M.S.
Captain, USAF

March 1996

AFIT/GOR/ENS/96M-18

THESIS APPROVAL

NAME: Victor D. Wiley, Captain, USAF **CLASS:** GOR-96M

THESIS TITLE: Optimization Analysis for Design and Planning of Multi-Project Programs

DEFENSE DATE: 29 February 1996

COMMITTEE:	**NAME/TITLE/DEPARTMENT**	**SIGNATURE**
Advisor	Richard F. Deckro Professor Department of Operational Sciences	
Reader	Jack A. Jackson, Jr., Lt Col, USAF Assistant Professor Department of Operational Sciences	

Acknowledgments

I am indebted to my thesis advisor, Dr. Richard Deckro. His insight and guidance during this research effort was invaluable. He provided motivation without limiting the learning from the freedom to explore. My special thanks to Dr. Deckro for his constant patience and understanding.

My thanks to Lt Col Jack Jackson, Jr. His comments and suggestions helped to keep the focus of this thesis on the reader.

I would like to thank my wife, Pamela, who provided support and encouragement during the thesis process. The countless hours she spent entertaining our son during this time were a godsend.

Victor D. Wiley

Table of Contents

List of Tables

List of Figures

Abstract

Program management concerns the long-term planning, coordination and control of major technological, engineering, scientific, and/or developmental activities. In general, programs tend to be exceptionally large, consisting of several parallel or sequential projects or groups of projects. While many of the approaches developed for projects have been used in program management, there are critical differences between programs and projects. A key difference is the magnitude of programs, both in program duration and number of tasks to be coordinated. In addition, project parameters are driven by program decisions.

While a large number of modeling efforts have focused at the project level, very little optimization literature deals directly with aspects of initial program design and development. This thesis effort looks at the application of optimization techniques to the *initial* design and development of multi-project programs. The classic work breakdown structure is used as a framework to provide an aggregate model to investigate the effects of funding levels, resource allocation, and program durations.

At the aggregate, program manager/planner level, the classic Dantzig-Wolfe block angular structure may be derived directly from the work breakdown structure. By varying program and project funding levels and durations, the program manager is provided with a decision support tool for designing and initializing the program and its projects. With the solution of a particular funding/duration scenario, sensitivity analysis is conducted to provide the program designer with insight into the inter-relations found in the system.

"OPTIMIZATION ANALYSIS FOR DESIGN AND PLANNING

OF MULTI-PROJECT PROGRAMS"

I. Introduction

Background

The United States Air Force (USAF) conducts and contracts out a vast array of programs composed of multiple projects. Air Force Materiel Command (AFMC) is responsible for managing every aspect of the USAF's weapon systems, from their inception on the drawing board through support during their operational lives, to final disposition. Because of this responsibility, AFMC conducts a majority of the multi-project programs within the USAF. To meet this and its other responsibilities, AFMC's fiscal budget for 1996 is $35.6 billion, approximately 49% of the USAF's *total* budget. The Aeronautical Systems Center (ASC) is one of four product centers within AFMC. ASC, as of May 95, had approximately 110 acquisition programs accounting for $10-12 billion dollars. Clearly, a material portion of the USAF resources are committed to large programs. To maintain our technological edge in the face of decreasing funding levels, it is imperative that the USAF's resources are well managed.

However, such large scale technological, development, and acquisition programs involve a vast number of activities and can be extremely complex. For example, Dowden describes a plant construction program that consists of thousands of activities, most requiring resources, constructed from detailed work breakdown structures (18:15). Complex R&D efforts such as a missile sytem or an aircraft could easily have millions of components and activities.

Introduction

Program management concerns the long term planning, coordination and control of major technological, engineering, scientific, and/or developmental activities. In general, programs tend to be exceptionally large, consisting of several parallel and/or sequential projects or groups of projects. While many of the approaches developed for project scheduling and control have been applied to program management, there are critical differences between establishing a program and scheduling a project to meet pre-established due dates with predetermined resource allocations. One key difference is the magnitude of programs, both in duration and the total number of tasks to be coordinated, when compared to a project. In addition, project parameters are driven by program decisions; these program decisions dictate due dates, resource levels, priority levels, and establish individual project funding levels.

While a large number of modeling efforts have focused at the project level, very little optimization literature deals directly with aspects of initial program design, planning, and development. This paper looks at the application of optimization techniques to the initial design, planning, and development of multi-project programs. The classic work breakdown structure (WBS) is used as a framework to provide an aggregate model to investigate the effects of funding levels, resource allocation, and program and project durations.

Aggregate Program Planning with the WBS

According to the Project Management Institute, program management involves the planning, organizing, staffing, directing, and controlling of a program to achieve its objectives (44:142). The responsibility for the program's planning, resource allocation, direction, and control are the duties of the program manager, who must determine the proper information and control system for coordinating all project activities (6:118). In production and operational settings, aggregate level planning is a hierarchical structure that parallels the management

decision process, reduces the amount of detailed information needed for planning purposes, and increases the accuracy of forecasts (5:22). This hierarchical planning structure utilized in production parallels the Work Breakdown Structure utilized in program and project management. The Work Breakdown Structure (WBS) used in project management is defined as the framework relating statements of work, contract line items, configuration items, technical and management reports, and the hardware, software, and data elements of the system (16:88). The WBS translates the results of the systems engineering analysis of the system requirements into a structure of the products and services which comprise the entire work effort (16:88), just as the aggregate production plan translate aggregated requirements into material, workforce, and capacity requirements in a production environment.

Kerzner states that the Work Breakdown Structure serves as the initial control from which all planning emanates (see Figure 1).

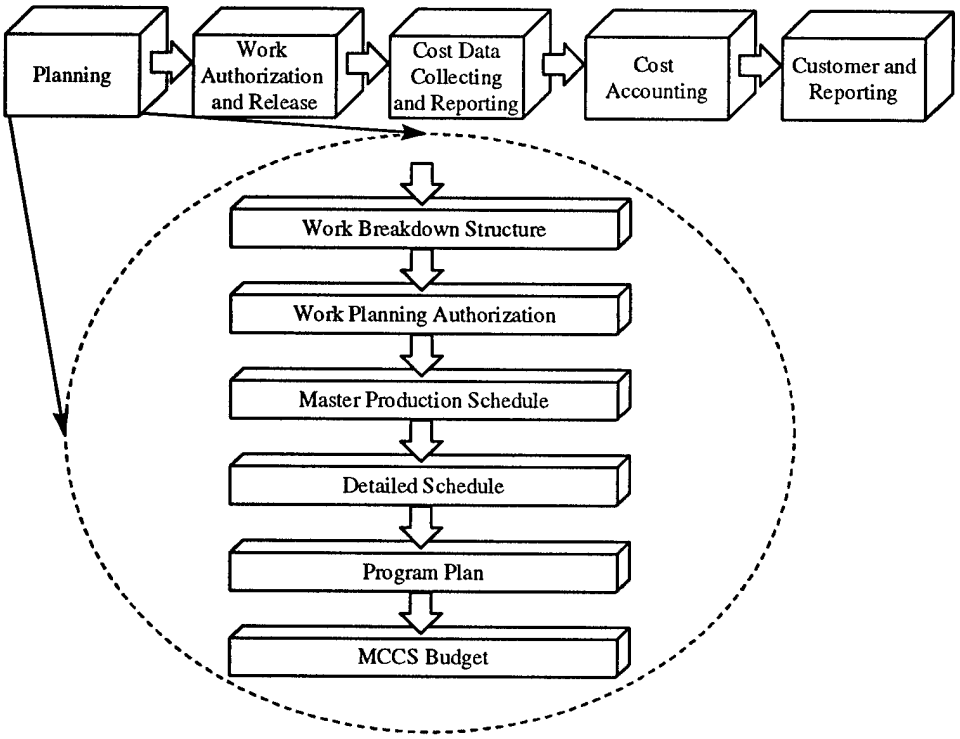

Figure 1. The Planning Cycle (31:604)

In this study, we have created a program aggregate planning model, based upon the WBS, to establish project parameters. We assume the analysis is being conducted in the early portion of the initiation stage of the program and is to be used to aid the program manager and his or her staff to establish program and project requirements. As in conventional production hierarchical planning, the variables will be assumed to be continuous at this aggregate level.

At the aggregate, program manager/planner level, the classic Dantzig-Wolfe block angular structure may be derived directly from the work breakdown structure. Program level requirements and resources serve as the overall binding constraints, while the individual project requirements provide the distinct blocks within the structure. By varying program and project funding levels and durations, the program manager is provided with a tool which supports the decisions undertaken in designing and initializing the program and its projects. With the solution of a particular funding/duration scenario, sensitivity analysis is conducted to provide the program designer with insight into the inter-relations found in the system.

If the cost and duration of each project, and the activities within the projects, are deterministic and fixed, then (assuming a feasible solution exists) the solution to the multi-project program involves simply minimizing the makespan till delivery of the program end-product, assuming there is adequate resources, flexibility, and funds to meet any program deadlines. However, project costs and activity durations for programs in the USAF tend to be dynamic as decision makers trade-off funds against activity durations.

Problem Statement

Kerzner lists 14 typical reasons why plans fail:

1. Corporate goals are not understood at the lower organizational levels,
2. Plans encompass too much in too little time,
3. Financial estimates were poor,
4. Plans were based on insufficient data,
5. No attempt was made to systematize the planning process,
6. Planning was performed by a planning group,
7. No one knows the ultimate objective,
8. No one knows the staffing requirements,
9. No one knows the major milestone dates, including written reports,
10. Project estimates are best guesses, and are not based on standards or history,
11. Not enough time was given for proper estimating,
12. No one bothered to see if there would be personnel available with the necessary skills,
13. People are not working toward the same specifications, and
14. People are consistently shuffled in and out of the project with little regard for schedule (31:605).

Kerzner points underscore the need for an analysis tool that provides the program decision makers with the ability to generate feasible program schedules, determine appropriate funding levels for each project, and to conduct sensitivity analysis on these schedules and funding levels *before* the establishing of a multi-project program. The need for this tool increases as the number of projects and the number of feasible project schedules within a program increases. With the large

number of possibilities available in setting up a program, a method for determining an effective resource allocation and schedule for all the projects within the program is desired. The method chosen should be able to deal with this large interconnected problem, as well as provide useful information to the decision maker. This useful information would include information such as the solution's sensitivity to funding levels and the ability to review effects of alternative program strategies.

This thesis provides a review of the literature associated with program management, aggregate level planning, multi-project and (large) single-project modeling, hierarchical planning, and decomposition in Chapter 2. Chapter 3 discusses the benefits to the decision maker of knowing information about feasible program schedules, appropriate funding levels, time/cost trade-offs, and sensitivity analysis. Chapter 3 presents the model and utilizes the Work Breakdown Structure to decompose the model into smaller more tractable problems. Chapter 4 presents an illustrative example of the model developed in Chapter 3, applies the decomposition methodology to this model, and discusses the results of decomposition on the model. Chapter 5 summarizes this paper's work and its conclusions and considers possible extensions to the model presented.

II. Literature Review

Introduction

Chapter I introduces the need for analytical tools to provide decision makers with the ability to generate feasible program schedules, determine appropriate funding levels for each project within a program, and to conduct sensitivity analysis upon these schedules and funding levels. While this study concentrates on the analytical techniques for program management, it is necessary to review literature from a number of areas to address the following:

1) What are the responsibilities of a program manager in establishing and directing a multi-project program?
2) What are the existing techniques for project scheduling? How have they been applied to the multi-project scheduling problem?
3) What analytical techniques have been developed for solving large-scale multi-project scheduling problems?

For this purpose, it is essential to survey the works to date on program management, aggregate level planning, multi-project and (large) single-project modeling, hierarchical planning, and decomposition.

Program Management

According to the Project Management Institute, program management involves the planning, organizing, staffing, directing, and controlling of a program to achieve its objectives (44:142). The responsibility for the program's planning, resource allocation, direction, and control are the duties of the program manager, who must determine the proper information and control system for coordinating all project activities (6:118).

The Department of Defense (DOD) defines a program manager as one who manages a directed, funded effort that is designed to provide a new or improved materiel capability in response to a validated need (16:279,292). Each program manager is responsible for making program

decisions and resource commitments based on plans for, and progress in, controlling risk using acquisition strategies and program plans (15:6). An acquisition strategy provides an event-driven master schedule designed to achieve program objectives, within the resource constraints imposed, while explicitly linking major contractual commitments and milestone decisions with demonstrated accomplishments in development and testing (16:279, 15:6). DOD Directive 5000.1 states that "program plans must provide for a systems engineering approach to the simultaneous design of the product and its associated manufacturing, test, and support processes" (15:6).

Program planning converts goals, objectives, demands, or problems into schemes, decisions, stated intentions, or solutions (37:1-1). Program planning influences the success, satisfaction, and productiveness of the item being planned during its useful life for a relatively small cost (37:1-6). Program planning permits the manager to see projects as parts to the whole program and provides a mechanism for the interrelated parts to be coordinated; thus, avoiding suboptimization of parts at the expense of the whole (43:36,37).

Managers make decisions on program planning through intuitive and/or analytical means. Analytical decision making techniques aid the intuitive-based decisions and situations where intuition fails (i.e. a program too large and complex to be handled without the support of analytical techniques). They do not, however, replace insight or experiences, rather they supplement it. By providing a structure for a situation, analytical techniques aid in making the best choice clearer to the decision maker (2:9).

The responsibilities ascribed to program managers for multi-project programs are akin to aggregate planning in operational settings. The following section reviews the use of aggregate level planning to develop a top-down hierarchical approach to program and project planning.

Aggregate Level Planning

The most complex problem can be divided into manageable terms by establishing a starting point, the precedence relationships of the critical variables, and the integrating stages of the problem (2:3).

Aggregate level planning provides a means of generalizing a large detailed program into distinct levels to simplify the analysis procedure (5:21). Bitran and Tirupati state that aggregation provides three advantages: 1) substantial savings in the costs of data collection for demand forecasting, 2) reduced variance in the aggregate demand forecast, and 3) increased managerial understanding of the key tradeoffs involved in the production decisions (4:532).

Candea states that aggregate level planning viewed as a hierarchical structure parallels the management decision process, reduces the amount of detailed information needed for planning purposes, and increases the accuracy of forecasts (5:22). Further, Iyer, Jarvis, and Ratliff state that a structure providing initial guidelines for aggregation, utilizing detailed plans for revising the aggregate plan, and iterating until an acceptable overall plan is obtained is necessary for coordinating the efforts of all participants and the decision making hierarchy (29:1).

Bates and Eldridge provide basic steps in dividing programs into distinct levels to include:

1) Divide the program activities into projects
2) Note the relationship between projects and any necessary sequences
3) Decide who is responsible for accomplishing each project
4) Determine the resources required by each project
5) Estimate the length of time for each project
6) Assign definite dates for each project (2:215)

Aggregate level planning has been shown to have many applications in manufacturing, production, and other operational settings, but very

little work has applied aggregate level planning directly to project scheduling, other than that captured in the Work Breakdown Structure.

Dincerler provides an example of aggregate level planning being applied to project scheduling for a multi-project environment (17:6). He applies a heirarchical approach to the problem because of its suitability to organizational structure and its ability to make the planning and scheduling of the individual projects within the multi-project environment easier to handle (17:31).

This section reviewed the similarities between aggregate level planning and hierarchical decision making and how these similarities have applied to the multi-project scheduling problem. The next section reviews the Work Breakdown Structure form and its similarities to both aggregate level planning and hierarchical decision making.

Work Breakdown Structure

According to Muther, "A simple breakdown of tasks constitutes a plan" (37:8-4). The Department of the Air Force's Acquisition Fundamentals course states "The common thread that ties all plans and documents together is the Work Breakdown Structure" (15:I-4-19).

The DOD describes the Work Breakdown Structure (WBS) as the framework relating statements of work, contract line items, configuration items, technical and management reports, and the hardware, software, and data elements of the system (16:88). The WBS translates the results of the systems engineering analysis of the system requirements into a structure of the products and services which comprise the entire work effort (16:88). The WBS shall:

1) Define the total system to be developed or produced;
2) Display it as a product oriented family tree composed of hardware, software, services, and data; and
3) Relate the elements of work to each other and to the end product (16:89).

The DOD requires a WBS for each program and for each individual contract within the program (16:89).

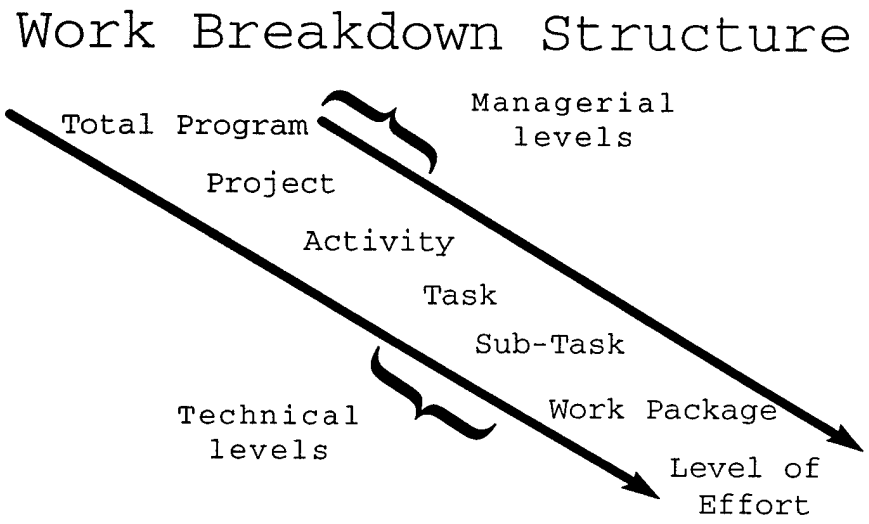

Figure 2. Work Breakdown Structure (31:592)

As described above, the WBS at the top levels (see Figure 1) generalizes a program into a hierarchical structure like aggregate level planning. The WBS defines the association between all of the elements within the program in much the same way as Bates and Eldridge (2) described in their basic steps for hierarchical decision making.

Despite the DOD requirement for WBSs and the similarities between aggregate level planning, hierarchical decision making, and the WBS, a review of the available literature has found virtually no documentation of project scheduling analysis efforts making use of the WBS with the exception of Deckro, Hebert, and Verdini (8).

Deckro et al illustrate the use of the WBS to facilitate the construction of a mathematical programming model (8:172). Their model assumes a linear time/cost tradeoff function for activity durations along with completion time requirements and budget limitations to find the least cost project schedule (8:171,172). Deckro and Hebert suggest

that the least cost schedule may be found by allowing some of the activities' durations to be reduced, or "crashed", by applying additional resources (money, personnel, equipment, and so forth) to the activities (11:72). Kelley discusses the mathematical basis of activity crashing in Critical Path planning and scheduling (30:298). Wiest and Levy also discuss project "crashing" and provide a simple example (47:62-70). Deckro et al suggest that the WBS can be used in the planning/proposal stage to aid in developing the schedules for large projects by using the higher levels of the WBS (program, project, component) to set the overall parameters for the lower levels (task, sub-task, activity) (8:177). Deckro et al also note that the WBS provides a model in a form (block diagonal) that decomposition techniques can effectively utilize (8:177). This suggestion will be built upon in this work.

This section reviewed the similarities of the WBS to aggregate level planning and hierarchical decision making. The next sections discusses multi-project and (large) single-project models.

Multi-Project and (Large) Single-Project Models

A large number of project scheduling and modeling techniques have been developed over the years. While all these approaches will not be explicitly discussed here, the key factors of these types of models will be reviewed.

Although different, most project scheduling techniques consider the overall program objective and the inter- and intrarelationships of the projects as well as the activities that make up the projects.

Examples of the types of objective functions include:

1) Minimize the completion time of the program (40:94)

2) Minimize the total program cost (8:176).

Examples of the types of inter- and intrarelationships between the projects include:

1) Limited resources (40:94)

2) Precedence relationships (40:94)

3) Project and Activity due dates (40:94).

The two objective functions above describe the classic time/cost trade-off that occurs when planning any program. The trade-off between time and cost creates a number of alternatives between the optimum strategies of minimizing program cost versus minimizing program duration (6:132). In a linear programming model, the time/cost trade-off is represented by a linear function that weights the benefits of decreasing program duration versus the costs associated with attaining the shorter program duration (47:79). Weist and Levy also describe the use of nonlinear time/cost trade-off curves (47:81). Deckro and Hebert also discuss various representational forms of the time/cost trade-off function (9).

The following sections will provide a brief description of several different approaches to solving project scheduling problems. While some of the older techniques described were developed before the advent of today's high speed computers, they are still valid and in use today. If, however, a program is of any significant size, the use of these older approaches becomes intractable without computerized mathematical programming techniques (47:73).

Program Evaluation and Review Technique (PERT). PERT, originally designed to plan and accelerate development of the Polaris ballistic missile, was an early attempt to capture the risk inherent in innovative technological development programs. PERT has been traditionally applied to research and development programs (47:41). PERT takes the uncertainties involved with research and development programs and calculates a weighted average of the expected duration of a project (47:42). PERT uses this weighted average along with its estimated

variance to determine the expected length of the program and the probability of completing the program by a certain date (47:51).

Critical Path Method (CPM). CPM, unlike PERT, tends to be applied to project scheduling problems where deterministic activity duration times are available (47:62). With the activity durations known, CPM focuses on the costs, both direct and indirect, associated with the completion of the program (47:62). CPM based models searches for the least costly schedule by utilizing time-cost trade-off functions for shortening or lengthening the durations of the various projects and activities within a program (47:64).

0-1 Integer Programming. Pritsker, Watters, and Wolfe suggest a model utilizing the types of objective functions and constraints discussed previously. They limit the decision variables within a project to be 1 if an activity ends during a particular time period, 0 otherwise (40:94). Pritsker et al. also describe a project level variable that contains a value of 1 when all the activities in a project are completed, 0 otherwise (40:95). This approach has the advantage of limiting the decision variables to 0-1 variables, but has the potential disadvantage of having to declare a 0-1 variable for each possible time period that a activity/project could be completed. If the time window for an activity completion is wide, many 0/1 variables are required. Sweeney and Murphy have noted that when a large number of integer variables are involved in a problem, the user is usually forced to accept a "good", rather than optimal, solution (45:1128).

Branch and Bound. Branch and Bound methods efficiently implicitly enumerate the combinations of values that the decision variables can obtain (48:502). Patterson describes a branch and bound procedure developed by Stinson that looks at precedence and resource constraint feasibility for subsets of the activities within a project (39:857). Stinson's procedure utilizes a series of tie-breaking rules for selecting the next candidate solution to enumerate and establishes

search origins based on the generation of complete schedules and improved bounds on the completion time (39:857).

Demeulemeester, Herroelen, Simpson, Baroum, Patterson, and Yang review a branch and bound procedure presented by Christofides, Alvarez-Valdes, and Tamarit that makes use of delay arcs for resolving resource conflicts for specific time periods within a schedule duration (12:219). The delay arcs result from the use of partial schedule sets that consider, for an instant in time, those activities still in process and their resource requirements conflicts (12:220). Demeulemeester et al. note that this procedure is highly effective but cite by counterexample that the procedure does not guarantee optimality (12:222). Demeulemeester et al. provide a modification to the Christofides, Alvarez-Valdes, and Tamarit procedure to ensure that all appropriate partial schedules are contained in the solution space (12:223).

Demeulemeester and Herroelen offer another branch and bound procedure similar to the Christofides, Alvarez-Valdes, and Tamarit procedure; however, Demeulemeester and Herroelen's branch generation and node selection differ significantly (13:1807). The difference involves the evaluation of a set of minimal delaying alternatives to compute a critical sequence lower bound that allows for the fathoming of a relatively large number of nodes during backtracking (13:1808).

Heuristic Solution Procedures. Kurtulus and Davis list 9 scheduling rules for the multi-project problem (32:165). These rules provide a means to schedule jobs within the various projects based on their durations and resource requirements (32:165). Kurtulus and Davis also devised two summary statistics, Average Resource Load Factor (ARLF) and Average Utilization Factor (AUF), to measure and compare the performance of the 9 scheduling rules against each other while solving the same problem set (32:162,163). These statistics, ARLF and AUF, account for the differences between single-project and multi-project problems by measuring the resource profiles obtained by scheduling each

activity at its early start time (32:162). A resource-constrained single-project scheduling problem consists of a number of activities scheduled according to the appropriate resource limitations, precedences, and objective function. A resource-constrained multiple-project scheduling problem consists of two or more single-project scheduling problems that have additional precedence relationships and competition over resources that must be accounted for in the overall schedule produced.

These sections reviewed just a few of the existing project scheduling techniques and how they have been applied to the scheduling problems. The following section reviews some of the various decomposition techniques for solving large-scale problems.

Decomposition

The need to solve large mathematical programs that could not be effectively solved by direct methods, i.e. simplex, led to the development of indirect methods such as decomposition (33:105). Although the computational power of computers has dramatically improved, there still exist very large problems which are intractable, for practical purposes, without the use of decomposition. In addition, if the purpose of mathematical programming is insight, not numbers as Geoffrion points out, decomposition can provide a synergy in its analysis steps which may be lost in direct methods (22:81)

Geoffrion describes various decomposition manipulations (Projection, Inner Linearization, Outer Linearization) and strategies (Piecewise, Restriction, Relaxation) to overcome the size problem of large mathematical programs (21:657,665). Geoffrion also compares these manipulations and strategies to existing decomposition techniques (21:676). A review of more recent literature shows work with two of the techniques mentioned by Geoffrion, Dantzig-Wolfe and Benders', as well as Lagrangean, Cross, and Time-based decomposition techniques. Each of these methods and their applications will be briefly outlined.

Time-Based Decomposition. Time-based decomposition approaches use time as a means of decomposing the original problem into smaller subproblems and a master problem. Edmunds and Bard state that the need for time based decomposition in optimal control problems often arises from discrete approximations to continuous formulations (19:61). Nagurney and Kim use time-based decomposition to create subproblems with existing efficient solution algorithms and solve these subproblems using parallel processors (38:56). Haurie focuses on time-based decomposition between part-surplus dynamics and operational state dynamics (24:350). These articles present the utilization of time-based decomposition as a result of the natural time related function in dynamic programming models (19:64, 38:59-60, 24:339).

Lagrangean Relaxation. Lagrangean decomposition is a technique that shows, if the Lagrangean function of a problem is additively separable, the problem may be decomposed by LaGrange multipliers (33:396). Michelon and Maculan use Lagrangean decomposition to solve integer nonlinear programming problems with linear constraints (36:303). Reinoso and Maculan show Lagrangean decomposition applied to integer linear programming to provide at least as good bounds as the classical Lagrangean relaxation with the use of just a small number of dual variables (41:5). Of more importance to this thesis is the work of Vercellis, who extended Lagrangean decomposition to the constrained multi-project planning problem (46:271). Vercellis demonstrated that Lagrangean decomposition provides for an explicit coordination between different projects to allocate resources, trade-off between the use of resources, and time and cost trade-offs for activity alternatives by interpreting the Lagrangean multipliers as marginal prices for the allocations of the resources to each project in each time period (46:274).

Dantzig-Wolfe Decomposition. Dantzig-Wolfe decomposition decomposes a problem into a number of subproblems and a master problem by separating the constraints of the original problem (33:153-154). Ho and Loute suggested that, until fairly recently, the use of Dantzig-Wolfe decomposition could not outperform the simplex approach (26:304). They suggest that a decomposition code must be able to solve the master and subproblems, handle the updating and processing of the master and subproblems, and provide a number of computational strategies (26:307). Ho and Loute show the benefits of an advanced Dantzig-Wolfe decomposition implementation by applying the Dantzig-Wolfe technique to a variety of test problems (26:323-324). They further test the robustness of their code and conclude that decomposition will provide better efficiency than the simplex approach when dealing with very large problems (25:289).

Benders' Decomposition. The Benders' decomposition technique partitions a problem into two parts, usually determined by the variable type, i.e. discrete and continuous (33:371). By fixing the discrete variables, the resultant problem is easier to solve (33:372). Rewriting the problem in terms of a fixed set of discrete variables and taking the dual of this problem allows the problem to be expressed as a minimization (maximization) master problem with constraints defined by the extreme points and rays of the feasible set of solutions (33:374). For each fixed discrete variable set, the resultant linear subproblem solution is checked against an optimality test which, if failed, produces a new constraint within the master problem (33:378). Benders' approach has been utilized in a number of settings. For example, Geoffrion and Graves apply Benders' decomposition to determine the optimal location for distribution facilities (23:822). Erengue, Tufekci, and Zappe modify Benders' decomposition to solve time/cost trade-off problems with discounted cash flows (20:30).

Cross Decomposition. Cross Decomposition, introduced by Van Roy in 1983, combines the Benders' and Dantzig-Wolfe Decomposition techniques to solve a series of primal and dual based subproblems for block-angular linear problems (1:395). Aardal and Ari compare cross decomposition and the Kornai-Liptak algorithm and note that the main difference between the two involves a weighting scheme employed by the Kornai-Liptak algorithm on the solution process (1:396). Of interest is Aardal's and Ari's suggestion that the use of an arithmetic mean in the solution information may provide the convergence guarantee missing from cross decomposition (1:398). Using this suggestion as impetus, Holmberg presents a method named Mean Value cross decomposition in which he proves convergence utilizing the mean value of the solutions found from the primal and dual subproblems (28:61). Holmberg further extends the merits of cross decomposition by showing that it can be used to provide good lower bounds for pure integer programming problems (27:657). Lee provides another application of cross decomposition in a specialized algorithm to the mixed integer problem associated with a multiproduct-multitype facility location problem (34:535). Lee utilizes the same algorithm and approach to a location and allocation problem in distributed computer network systems (35:371).

Applications to Program Management. The amount of literature on decomposition techniques is extensive; however, very few of the works (10,46) found apply decomposition approaches directly to Program Management.

In the establishing of a program, several advantages can be gained from using decomposition techniques. Not only can an optimal or near-optimal solution to the project scheduling problem be found in a computationally tractable time, but sensitivity analysis can be conducted on this solution. In addition, Baumol and Fabian describe how the suboptimal solutions found during each iteration of the decomposition procedure can be used to gain further knowledge about the

cost tradeoffs being made (3:6). Baumol and Fabian's statement on marginal profitability suggest that the simplex multipliers can represent the decrease in program cost that results from an increase in the amount of a resource (3:6). Similarly, their statement on provisional opportunity costs suggest that the simplex evaluators can represent the decrease in program cost that results from the introduction of a new schedule (3:6). Both the simplex multipliers and evaluators provide the opportunity to determine what resources and what critical paths within the schedules are contributing the most to the program costs.

These sections reviewed some of the decomposition approaches and their applications to the multi-project scheduling problem.

III. Model Formulation

Introduction

The preceding chapters discussed the need for an analysis tool to support program design by generating feasible program schedules, determining appropriate funding levels, providing time/cost trade-off information, and conducting sensitivity analysis.

Feasible Program Schedules. Feasible program schedules are a result of resource (money, people, equipment and so forth), time, and precedence limitations imposed by the program.

A Work Breakdown Structure (WBS) provides a simple means of identifying the network interrelations within a program. For any sizable DOD program, the WBS is required and available.

The time limits, or due dates, on a program represent the urgency with which a program needs to be accomplished. The time limits can be imposed at a number of levels. Time limits can be placed on the program completion, on important milestones, or on critical activities.

The resource limits on a program represent the fact that most resources are not available in unlimited supplies. For DOD programs, there is a limit on the amount of funding that Congress will appropriate. The funding limit can affect the time table of a program by reducing the amount of funding available for shortening critical activities.

Appropriate Funding Levels. Appropriate program funding levels need to be established in order to have enough funding to complete the program on time. In addition, adequate reserves must be kept to deal with any problems that arise. Also, appropriate program funding levels should not be set so high as to tie up funds that could be made available to other programs.

Time/Cost Trade-off Information. The time/cost trade-off information for a program is critical. The time/cost trade-off curve

associated with an activity indicates the change in program cost as a function of activity duration (20:25). In finding an optimal program schedule, the time/cost trade-off curves are used to determine the activity durations that minimize the overall cost to the program (20:26).

The model developed in this chapter will make use of the time/cost trade-off information to answer questions like:

1) How much shorter/longer will the program take if funding levels increase/decrease?
2) How much does the cost profile (cash flow) change if the time table for the program is altered?

Whichever question is being asked, the bottom line is if a program schedule can not be met with the current level of resources, either more resources are acquired (cost) or the program is delayed (time).

Sensitivity Analysis. Sensitivity analysis includes time/cost trade-off information, but also looks at how the program is affected by changes to the original constraints and coefficients of the model. Looking at the sensitivity of the original set-up of the program identifies areas that may require additional research, e.g. if the cost of overhead significantly affects the program schedule, then it may be worthwhile to invest additional time and money to determine the nearest approximation of the actual cost of overhead per time unit. By the same token, if a particular activity is critical under a number of scenarios, the estimation of that activity's duration and resource needs may merit more careful analysis.

An analysis tool that can investigate the questions outlined above could be utilized to estimate effects on program costs and duration due to funding and other resource constraints before a program has been established. Such information would be valuable in setting contract due dates and funding levels on the individual projects and activities that make up the program. "What if" questions could be addressed concerning

personnel, material, milestone, and funding questions all *before* establishing the program.

Chapter II reviewed the works to date on program management, aggregate level planning, multi-project and (large) single-project models, hierarchical planning, and decomposition.

A linear programming model, based on the works reviewed, will now be formulated to provide an analysis tool that meets the aforementioned needs. A description of the model will first be presented, followed by the model assumptions and the basic notation of the model. Next, the general form of the model will be constructed. From the general form presented, the Dantzig-Wolfe Decomposition technique will be applied to form a Master problem and Subproblems.

Model Description

The model is a linear program that represents a multi-project program that could range from anything as simple as the illustrative example presented in Chapter IV to as complex as the development of a new aircraft or other weapons system. From the basic model, the effects of various planning decisions required to establish a program may be investigated.

Since we are looking at the pre-program planning stage, the model represents a top-level aggregate model of the first three levels of the Work Breakdown Structure (see Figure 2).

The model objective is to minimize the overall cost of the program while meeting all of the program requirements and constraints.

The program and project structures are precedence network relationships between the various activities that can be attained from the Work Breakdown Structure. Activities within the project and program networks have normal duration times and costs associated with them as well as "crashed" duration times and costs.

The normal duration of an activity occurs when the usual, or normal, amount of resources required by that activity are met. The

normal cost of an activity is the cost of the normal amount of resources associated with an activity. Activities that are "crashed" are activities that have had additional resources (funding, personnel, equipment, and so forth) allocated to them for the purpose of shortening the duration time of the activity.

The model contains resource constraints at the program and project level. The resource constraints provide an upper bound on the availability of each type of resource to the program and to each project. The resources, at this aggregate level, are chosen to be represented by continuous linear variables to maintain the linearity of the model itself.

The program has a defined target completion date (TCD) and absolute completion date (T). To meet these due dates and comply with the objective function, the model must determine the most cost efficient schedule for "crashing" activities.

Model Assumptions

In setting up the program planning model, the following assumptions are made:

1) The normal activity duration times are known and constant,
2) Activity crashing is allowed up to an upper bound (M_{ij}) for each activity (i,j),
3) The time/cost tradeoff function for activity "crashing" will be modeled as a linear relation at this aggregate level, and
4) All variables will be nonnegative.

Model Notation

An adaptation of the notation presented by Wiest and Levy (45:80) will be used to set up the model. The identification of an event node, X_i, to a project is given as part of the i subscript, i.e. i=A1 represents the first event in project A. Let:

X_i = realization time of node i

T_{ij} = normal duration of activity (i, j)

D_{ij} = the cost of activity (i, j) at normal duration

Y_{ij} = number of time units activity (i, j) crashed

K_{ij} = cost per time unit activity (i, j) crashed (Y_{ij})

M_{ij} = upper bound on Y_{ij}

B_k = the budget for project k

B = the budget for the program

OH = overhead cost per time unit

T = total program time allowed

TCD = target completion date $(TCD \leq T)$

n = the terminal node for the program

EB = # of time units program finishes early

eb = reward bonus per time unit for finishing early

I_k = the set of connected arcs (i, j) in project k

I = the set of connected arcs (i, j) in the program

PH_l = the program upper limit on additional hours for type l personnel

PH_{kl} = the project k upper limit on additional hours per type l personnel

S_{ijl} = the type l personnel cost associated with "crashing" activity (i, j)

General Model

Objective Function. The objective function, depending on the nature of the project, can take several forms. The first objective function form determines the least costly program schedule. This form includes a fixed overhead cost, the cost of "crashing" activities, and a reward or bonus for finishing the program early. The normal activity

costs, $\sum_i \sum_j D_{ij}$, a fixed amount, have been excluded from this relation as it does not affect the optimization.

1) Minimize the program cost

$$z = OH \cdot X_n + \sum_i \sum_j K_{ij} Y_{ij} - eb \cdot EB$$

However, since *EB=TCD-XN*, we have, in the objective function above, *-eb(TCD-XN)= -eb(TCD) + eb(XN)*. The constant *-eb(TCD)* does not affect the optimization and so will be eliminated to yield the following objective function form:

$$z = (OH + eb) \cdot X_n + \sum_i \sum_j K_{ij} Y_{ij}$$

The first term represents the net of overhead and any early bonus when the program is completed by Xn while the second term represents any funds expended to accelerate or "crash" any particular activity *j* of project *i* in the program.

The second objective function form, unlike the first form, determines the shortest program duration possible without regard to cost.

2) Minimize the program duration

$$z = X_n$$

Since the objective function will vary depending on the analysis to be conducted with the model, these relationships may also be constraints in the model.

Budget Constraints. The budget constraints for each project limit the amount of total dollars spent on activity resources within that project. The costs of applying additional resources to "crash" activities within a *project* are limited by the project budget and the normal costs, $\sum_i \sum_j D_{ij}$, associated with performing the activities. This constraint will be of the following form:

$$\sum_i \sum_j K_{ij} Y_{ij} + \sum_i \sum_j D_{ij} \leq B_k \qquad \forall k \text{ and } (i,j) \in I_k .$$

Since the normal costs of the activities are fixed amounts, we rewrite this constraint by moving the normal activity costs over to the right-hand side of the equation as follows:

$$\sum_i \sum_j K_{ij} Y_{ij} \leq B_k - \sum_i \sum_j D_{ij} \qquad \forall k \text{ and } (i,j) \in I_k$$

Similarly, the overall program budget will put a limit on the total dollars spent by the program. The program budget will limit the costs of performing the activities normally and "crashed" as well as the additional overhead. The constraint, with normal activity costs placed on the right-hand side, will be of the following form:

$$OH \cdot X_n + \sum_i \sum_j K_{ij} Y_{ij} \leq B - \sum_i \sum_j D_{ij} \text{ for } (i,j) \in I$$

Personnel-Month Constraints. The personnel-month constraints for the program and the projects will limit the amount of additional personnel-months available for "crashing" activities. The number of personnel-months required by each activity for various classifications

of personnel when crashed is determined as a function of the "crash" cost associated with each activity. The associated project constraint would be as follows:

$$\sum_i \sum_j \sum_l S_{ijl} Y_{ij} \leq PH_{kl}, \quad \forall k, l \text{ and } (i,j) \in I_k$$

The program personnel-month constraint will be similar to the project personnel-month constraint and will be as follows:

$$\sum_i \sum_j \sum_l S_{ijl} Y_{ij} \leq PH_l, \quad \forall l \text{ and } (i,j) \in I$$

Due Date Constraints. One of the main concerns of most programs is finishing the program by a set due date, *TCD*. In establishing the program due date, the program can be solved using the second objective function, minimizing program duration. Once the minimum possible program duration is known, an appropriate *TCD* can be set. Otherwise, if the constraint below is in the model and an infeasible solution occurs, it is unknown if the infeasibility was caused by this constraint or some other constraint. To ensure that the solutions generated by the model occur on or before the *TCD*, the following constraint is included in the model:

$$X_n \leq TCD.$$

Similarly, each project can be given a due date constraint based on the last node and activity(ies) associated with that project.

Precedence Constraints. In most programs, there are some projects or activities that must be completed before others can begin. For example, in the production of an aircraft, the wings must be attached to

the fuselage before the engines are mounted on the wings. To model finish-to-start relationships as described above, the event time of an activity j, X_j, beginning must occur after the event time of activity i, X_i, beginning plus the duration time of activity (i,j), T_{ij}. When activity "crashing" is permitted, the amount of time an activity is "crashed", Y_{ij}, is subtracted from the normal activity duration time, T_{ij}. The following constraint models this occurrence:

$$- X_i + X_j \geq T_{ij} - Y_{ij} \text{ for } (i,j) \in I_k \ \forall k$$

Rewriting this constraint with all of the variables on the left side and the constant terms on the right hand side gives the following constraint:

$$- X_i + X_j + Y_{ij} \geq T_{ij}, \quad \text{for } (i,j) \in I_k \ \forall k.$$

The program precedence constraints would contain the project network as well as any program specific precedence constraints. The constraint for the model then becomes:

$$- X_i + X_j + Y_{ij} \geq T_{ij}, \quad \text{for } (i,j) \in I$$

At this point all of the constraints for the general model have been developed and we will put it all together. We first repeat the notation and then give the model in its entirety.

X_i = realization time of node i

T_{ij} = normal duration of activity (i, j)

D_{ij} = the cost of activity (i, j) at normal duration

Y_{ij} = number of time units activity (i, j) crashed

K_{ij} = cost per time unit activity (i, j) crashed (Y_{ij})

M_{ij} = upper bound on Y_{ij}

B_k = the budget for project k

B = the budget for the program

OH = overhead cost per time unit

T = total program time allowed

TCD = target completion date $(TCD \leq T)$

n = the terminal node for the program

EB = # of time units program finishes early

eb = reward bonus per time unit for finishing early

I_k = the set of connected arcs (i, j) in project k

I = the set of connected arcs (i, j) in the program

PH_l = the program upper limit on additional hours for type l personnel

PH_{kl} = the project k upper limit on additional hours per type l personnel

S_{ijl} = the type l personnel cost associated with "crashing" activity (i, j)

$$\text{Min} \quad z = (OH + eb) \cdot X_n + \sum_i \sum_j K_{ij} Y_{ij} \tag{1}$$

$$\text{s.t.} \quad OH \cdot X_n + \sum_i \sum_j K_{ij} Y_{ij} \leq B - \sum_i \sum_j D_{ij} \quad \text{for } (i, j) \in I \tag{2}$$

$$\sum_i \sum_j K_{ij} Y_{ij} \leq B_k - \sum_i \sum_j D_{ij} \quad \forall k \text{ and } (i, j) \in I_k \tag{3}$$

$$\sum_i \sum_j \sum_l S_{ijl} Y_{ij} \leq PH_l, \quad \forall l \text{ and } (i, j) \in I \tag{4}$$

$$\sum_i \sum_j \sum_l S_{ijl} Y_{ij} \leq PH_{kl}, \quad \forall k, l \text{ and } (i,j) \in I_k \tag{5}$$

$$X_n \leq TCD. \tag{6}$$

$$-X_i + X_j + Y_{ij} \geq T_{ij}, \quad \text{for } (i,j) \in I \tag{7}$$

$$X_j, Y_{ij} \geq 0 \tag{8}$$

Linear Programming and Parametric Programming

At this point, the model could be put into a linear optimization code and solved using direct methods such as the simplex method of linear programming.

Direct methods such as the simplex method search for the optimal solution to a mathematical model by starting at an initial basic feasible solution. The value of the current objective function at the initial basic feasible solution is compared with the values of the objective functions associated with the adjacent extreme points within the solution space for the model (33:31).

For very large and complex problems, the time spent by the solver searching all of the possible adjacent extreme points at each new basic feasible solution can take an intractable amount of time. Because of this, indirect methods for solving very large and complex methods provide a computational advantage over direct methods (33:105).

However, linear programming does provide a feature that could provide additional useful information to a decision maker. This feature is parametric analysis.

At the beginning of this chapter, sensitivity analysis, in relation to the right-hand side values of the constraints, was described as the range that a right-hand side could be within without causing the optimal basis to change. Parametric analysis determines how the optimal basis of a problem changes in response to changes in the right-hand side values beyond the bounds given by the sensitivity analysis (48:218).

The optimal bases associated with the new values of the right-hand sides provide additional information about the influence of program resource levels on the overall program. The bases provide alternate feasible program schedules that a decision maker could to determine the best overall program schedule.

Dantzig-Wolfe Decomposition Problem Formulation

Background. Decomposition is an indirect method for solving large scale problems. Decomposition is used to partition large problems into smaller, more tractably solved problems. Dantzig-Wolfe Decomposition separates a problem into a single master problem and several subproblems. The classic problem style for Dantzig-Wolfe Decomposition is the Block Angular design (see Figure 3). In the following paragraphs, the development of the Dantzig-Wolfe Master Problem and Subproblems will follow Lasdon's (33) presentation of Dantzig-Wolfe Decomposition.

Block-Angular Design

Minimize $z=c_1x_1+c_2x_2+\ldots+c_px_p$

subject to

$$\begin{aligned} A_1x_1+\ A_2x_2+\ldots+A_px_p &= b_0 && \text{(Program)}\\ B_1x_1 &= b_1 && \text{(Project 1)}\\ B_2x_2 &= b_2 && \text{(Project 2)}\\ &\ldots\\ B_px_p &= b_p && \text{(Project p)} \end{aligned}$$

where $x_t \geq 0$, for $t=1,\ldots,p$

x_u, b_u are column vectors for $u=0,\ldots,p$

c_t are row vectors

A_t, B_t are matrices

Figure 3. Block-Angular Design Structure

The Dantzig-Wolfe Subproblems contain sets of constraints from the original problem (Base Model) that are unique to each subproblem, i.e. the constraints for each subproblem do not contain variables included in any other subproblem. For a multi-project program being represented at the Managerial level of the Work Breakdown structure, a Dantzig-Wolfe Subproblem would contain the variables and constraints associated with a particular project of the program and only that project. The project itself would consist of an activity network with project level requirements and constraints.

The objective function for a subproblem (project) contains the original cost coefficients, from the program objective function, associated with the variables of that subproblem (project). These coefficients are adjusted at each iteration by subtracting off the multiple of the dual (shadow) prices from the Master Problem current optimal solution and the coefficients of the subproblem's (project's) variables located in the common (program) constraints of the Master Problem. While the common constraints will be defined in the next paragraph, the objective function form of the subproblems, using Figure 3, will be as follows:

$$\text{Minimize} \quad (c_t - \pi A_t)x_t$$

where

π = the dual (shadow) prices associated with the common constraints

The Dantzig-Wolfe Master Problem contains the common (program level) resource constraints that are to be shared among the distinct subproblems (projects). These constraints, however, have been transformed using the following convexity theorem:

Let $X_t=\{x_t / A_t x_t=b_t,\ x_t \geq 0\}$ be nonempty and bounded for $t=1,\ldots,p$ and let $x_t^i, i=1,\ldots,r_t$ be extreme points (finite #).

Then any element of X_t may be written as a convex combination of the extreme points, x_t^i (32:146).

In the theorem, the x_t^i term represents the ith extreme point, or basic feasible solution, associated with subproblem t. For this model, x_t^i represents the ith bid, or proposal, for utilizing program resources from project t to the program coordinator (the Master Problem).

This theorem states that any point in a set can be defined by the set's extreme points. This is similar to being able to define any point on a line as a combination of line's endpoints. To maintain the convexity conditions being utilized, a number of additional convexity constraints equal to the number of subproblems have been introduced to the Master Problem. Using the Block-Angular Design from Figure 3 and implementing this theorem allows us to rewrite the program-level constraints of Figure 3 as:

$$\sum_j \left(A_t x_t^j\right)\lambda_{tj} = b_0 \text{ for } t = 1,2,3$$

The x_t^j are the extreme points (proposals) generated by the Subproblems (projects) and the $A_t x_t^j$ are merely constant terms in the Master Problem. These constant terms represent the amount of program resources used by the jth proposal of project t. The λ_{jt}'s in the rewritten constraints above are variables which represent the weights associated

with each of the extreme points from a given subproblem. To maintain the convexity requirement for these constraints, the following convexity constraints, mentioned above, are added to the Master Problem:

$$\sum_j \lambda_{tj} = 1 \text{ for } t = 1,2,3$$

$$\lambda_{tj} \geq 0$$

Similarly, the objectives coefficients associated with the Subproblems variables are transformed into the following:

$$\sum_j \left(c_t x_t^j\right)\lambda_{tj} \text{ for } t = 1,2,3$$

where $c_t x_t^j$ are merely constant terms in the Master Problem. Thus, the general form of the Dantzig-Wolfe Master Problem would be as shown below in Figure 4.

$$\text{Minimize} \quad \Sigma_t \Sigma_j (c_t x_t^j)\lambda_{tj}$$

$$\text{subject to} \quad \Sigma_t \Sigma_j (A_t x_t^j)\lambda_{tj} = b_0$$

$$\Sigma_j \lambda_{tj} = 1 \quad \text{for } t=1,\ldots,p$$

$$\lambda_{tj} \geq 0$$

Figure 4. Master Problem General Form

Master Problem. The Master Problem construction follows the classic style presented by Dantzig and Wolfe (7) and reiterated above. One exception from the original form presented by Dantzig and Wolfe will be the allowance of $\leq$ or $\geq$ signs in the common constraints. The original form of the Dantzig-Wolfe Master problem contained strictly equality constraints; however, current linear solvers automatically convert inequality constraints into equality constraints by adding slack, excess, and artificial variables as necessary. Therefore, the use of inequality constraints is permitted. The initial solutions generated from the Subproblems will be entered as columns in the Master Problem. The Master Problem is then solved to find the new level of prices for candidate solutions. These prices are then passed back to the Subproblems, creating a new objective function which is then resolved.

However, the implementation of Dantzig-Wolfe decomposition in the Master Problem requires a starting basic feasible solution which may not always be easily attained. In a large program, the combination of feasible project schedules that provides a feasible program schedule may not be readily available. In fact, the process of finding a feasible starting program schedule may be as difficult as finding the optimal program schedule. To account for this, a "deviational" variable, *dev*, has been added at an appropriately high objective function cost (Big M for example) to the overall budget constraint to drive the subproblem solutions generated at each iteration toward feasibility until a basic feasible solution is obtained. In effect, *dev* has relaxed the budgetary constraint. As long as the precedence relations and activity durations will allow the program to complete by T, a solution (which may not be budgetarily feasible) will be attained. This addition avoids the necessity of searching for feasible solutions to the subproblems to initiate the decomposition process as the objective function weight will force *dev* out if possible. As we will see, by adding the deviation to

the budget, it also provides an opportunity to examine potential schedules which are not feasible for the current proposed funding level. Once a basic feasible solution is identified, the deviational variable is forced out of the basis by the Master Problem. Thus, continuing Lasdon's (33) presentation, the initial form of the Dantzig-Wolfe Master Problem will be as follows:

$$\text{Minimize} \quad z = \sum_k \sum_i \left(c_k x_k^i\right) s_{ik} + \left(OH + eb\right) \cdot XN\left(+ M \cdot dev\right)$$

subject to

$$\sum_k \sum_i \left(a_k x_k^i\right) s_{ik} \left(+ dev\right) \left(\leq = \geq\right) b_m, \quad \text{for } m = 1, \ldots, p$$

$$\sum_i s_{ik} = 1, \quad \forall k$$

$$s_{ik} \geq 0$$

where

p = the number of coupling constraints
c_k = the objective function coefficients associated with project k
a_k = the coupling constraint coefficient matrix associated with project k
x_k^i = the ith proposal associated with project k
s_{ik} = the weight associated with x_k^i
dev = the "deviational" variable associated with the mth coupling constraint as needed
b_m = the RHS associated with the mth coupling constraint
M = the cost per unit of dev (a very large number)

See Appendix C for a complete listing of the initial Master Problem using CPLEX Version 3.0.

Subproblems. The structure of the subproblems matches the classical form for subproblems originally presented by Dantzig and Wolfe (7). The project based subproblems would each contain the following constraints obtained from the original problem and the adjusted objective function presented earlier in this section:

$$\text{Minimize} \quad z = (c_k - \pi a_k)x_k$$

subject to

$$\sum_i \sum_j K_{ij} Y_{ij} \leq B_k - \sum_i \sum_j D_{ij} \qquad \forall k \text{ and } (i,j) \in I_k$$

$$\sum_i \sum_j S_{ijl} \leq PH_{kl} \qquad \forall k, l \text{ and } (i,j) \in I_k$$

$$-X_i + X_j + Y_{ij} \geq T_{ij} \qquad \text{for } (i,j) \in I_k$$

$$0 \leq Y_{ij} \leq M_{ij} \qquad \text{for } (i,j) \in I_k$$

where

c_k = the objective function coefficients associated with project k

a_k = the coupling constraint coefficient matrix associated with project k

x_k = the variables associated with project k

π = the shadow prices associated with the common constraints received from the Master Problem

During the first iteration of the Dantzig-Wolfe decomposition technique, the subproblems will be solved using the zero vector as the shadow prices passed from the Master Problem. Thus, the Subproblems objective functions in the first iteration are simply the normal costs

for each project. These proposed candidate solutions will then be forwarded to the Master Problem to become columns for the convex combination. By solving the combination, a new set of "prices" is established which represents the updated net value of the resources vis a vis their costs. See Appendix D for a complete listing of the initial subproblems provided by CPLEX version 3.0.

Proposal Criteria. At each iteration, a subproblem produces a new solution based on the updated prices. However, the newly generated solution may not provide any additional benefit to the Master Problem. To determine whether a newly generated solution should be introduced to Master Problem, the following test is used:

If the objective function value, z_k, associated with the optimal solution generated by the subproblem k is less than the cost of all the previous solutions accepted by the Master Problem, then it enters the Master Problem. The measure for the cost of all the previous solutions comes from the dual (shadow) prices associated with the convexity constraints of the Master Problem. Let π_k be the shadow price associated with the convexity constraint of subproblem k. Then the test can be written as follows:

If $(z_k - \pi_k) < 0$, then proposal x_k enters the Master Problem.

Stopping Criteria. The stopping criteria that have been proposed include the following:

1. If every subproblem fails the test listed above, then the optimal solution to the Master Problem and the original problem has been found (33:154).
2. If the difference between iterative solutions of the master problem is less than a given amount then the procedures ends. (i.e. $z_{i+1}-z_i<\varepsilon$, then stop.)

Dual Interpretation. The stopping criteria listed above represent the use of Strong and Weak Duality, respectively. However, as the Dantzig-Wolfe Decomposition technique is employed, the dual variables associated with each master problem, as Baumol and Fabian suggest, have an economic interpretation as well (3).

At optimality, the dual variables associated with the Master Problem represent the normal dual (shadow) prices and (dual slack) opportunity costs (3:6). However, before optimality is achieved, the dual variables can be interpreted slightly differently.

At each iteration of the Master Problem, the dual variables associated with the current solution represent:

1) Dual (shadow) prices or the marginal profitability of introducing more of a given resource into the problem without changing the current basic solution to the problem.
2) Opportunity costs (dual slack) or the profitability of introducing another proposal to the current solution of the problem.
3) Dual (shadow) prices associated with the convexity constraints of each subproblem of the worth of the current proposals from each subproblem.

The dual prices of the program constraints represent the cost per unit of program resources that a project must pay for submitting a proposal that utilizes the program resources. The dual prices of the

convexity constraints represent the current value of the project's proposals to the total program. At each iteration, a project determines its best proposal based on the current "prices" of program resources and compares its value with the dual price of its associated convexity constraint. This difference of these values represent the opportunity cost (or gain) to the program if that proposal were included as part of its optimal schedule.

The dual variable values provide insight into what variables and resources are important within the model.

Conclusion

In this chapter, the basic multi-project model was presented and then converted into a Dantzig-Wolfe Master Problem and several Subproblems. These models serve as the cornerstone in proposing feasible multi-project scheduling solutions and in conducting sensitivity analysis in the next chapter.

IV. Computational Results

Introduction

In Chapter III, the structure of the original Base Model was established. From this structure, the Dantzig-Wolfe Master Problem and Subproblems were formed. In this chapter, a simple illustrative example using the Base Model is presented, decomposed, and the results obtained from solving the Dantzig-Wolfe Master Problem and Subproblems are reviewed. The review of the Dantzig-Wolfe Master includes the program schedule, the program cost, and the required program resources determined at each iteration of the Dantzig-Wolfe Decomposition algorithm. Reviewing this information will provide the decision maker with insight on how the program schedule and cost are affected by the availability of resources.

In addition to the review of the Dantzig-Wolfe Decomposition results, the optimal solution to the model obtained from the decomposition procedure is compared to the optimal solution obtained by a linear solver. Once the equivalency of the two solutions is shown, the optimal program schedule is then used to perform parametric analyses individually on the program budget and personnel months constraints, respectively.

Example

The network given in Figure 5 represents the illustrative program planning problem being evaluated. The network represents a three-project program with event nodes represented by the circles. The event nodes corresponding to a particular project are labeled with the project identifier (A, B, or C) listed first. The arrows represent a precedence relationship with the arrowhead pointing to node j from node i. The dotted arrow represents an interproject relationship between Project A and Project B; that is, Activity (B4,B14) of Project B can not start until Activity (A1,A3) of Project A is completed. For example, Project

A might be an Analytical Study with activity (A1,A3) a cost effectiveness analysis. Project B could be the Design and Layout with activity (B4,B14) a product processing sketches. The milestone, event A3, of completing the cost effectiveness analysis and choosing a design based on the cost effectiveness analysis must take place before the product processing sketches activity begins.

The program end node, X_n, represents the latest finish time between the three projects. The program's time of completion date, *TCD*, is 84 months, with a planning horizon, or drop-dead completion date, *T*, of 90 months. Projects A, B, and C have budgets of $11,800,000, $16,350,000, and $13,700,000, respectively. The program has an overall budget of $41,850,000.

Appendix A describes the data generation technique used to establish the appropriate activity, program, and project information for the illustrative Base Model. Table 1 provides the project and program variables information results while Table 2 provides the appropriate activity information results. Table A1, located in Appendix A, provides the additional program and project parameters. See Appendix B for the Linear Program (LP) form of the Base Model from CPLEX Version 3.0.

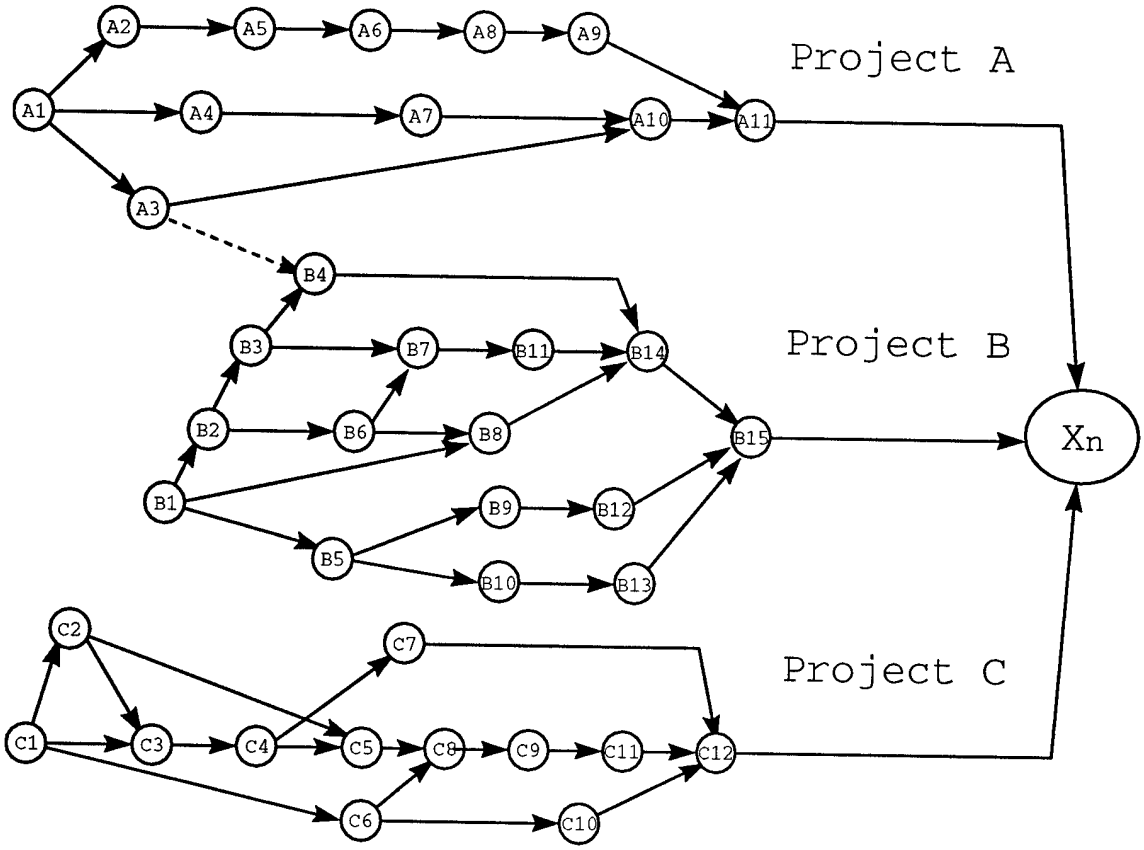

FIGURE 5. Program Activity Network

TABLE 1

PROJECT AND PROGRAM INFORMATION

Project	Budget (B_k) (in tens of thousands of dollars)	Over-head cost (OH) (in tens of thousands of dollars)	Upper Bound on additional engineer months (PH_{k1}, PH_1)	Upper Bound on additional manager months (PH_{k2}, PH_2)
A	1180		325	75
B	1635		125	25
C	1370		200	50
Overall	4185	0.305	650	150

TABLE 2

ACTIVITY INFORMATION

Activity	Normal Duration ($T_{i,j}$)	Normal Cost ($D_{i,j}$)	Upper Bound crashing ($M_{i,j}$)	Cost of crashing ($K_{i,j}$)
A1,A2	3.0	71.8	1.7	15.0
A1,A4	4.4	15.1	2.5	2.0
A1,A3	20.2	73.3	5.7	17.4
A2,A5	2.8	63.1	1.1	11.1
A4,A7	11.1	19.5	3.5	1.8
A3,A10	26.0	17.0	5.0	1.5
A5,A6	32.8	99.4	7.8	23.7
A6,A8	27.8	73.8	4.2	16.9
A7,A10	8.1	71.0	1.8	12.6
A8,A9	13.8	42.2	-	-
A9,A11	5.9	83.6	2.2	15.8
A10,A11	1.5	45.6	1.0	10.3
B1,B2	27.1	52.2	9.7	1.7
B1,B8	35.4	60.1	22.8	0.7
B1,B5	2.5	42.5	1.7	8.1
B2,B3	25.3	4.6	17.7	0.3
B2,B6	9.0	33.7	8.9	0.6
B3,B4	9.1	82.4	8.7	7.3
B3,B7	25.5	21.7	3.6	5.1
B4,B14	32.7	40.5	8.3	7.5
B5,B9	6.2	49.4	3.4	11.1
B5,B10	24.8	68.5	8.5	3.4
B6,B7	3.3	20.8	0.8	3.2
B6,B8	15.6	85.6	3.3	9.8
B7,B11	5.3	27.7	4.8	2.3
B8,B14	27.4	35.2	16.6	4.5
B9,B12	14.0	95.0	6.0	2.4
B10,B13	18.1	17.0	6.8	1.3
B11,B14	32.7	62.7	18.1	6.3
B12,B15	21.0	69.5	15.0	1.3
B13,B15	15.4	37.6	14.5	0.4
B14,B15	3.4	25.0	2.4	0.6
C1,C2	20.3	75.7	1.9	5.2
C1,C3	32.5	94.4	14.6	11.3
C1,C6	24.2	36.9	11.1	4.6
C2,C3	2.7	53.8	2.6	1.5
C2,C5	19.2	95.8	10.2	23.1
C3,C4	25.8	34.5	4.0	1.1
C4,C5	16.2	39.0	13.9	1.8
C4,C7	18.9	87.0	10.5	6.0
C5,C8	9.6	8.1	9.5	1.5
C6,C8	21.2	64.3	4.2	7.5
C6,C10	28.7	61.6	11.7	5.9
C7,C12	2.4	23.4	1.5	3.2
C8,C9	9.5	8.1	3.6	1.0
C9,C11	6.8	9.2	3.5	1.3
C10,C12	2.4	73.3	1.7	12.1
C11,C12	28.8	17.2	15.2	1.5

Implementation

In order to produce files readable by most commercially available solvers, a short program written in Visual Basic for Microsoft Excel Version 5.0. This program allowed the creation of Mathematical Programming Standard (MPS) format files from Microsoft Excel Spreadsheets. The program code and a sample spreadsheet can be found in Appendix E.

The created MPS format files (including the Base Model, the Dantzig-Wolfe Master Problem and Subproblems) were then read into CPLEX Version 3.0 and solved via CPLEX's linear optimization solver on a Sun Sparcstation. CPLEX was chosen as a platform for the simple illustrative example because of its unlimited variable capacity for solving real-world problems of much greater size and complexity (42:52).

For the Dantzig-Wolfe Master Problem and Subproblems the dual (shadow) prices were transferred to a MATHCAD 6.0 Plus worksheet where the new objective function coefficients for the Subproblems were calculated (See Appendix F for an example of the equations used). These new coefficients were transferred back to CPLEX for resolving the Subproblems. The Subproblem solutions were transferred back into MATHCAD and used to generate new proposals for the Master problem. The new proposals were entered into the Master problem and the cycle begun again with the generation of a new updated Master MPS format file.

Results

Dantzig-Wolfe Initial Schedule. The results of the Dantzig-Wolfe Master Problem in the first iteration are presented in Table 3. The objective function value has been rounded to the nearest dollar for display purposes only. Calculations for each iteration reflect use of the actual values.

TABLE 3

DANTZIG-WOLFE MASTER PROBLEM INITIAL RESULTS (ITERATION 1)

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$52,060,613,286
XN	129.2
$x_{A_1}^1$	1
$x_{B_1}^1$	1
x_C^1	1

Remember that the x_t^j represent the jth proposal (the first in this case) from project t.

The values obtained from the first iteration by the Master Problem correspond to each project minimizing its additional resource costs. Thus, the Subproblem proposals passed to the Master Problem represent each project's normal activity schedule (no activity "crashing" occurs). See Table 4 for the project's normal activity schedule.

The critical path for this first iteration comes from Project C and is C1-C3-C4-C5-C8-C9-C11-C12.

Under normal conditions, the program length would be 129.2 months, the completion time of the last project. The cost of this solution, the cost of the overhead plus the normal activity costs, is $47,047,286. This solution provides the most flexibility in coping with problems to the decision maker. If any activities experience delays, the succeeding activities can still be crashed to make up the time lost; however, this solution is unfortunately *over* the allowed budget of $41,850,000 *and* the drop-dead date of 90 months. While mathematically feasible to the "relaxed" model, it is not operationally feasible for the due date or budget requirements.

TABLE 4

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM INITIAL RESULTS

(ITERATION 1)

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0		
XA2	3		
XA3	20.2		
XA4	4.4		
XA5	5.8		
XA6	38.6		
XA7	15.5		
XA8	66.4		
XA9	80.2		
XA10	46.2		
XA11	86.1		
XB1	0		
XB2	27.1		
XB3	52.4		
XB4	61.5		
XB5	2.5		
XB6	36.1		
XB7	77.9		
XB8	51.7		
XB9	8.7		
XB10	27.3		
XB11	83.2		
XB12	22.7		
XB13	45.4		
XB14	115.9		
XB15	119.3		
XC1	0		
XC2	20.3		
XC3	32.5		
XC4	58.3		
XC5	74.5		
XC6	24.2		
XC7	77.2		
XC8	84.1		
XC9	93.6		
XC10	52.9		
XC11	100.4		
XC12	129.2		

Dantzig-Wolfe Interim Schedules. The interim results of the Dantzig-Wolfe Master Problem are plotted in Figure 6. Detail schedules can be found in Appendix G.

The interim results represent additional program schedules that could be implemented by the decision maker. At each iteration, the overall cost of the program is decreases while the program duration decreases or stays the same. However, as activities are "crashed" the decision maker's schedule and resource flexibility decreases.

Although these solutions are not optimal for the full requirements set, they provide additional time/cost trade-off information to the decision maker.

Dantzig-Wolfe Final Schedule. The results of the Dantzig-Wolfe Master Problem in the final iteration are presented in Table 5. The objective function value has been rounded to the nearest dollar and the program duration to the nearest tenth for display purposes only. Calculations for each iteration reflect use of the actual values.

TABLE 5

DANTZIG-WOLFE MASTER PROBLEM FINAL RESULTS

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$56,380,286
XN_5	79.5
X_{A_7}	.372638
X_{A_5}	.627362
X_{B_3}	1.000000
X_{C_4}	.997488
X_{C_5}	.002512

The values obtained from the final iteration by the Master Problem correspond to the program minimizing its overhead and additional resource costs. Thus, the Subproblem proposals chosen by the Master Problem represent the best combination of solutions for the whole program. See Table 6 for the project's final activity schedule.

TABLE 6

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM FINAL RESULTS

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0	(A1,A2)	1.7
XA2	1.3	(A2,A5)	1.1
XA3	20.2		
XA4	4.4		
XA5	3		
XA6	35.8	(A6,A8)	1.6
XA7	15.5		
XA8	62.0		
XA9	75.8	(A9,A11)	2.2
XA10	46.2		
XA11	79.5		
XB1	0	(B1,B2)	9.7
XB2	17.4	(B2,B3)	17.7
XB3	25	(B3,B7)	3.6
XB4	45.8		
XB5	21.2		
XB6	26.4		
XB7	46.9	(B7,B11)	4.8
XB8	42		
XB9	44.5		
XB10	46.0		
XB11	47.4	(B11,B14)	1.6
XB12	58.5		
XB13	64.1		
XB14	78.5	(B14,B15)	2.4
XB15	79.5		
XC1	0		
XC2	29.8		
XC3	32.5	(C3,C4)	4
XC4	54.3	(C4,C5)	13.9
XC5	56.6	(C5,C8)	9.5
XC6	24.2		
XC7	77.1		
XC8	56.7	(C8,C9)	3.6
XC9	62.6	(C9,C11)	3.5
XC10	52.9		
XC11	65.9	(C11,C12)	15.2
XC12	79.5		

The critical paths associated with this final iteration are as follows:

Project A: A1-A2-A5-A6-A8-A9-A11

Project B: B1-B2-B3-B7-B11-B14-B15

Project C: C1-C3-C4-C5-C8-C9-C11-C12

At optimality, the program length would be 79.5 months. The cost of this solution (the cost of the overhead plus the normal and crashed activity costs minus the early completion bonus) is $38,274,286. This optimal solution is approximately $10,000,000 less than the first solution obtained by the Master Problem. While this solution is less expensive and finishes early, it provides the least amount of schedule and resource flexibility.

With most of the critical path activities already planned to be "crashed", if any delays occur along the critical path, the entire program will be delayed. Also, since we are "crashing" a large number of activities, the number of critical path activities that have no slack to absorb delays increases as well.

If delays do occur, since the optimal solution makes use of the additional resources available, the optimal solution may leave the program with no additional resources left to deal with problems. The benefit of a planning tool such as the one discussed here is that the key decision makers can review these conditions and determine if the risk is acceptable, if additional resources must be acquired, and/or if program deadlines should be relaxed.

The decision maker at this point may ask what is being done with the activities that are not along the critical path. The activities not along the critical path could, if allowed by the nature of the activity, be "extended." "Extending" an activity occurs by applying less than the normal resource level to the activity in order to lengthen its duration. See Appendix H for the extension of the Base Model to include activity

"extending" and the results of the Dantzig-Wolfe Decomposition Technique on the extended Base Model.

As mentioned in the discussion of the Dantzig-Wolfe Interim Schedules, the Base Model provides a range of options in the form of solutions for the decision maker to apply their own insight and experience. This range of solutions can aid the decision maker in considering these key questions. Figure 6 represents the range of solutions provided by the Dantzig-Wolfe Master Problem. Note that any schedule after iteration 2 is budgetarily feasible.

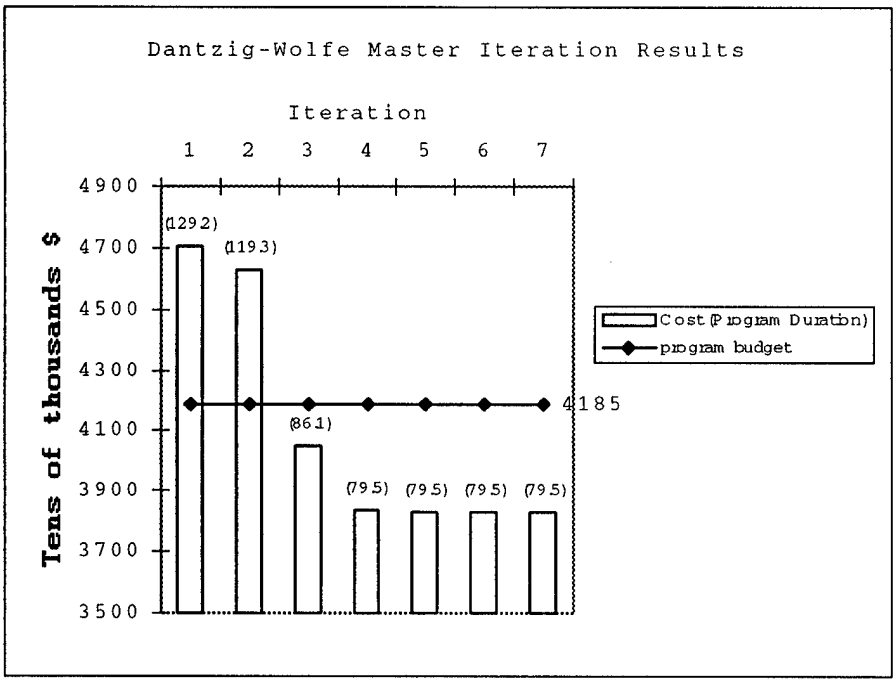

Figure 6. Dantzig-Wolfe Master Problem Iteration Results

Dantzig-Wolfe Master Problem Sensitivity Analysis. A classic sensitivity analysis of the right-hand sides for the constraints was conducted at each iteration of the Dantzig-Wolfe Master Problem. The results of these sensitivity analyses are provided in Table 7.

The ranges shown in Table 7 represent the bounds for each right-hand side (while all other right-hand sides remain fixed). As long as

the values of the right-hand side remain within these bounds, only the value of the objective function and the values of the basic (typically non-zero) variables within the Master Problem will change. If a right-hand side is changed beyond the bounds given, the value of the objective function and the actual set of basic variables and their associated values will change.

The sensitivity analysis from each iteration of the Dantzig-Wolfe Master Problem provides additional information about the feasible solution at each iteration. The sensitivity analysis shows project slack, establishes additional required resource levels, and the sets the minimum program budget needed for the current feasible schedule.

Looking at the results for the first iteration of the Dantzig-Wolfe Master Problem provides the decision maker with the following information (remember that in the first iteration, no activities were "crashed", so it represents the normal program schedule):

1. The lower/upper bounds of the Program End constraints represent how much shorter/longer a project's duration can be without affecting the optimal basis
2. The lower/upper bounds of the program precedence constraint, (A3,B4), represent the time frame that event A3 can take place in without affecting the optimal basis and the time of event B4.
3. The lower/upper bounds of the Program Budget constraint represent the amounts that additional funding dollars (above the normal program activity costs) can be changed without affecting the optimal basis.
4. The lower/upper bounds of the Personnel Months constraints represent the amounts that additional personnel months (above the normal program activity costs) can be changed without affecting the optimal basis.

TABLE 7

RIGHT HAND SIDE SENSITIVITY RANGES -

DANTZIG-WOLFE MASTER PROBLEM COUPLING CONSTRAINTS

Constraint Name	Identifier	Dual Price	Down	Current	Up
Iteration 1					
row2	Link AB	zero	-infinity	zero	41.3
row3	Program End A	zero	-infinity	zero	43.1
row4	Program End B	zero	-infinity	zero	9.9
row5	Program End C	179272	-9.9000	zero	+infinity
row6	Program Budget	-10000	-infinity	1795.6000	2315.3286
row7	Eng. Months	zero	zero	650.0000	+infinity
row8	Mgr. Months	zero	zero	150.0000	+infinity
Iteration 2					
row2	Link AB	zero	-infinity	zero	41.3
row3	Program End A	zero	-infinity	zero	33.2
row4	Program End B	156190	-23.3894	zero	9.9
row5	Program End C	23082	-9.9000	zero	44.2584
row6	Program Budget	-10000	-infinity	1795.6000	2160.7653
row7	Eng. Months	zero	34.2745	650.0000	+infinity
row8	Mgr. Months	zero	9.1399	150.0000	+infinity
Iteration 3					
row2	Link AB	zero	-infinity	zero	28.2212
row3	Program End A	64.8474	-6.5651	zero	9.1432
row4	Program End B	1.5717	-33.2000	zero	6.5651
row5	Program End C	1.5014	-43.1000	zero	6.6000
row6	Program Budget	zero	1659.8472	1795.6000	+infinity
row7	Eng. Months	zero	175.3382	650.0000	+infinity
row8	Mgr. Months	zero	46.7569	150.0000	+infinity
Iteration 4					
row2	Link AB	zero	-infinity	zero	25.6349
row3	Program End A	16.9551	-6.5651	zero	4.4936
row4	Program End B	49.1654	-0.0349	zero	6.5651
row5	Program End C	1.8000	-13.8651	zero	0.0349
row6	Program Budget	zero	1673.6740	1795.6000	+infinity
row7	Eng. Months	zero	372.5526	650.0000	+infinity
row8	Mgr. Months	zero	99.3474	150.0000	+infinity
Iteration 5					
row2	Link AB	zero	-infinity	zero	25.6349
row3	Program End A	15.5924	-6.5651	zero	2.6349
row4	Program End B	50.5281	-0.0349	zero	6.5651
row5	Program End C	1.8000	-13.8651	zero	0.0349
row6	Program Budget	zero	1664.5853	1795.6000	+infinity
row7	Eng. Months	zero	359.1336	650.0000	+infinity
row8	Mgr. Months	zero	95.7690	150.0000	+infinity
Iteration 6					
row2	Link AB	zero	-infinity	zero	25.6349
row3	Program End A	16.5219	-3.7651	zero	2.6349
row4	Program End B	49.5986	-0.0349	zero	3.7651
row5	Program End C	1.8000	-13.8651	zero	0.0349
row6	Program Budget	zero	1662.1612	1795.6000	+infinity
row7	Eng. Months	zero	355.4600	650.0000	+infinity
row8	Mgr. Months	zero	94.7893	150.0000	+infinity
Iteration 7					
row2	Link AB	zero	-infinity	zero	25.6349
row3	Program End A	16.9000	-1.5651	zero	2.6349
row4	Program End B	49.2205	-0.0349	zero	1.5651
row5	Program End C	1.8000	-13.8651	zero	0.0349
row6	Program Budget	zero	1661.2080	1795.6000	+infinity
row7	Eng. Months	zero	353.9655	650.0000	+infinity
row8	Mgr. Months	zero	94.3908	150.0000	+infinity

First Iteration. In the first iteration, the project durations of Project A and Project B can be shortened as much as desired (lower bounds are -infinity) but will not change the optimal basis since these projects are not critical to the end time of the program. The upper bounds of the Program End constraints for Project A and Project B represent the amount of time the projects can be delayed (slack) compared with the current program duration.

Since Project C is the critical path for the program in the first iteration of the Dantzig-Wolfe Master Problem, the bounds have a different interpretation. The lower bound of the Program End constraint for Project C represents the amount of time Project C can be shortened before a different project becomes part of the critical path. The upper bound states that Project C can be lengthened by any amount and it will still be on the critical path.

The Program Precedence constraint bounds simply represent time window that event A3 can occur in without affecting the current optimal solution. The lower bound states that event A3 can occur as soon as possible. The upper bound represents the amount of time event A3 can be delayed (slack) without affecting event B4 or the program.

The Program Budget lower bound in the first iteration states that no matter how little money is budgeted, the current optimal basis will not change. The upper bound, however, represents, in this problem, the amount the additional funding level needs to be set at to make the current solution feasible. Remember that the "deviational" variable is relaxing the Program Budget constraint so that the Dantzig-Wolfe Master Problem believes a feasible basic solution exists.

The Personnel Months constraint bounds in the first iteration simply state that additional personnel months play no part in the current optimal solution.

Interim and Final Iterations. The lower and upper bounds on the Program End constraints for the interim Dantzig-Wolfe Master Problem

solutions provide, for critical path projects, the time frame that the projects can be shortened or lengthened. The lower bound represents the point at which a shorter project duration that would require a change in the project set of activities "crashed" to attain. The upper bound represents a point at which a longer project duration relaxes the need for as much activity "crashing", and changes the activity "crashed" set within the project.

For non-critical path projects, the bound representation does not change from that described by the first iteration. For this problem, the same can be stated for the Program Precedence constraint linking Project A to Project B.

Until a feasible program solution is attained, the Program Budget constraint bounds description remains the same as that described in the first iteration. However, once a feasible program schedule has been established and the "deviational" variable is no longer needed to relax the Program Budget constraint, the interpretation of the bounds changes.

The lower bound of the Program budget constraint represents the minimum amount of additional funds necessary to implement the current program schedule. While this bound provides the least amount of additional funding required to complete the program based on the current program schedule, it also represents the least flexible funding profile for the program. Any delays or problems that occur will have no reserve funds to utilize, other than any cushion built into the target due date itself.

The upper bound of the Program Budget constraint, once the solution is budgetarily feasible, now represents the amount of additional funds that could be applied before a different optimal solution could be attained. In this simple illustrative example, once a feasible program schedule was found, the Program Budget constraint was no longer critical (upper bound +infinity).

The bounds on the Personnel Months constraints are similar to the Program Budget constraint bounds in representation. The lower bound represents the minimal amount of additional personnel months for engineers and managers required to implement the current feasible program schedule without affecting the basis. The upper bounds represent the how many additional personnel months can be added before the current optimal basis changes (+infinity for this simple illustrative example). See Figure 7 for a listing of the program minimum personnel hours required at each iteration of the Dantzig-Wolfe Master Problem.

Analyzing the Base Model. One of the reasons for using the Dantzig-Wolfe model is to make the solution time of very large complex problems tractable. Once an optimal solution to the Dantzig-Wolfe Master Problem is attained and translated back into the Base Model variables, the basis associated with this optimal solution can be used to "jump start" the linear solver.

For example, CPLEX is limited only by available memory in the number of variables allowed within a problem; however, the number of variables in a problem is only one factor that causes solution times to problems to be intractable (others factors including complexity and the number of constraints). By providing CPLEX with an optimal basis start, the solution time of the program is minimized.

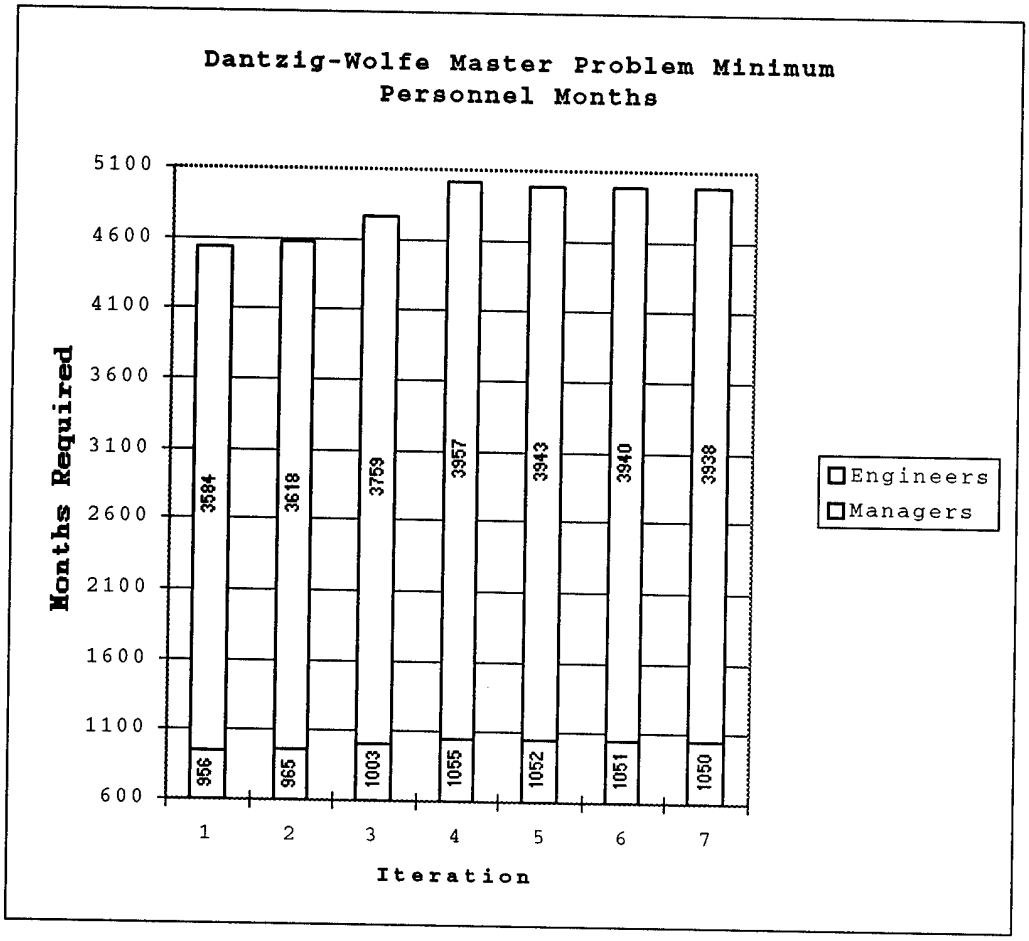

Figure 7. Dantzig-Wolfe Master Problem Minimum Personnel Months

Base Model Sensitivity Analysis. A classic sensitivity analysis of the right-hand sides for the constraints was conducted on the Base Model. The results of this sensitivity analysis are provided in Table 8.

TABLE 8

RIGHT HAND SIDE SENSITIVITY RANGES -

BASE MODEL CONSTRAINTS

Constraint Name	Identifier	Dual Price	Down	Current	Up
row66	Link AB	zero	-infinity	zero	25.6349
row55	Program End A	16.9000	-1.5651	zero	2.6349
row56	Program End B	49.2205	-0.0349	zero	1.5651
row57	Program End C	1.8000	-13.8651	zero	0.0349
row54	Program Budget	zero	1661.2825	1795.6000	+infinity
row64	Eng. Months	zero	353.9655	650.0000	+infinity
row65	Mgr. Months	zero	94.3908	150.0000	+infinity

Comparing the results of Table 7, Iteration 7 and Table 8 reveals that the sensitivity analysis conducted on the final Dantzig-Wolfe Master Problem is the same as that of the optimal solution in the Base Model. The only difference between the two tables occurs in the lower bound for the program budget, with a difference of $745, which is typical of standard computer round-off error.

If the decision maker is only concerned with the sensitivity of the optimal solution, then the sensitivity analysis obtained from the Dantzig-Wolfe final Master Problem should, in this aggregate level, be sufficient and there is no need to solve the Base Model. However, if the decision maker desires more information on the sensitivity of the Base Model to specific changes then the Base Model should be solved and a parametric analysis of the optimal solution conducted for the specific information desired.

Parametric Analysis of the Base Model. Linear programming sensitivity analysis provides the bounds for the right-hand sides at which the optimal basis remains optimal if only a single change is made. A parametric analysis of the right-hand side values looks at varying right-hand sides over defined sets of change. The defined sets of change can be for a single right-hand side or for a combination of right-hand sides. For the Base Model, a single right-hand side will be changed at a time using the linear programming solver HyperLindo. HyperLindo was chosen because CPLEX does not provide a parametric analysis function. In varying a single right-hand side over the specified range, the values of that right-hand side that cause a change in the optimal basis are identified. For the Base Model, we are concerned with the effects of varying the levels of resources available to the program. Figure 8 shows a parametric analysis of the program budget over the range of [0, 4185].

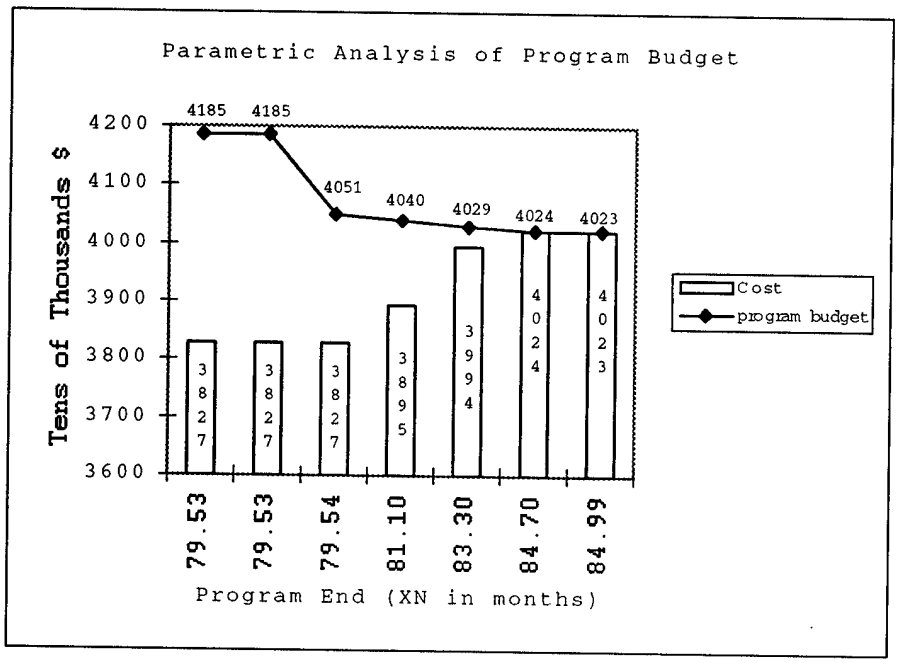

Figure 8. Parametric Analysis of Program Budget

Figure 8 displays how, as the program budget is decreased, the overall program cost and duration increase. The increase in overall cost, even though the budget is being reduced, results from fewer resources ($) being available for "crashing" activities, causing the program to take longer. The longer program duration creates more overhead and keeps the program from realizing its bonus for finishing ahead of schedule. This type of overall budgeting profile will be invaluable to the decision maker in determining appropriate trade-offs between funding levels and program duration.

Figure 9 provides a look at the parametric analysis results of individually varying the amount of additional Engineering and Managerial Months available to the program.

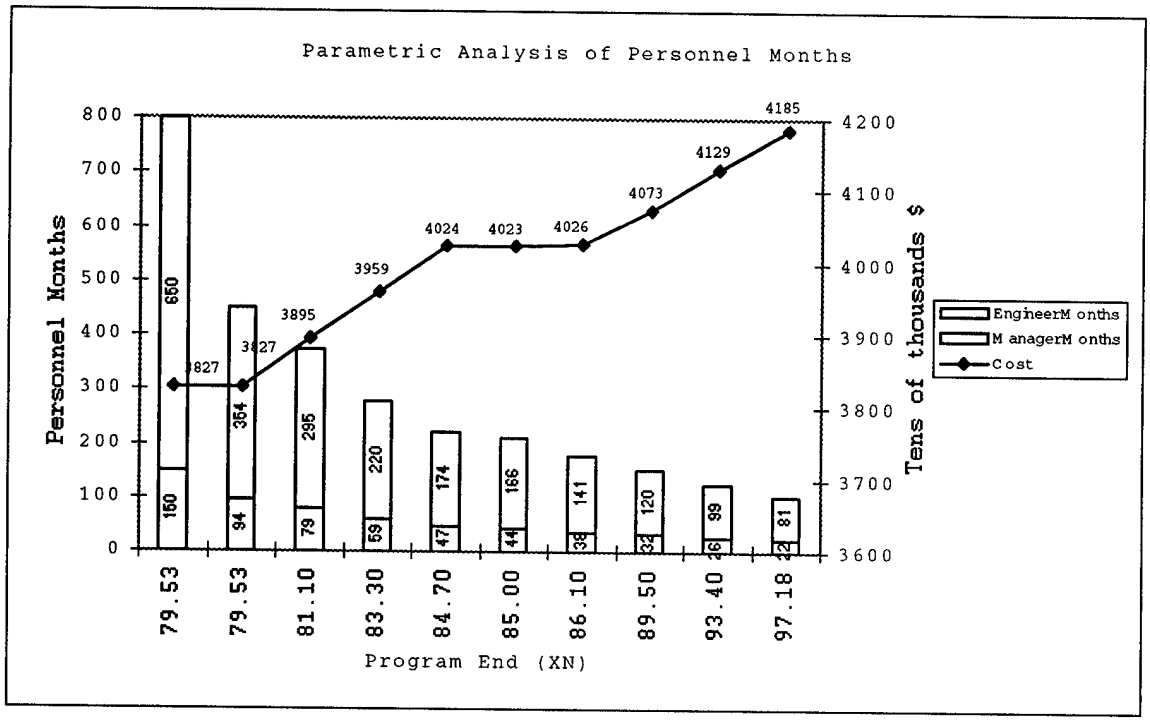

Figure 9. Parametric Analysis of Personnel Months

Figure 9 displays how, as the number of available engineering and managerial months decreases, the overall program cost and duration increase. The increase in program cost, once again, occurs due to lack of available resources to "crash" activities.

For the Base Model, the parametric analyses provide additional schedule possibilities. These possibilities provide additional information to the decision maker on how sensitive the program is to changes in resources. Other changes, such as the estimated normal duration of a critical activity (i.e., one that shows up as critical a majority of the time in a number of simulation runs), the project budgets, and the project personnel months, are candidates for parametric analysis.

Parametric analyses provide additional information for the decision maker to use in planning a new program.

Conclusion

This chapter provided the results from implementing the Dantzig-Wolfe Decomposition technique on the Base Model. The results included a range of scheduling and planning possibilities and a comparison of sensitivity analyses from the final Dantzig-Wolfe Master Problem and the Base Model. This chapter also demonstrated the usefulness of conducting parametric analyses for producing additional scheduling and planning information.

V. Conclusions and Future Research

Summary

The specific problem addressed in this study is to provide decision makers a decision support tool that generates feasible multi-project program schedules, determines appropriate funding levels for each project, and conducts sensitivity analysis upon these schedules and funding levels.

This study has demonstrated a method utilizing the Work Breakdown Structure and Dantzig-Wolfe Decomposition for generating feasible aggregate level multi-project program schedules. To implement the Dantzig-Wolfe Decomposition technique, this study presented the use of "deviational" variables to relax some of the constraints in the Dantzig-Wolfe Decomposition Master Problem until a starting basic feasible solution could be attained.

Upon generating the optimal solution in the Dantzig-Wolfe Decomposition procedure, a traditional sensitivity analysis was conducted on this solution. The traditional sensitivity analysis provided information on how variable the optimal assignment of resources would be to changes in a single factor. To validate the results of the sensitivity analysis conducted on the Dantzig-Wolfe optimal solution, a sensitivity analysis was conducted on the Base Model optimal solution and the two sensitivity analyses compared.

If the information from the Dantzig-Wolfe Decomposition technique (schedules, funding levels, and sensitivity analysis) pointed to the need for more information about certain parameters within the model, a method for using an optimal solution generated by the Dantzig-Wolfe Decomposition technique to "jump-start" a linear solver was presented.

Once an optimal solution to the linear model was obtained, an individual parametric analysis was conducted on the program budget, engineering hours, and managerial hours constraints of the Base Model.

The parametric analyses conducted provided additional information to the decision maker about the interaction between program cost and duration as the individual resource levels varied.

Conclusions

Implementing the Dantzig-Wolfe Decomposition procedure by utilizing the Work Breakdown Structure appears to be an intuitive approach to determining initial schedules for multi-project programs. The Dantzig-Wolfe procedure provides a means of generating interim solutions and their appropriate funding profiles. The decision maker could then choose any one of these solutions besides the optimal solution based upon his/her own experience and risk tolerance.

Future Research

Some natural extensions to this research would involve stochastic activity durations, nonlinear activity "crashing" functions, discrete resource modeling, and implementation of Benders' Partitioning in the subproblem solution phase.

Stochastic Activity Durations. One of the simplifying assumptions made in initially demonstrating this concept was fixed activity durations. However, in any planning stage, the actual activity times may not be accurately known. Therefore, an activity duration should be modeled by some known distribution (beta, normal, exponential, and so on) or by an established activity duration estimation technique such as PERT. Stochastic programming approaches may be considered.

The inclusion of stochastic activity durations may require the use of simulation modeling to determine the criticality of the various program and project paths. Determining the criticality of a particular path would identify to the decision maker the activities that are most likely to affect the program duration.

The simulation modeling would most likely occur at the project level, with each project being simulated to determine its individual project duration. The criticality of each individual project's

activities could then be used as part of a heuristic for determining the order with which to "crash" activities. An example of just such a heuristic might be the criticality divided by the cost of crashing.

Nonlinear Activity "Crashing". Another simplifying assumption that was made in establishing this model was to assume all activity "crashing" functions to be linear. However, a nonlinear activity "crashing" function would provide a realistic account of the theory of diminishing returns. This economic theory states that as more and more resources are applied to an activity, the per unit return decreases. For example, on an aircraft assembly line, the addition of personnel for assembling the aircraft may reduce the amount of time to produce the aircraft, but as more and more personnel are added, congestion of personnel will decrease the benefit of the additional personnel.

Baumol and Fabian demonstrated that the inclusion of convex nonlinear constraints in the subproblems do not affect the performance of the Dantzig-Wolfe Decomposition technique. Since the nonlinear constraints are not contained in the objective function or the corporate constraints, they only apply to the subproblems (3:18-20). The now nonlinear subproblems could be solved using commercially available nonlinear solvers.

Discrete Resource Modeling. The model presented in this study represented resources (dollars and personnel hours) as continuous functions of cost. In the aggregate model, it may be possible to represent every resource as a function of cost. However, once the aggregate model has provided general bounds on the required resources, it may be necessary to use discrete variables in a disaggregate model. For example, in the aggregate model it may have been enough to say that we need $x for aircraft assembly personnel. In establishing the actual project schedule, the exact number of aircraft assemblers required needs to be determined.

If the project is small enough, the problem may be solved using an integer programming solver. However, for large projects, the number of integer variables may make the solution time intractable. Several techniques for dealing with large integer programs have been developed. These techniques include Sweeney-Murphy Decomposition for Integer Programs (45), Deckro and Hebert's resource bundle method (9), and Benders' Decomposition (33).

An initial look at utilizing Benders' Partitioning to generate a series of proposals from the Dantzig-Wolfe Decomposition Subproblems at each iteration was made. While the results were not favorable in this application (the continuous linear variable restriction presented no advantage to the partitioning over standard linear programming), the use of Benders' Partitioning appears to have promise for models that include other types of variables.

For example, if the original problem contained integer variables and, subsequently, the subproblems became integer programming problems, Benders' Decomposition could be utilized to generate more than one feasible solution per iteration from each subproblem as candidate solutions in the master problem. Since we would now be dealing with integer variables, we would use Sweeney-Murphy Decomposition for Integer Programs to solve for the optimal integer solution to the program as a whole.

The use of Benders' Decomposition in the subproblems is not necessary for Sweeney-Murphy Decomposition; however, the fact that Benders' Decomposition would provide a number of feasible solutions to select from at each iteration, the overall convergence rate of the Sweeney-Murphy Decomposition technique may be improved to warrant the use of Benders' Decomposition at the Subproblem level.

APPENDIX A: BASE MODEL DATA GENERATION

Data Generation. The data generation method used for determining the data points in Tables 1 and 2 follows. The following additional variables are introduced as part of the data generation effort:

P = the percentage of K_{ij} assigned to personnel costs
P_l = the percentage of P for type l personnel
W = average wage rate for personnel
w_l = scaling factor for personnel classification type l rates

Activity Duration. The activity duration amounts for normal duration (in months), T_{ij}, were generated from a Uniform [0,36] distribution. The elements were truncated to 1 decimal place with rejection of any actual 0 element generated.

Activity Crashing Bounds. The activity crashing upper bounds, M_{ij}, were generated by the formula, $T_{ij} * P_t$, where $P_t \sim U[0,1]$ with the same truncation protocol as used in the activity duration times generation.

Activity Costs. The costs (in tens of thousands of dollars) associated with each activity at normal duration, D_{ij}, were generated from a Uniform [0,100] distribution. Given the data in Table 1, at normal duration, the total program cost of activities is $2389.4, for project A is $675.4, for project B is $931.7, and for project C is $782.3.

Crashing Costs. The costs associated with crashing an activity , K_{ij}, were generated using the formula, D_{ij} * P_c, where P_c ~ U[0, 0.25].

Program and Project Budgets. The program and project budgets were generated by taking 175% of the normal duration costs and rounding up to the nearest 5.

Personnel Costs and Months. The personnel costs, P, associated with each activity are taken to be 60% of the normal or "crash" cost of the activity. Of this set percentage, the amount that goes to the various classifications of personnel will be split out (P_l). The personnel cost split for this program will be 75/25 for engineers and managers, P_1 and P_2, respectively. To determine the actual number of additional months needed by each classification of personnel, an average wage rate, W, multiplied by a scaling coefficient, w_l, is divided into the "crash" cost of an activity. For this model, the personnel months required for activity "crashing" are determined using as an average wage rate W=.3 (in tens of thousands of dollars). The scaling factor for engineers and managers is w_1=1 and w_2=1.25, respectively. These values are used to calculate the type l personnel costs associated with an activity, S_{ijl}, in the following manner:

$$S_{ijl} = \frac{P \cdot P_l \cdot K_{ij}}{w_l \cdot W}$$

Substituting this expression into the project personnel-hour constraint gives the following form:

$$\frac{P \cdot P_l \cdot \sum_i \sum_j K_{ij} Y_{ij}}{w_l \cdot W} \leq PH_{kl}, \quad \forall k, l \text{ and } (i, j) \in I_k$$

The program personnel-hour constraint will be similar to the project personnel-hour constraints and will be as follows:

$$\frac{P \cdot P_1 \cdot \sum_i \sum_j K_{ij} Y_{ij}}{w_1 W} \leq PH_1, \quad \forall 1 \text{ and } (i,j) \in I.$$

For example, let the "crash" cost for a single activity be 10 per time unit "crashed", the percentage applied to personnel costs be P=60%, the split for two classifications of personnel be P_1=75% and P_2=25%, the wage rate be W=.3, and the scaling factor be w=1. Then the left side of the constraint above for type 2 personnel would be:

$$\frac{(0.6 \cdot 0.75 \cdot 10) Y_{ij}}{1 \cdot 0.3} = 15 Y_{ij}$$

To determine the total number of personnel months required for each classification of personnel within each project and for the program, add the normal activity costs, D_{ij}, to the numerator above to create the following equation:

$$\frac{P \cdot P_1 \sum_i \sum_j \left(K_{ij} Y_{ij} + D_{ij}\right)}{w_1 \cdot W} \quad \forall 1 \text{ and } (i,j) \in I \text{ (or } I_k)$$

Overhead Costs. The overhead cost associated with the program is formulated by taking .75% of the normal activity costs per month, $OH = (\Sigma D_{ij}) * 0.0075$.

TABLE A1

ADDITIONAL PROJECT AND PROGRAM INFORMATION

Parameter	Value
W	.3
w_1	1
w_2	1.25
P	.6
P_1	.75
P_2	.25

APPENDIX B: BASE MODEL FORMULATION

```
\Problem name: all.lp

Minimize
 OBJ: 15 YA1A2 + 17.4 YA1A3 + 2 YA1A4 + 11.1 YA2A5 + 1.8 YA4A7 + 1.5 YA3A10
 + 23.7 YA5A6 + 16.9 YA6A8 + 12.6 YA7A10 + 15.8 YA9A11 + 10.3 YA10A11
 + 1.7 YB1B2 + 0.7 YB1B5 + 8.1 YB1B8 + 0.3 YB2B3 + 0.6 YB2B6 + 7.3 YB3B4
 + 5.1 YB3B7 + 7.5 YB4B14 + 11.1 YB5B9 + 3.4 YB5B10 + 3.2 YB6B7 + 9.8 YB6B8
 + 2.3 YB7B11 + 4.5 YB8B14 + 2.4 YB9B12 + 1.3 YB10B13 + 6.3 YB11B14
 + 1.3 YB12B15 + 0.4 YB13B15 + 0.6 YB14B15 + 5.2 YC1C2 + 11.3 YC1C3
 + 4.6 YC1C6 + 1.5 YC2C3 + 23.1 YC2C5 + 1.1 YC3C4 + 1.8 YC4C5 + 6 YC4C7
 + 1.5 YC5C8 + 7.5 YC6C8 + 5.9 YC6C10 + 3.2 YC7C12 + YC8C9 + 1.3 YC9C11
 + 12.1 YC10C12 + 1.5 YC11C12 + 67.9205 XN
Subject To
 row2:  YA1A2 - XA1 + XA2 >= 3
 row3:  YA1A4 - XA1 + XA4 >= 4.4
 row4:  YA1A3 - XA1 + XA3 >= 20.2
 row5:  YA2A5 - XA2 + XA5 >= 2.8
 row6:  YA4A7 - XA4 + XA7 >= 11.1
 row7:  YA3A10 - XA3 + XA10 >= 26
 row8:  YA5A6 - XA5 + XA6 >= 32.8
 row9:  YA6A8 - XA6 + XA8 >= 27.8
 row10: YA7A10 - XA7 + XA10 >= 8.1
 row11: - XA8 + XA9 + YA8A9 >= 13.8
 row12: YA9A11 - XA9 + XA11 >= 5.9
 row13: YA10A11 - XA10 + XA11 >= 1.5
 row14: - XA3 + XB4 >= 0
 row15: YB1B2 - XB1 + XB2 >= 27.1
 row16: YB1B8 - XB1 + XB8 >= 35.4
 row17: YB1B5 - XB1 + XB5 >= 2.5
 row18: YB2B3 - XB2 + XB3 >= 25.3
 row19: YB2B6 - XB2 + XB6 >= 9
 row20: YB3B4 + XB4 - XB3 >= 9.1
 row21: YB3B7 - XB3 + XB7 >= 25.5
 row22: YB4B14 - XB4 + XB14 >= 32.7
 row23: YB5B9 - XB5 + XB9 >= 6.2
 row24: YB5B10 - XB5 + XB10 >= 24.8
 row25: YB6B7 - XB6 + XB7 >= 3.3
 row26: YB6B8 + XB8 - XB6 >= 15.6
 row27: YB7B11 - XB7 + XB11 >= 5.3
 row28: YB8B14 - XB8 + XB14 >= 27.4
 row29: YB9B12 - XB9 + XB12 >= 14
 row30: YB10B13 - XB10 + XB13 >= 18.1
 row31: YB11B14 + XB14 - XB11 >= 32.7
 row32: YB12B15 - XB12 + XB15 >= 21
 row33: YB13B15 - XB13 + XB15 >= 15.4
 row34: YB14B15 - XB14 + XB15 >= 3.4
 row35: YC1C2 - XC1 + XC2 >= 20.3
 row36: YC1C3 - XC1 + XC3 >= 32.5
 row37: YC1C6 - XC1 + XC6 >= 24.2
 row38: YC2C3 - XC2 + XC3 >= 2.7
 row39: YC2C5 - XC2 + XC5 >= 19.2
 row40: YC3C4 - XC3 + XC4 >= 25.8
 row41: YC4C5 + XC5 - XC4 >= 16.2
 row42: YC4C7 - XC4 + XC7 >= 18.9
 row43: YC5C8 - XC5 + XC8 >= 9.6
 row44: YC6C8 - XC6 + XC8 >= 21.2
 row45: YC6C10 - XC6 + XC10 >= 28.7
 row46: YC7C12 - XC7 + XC12 >= 2.4
 row47: YC8C9 - XC8 + XC9 >= 9.5
 row48: YC9C11 - XC9 + XC11 >= 6.8
 row49: YC10C12 - XC10 + XC12 >= 2.4
 row50: YC11C12 + XC12 - XC11 >= 28.8
 row51: 15 YA1A2 + 17.4 YA1A3 + 2 YA1A4 + 11.1 YA2A5 + 1.8 YA4A7 + 1.5 YA3A10
        + 23.7 YA5A6 + 16.9 YA6A8 + 12.6 YA7A10 + 15.8 YA9A11 + 10.3 YA10A11
        <= 504.6
 row52: 1.7 YB1B2 + 0.7 YB1B5 + 8.1 YB1B8 + 0.3 YB2B3 + 0.6 YB2B6 + 7.3 YB3B4
        + 5.1 YB3B7 + 7.5 YB4B14 + 11.1 YB5B9 + 3.4 YB5B10 + 3.2 YB6B7
        + 9.8 YB6B8 + 2.3 YB7B11 + 4.5 YB8B14 + 2.4 YB9B12 + 1.3 YB10B13
        + 6.3 YB11B14 + 1.3 YB12B15 + 0.4 YB13B15 + 0.6 YB14B15 <= 703.3
 row53: 5.2 YC1C2 + 11.3 YC1C3 + 4.6 YC1C6 + 1.5 YC2C3 + 23.1 YC2C5 + 1.1 YC3C4
        + 1.8 YC4C5 + 6 YC4C7 + 1.5 YC5C8 + 7.5 YC6C8 + 5.9 YC6C10
        + 3.2 YC7C12 + YC8C9 + 1.3 YC9C11 + 12.1 YC10C12 + 1.5 YC11C12
        <= 587.7
 row54: 15 YA1A2 + 17.4 YA1A3 + 2 YA1A4 + 11.1 YA2A5 + 1.8 YA4A7 + 1.5 YA3A10
```

```
        + 23.7 YA5A6 + 16.9 YA6A8 + 12.6 YA7A10 + 15.8 YA9A11 + 10.3 YA10A11
        + 1.7 YB1B2 + 0.7 YB1B5 + 8.1 YB1B8 + 0.3 YB2B3 + 0.6 YB2B6
        + 7.3 YB3B4 + 5.1 YB3B7 + 7.5 YB4B14 + 11.1 YB5B9 + 3.4 YB5B10
        + 3.2 YB6B7 + 9.8 YB6B8 + 2.3 YB7B11 + 4.5 YB8B14 + 2.4 YB9B12
        + 1.3 YB10B13 + 6.3 YB11B14 + 1.3 YB12B15 + 0.4 YB13B15 + 0.6 YB14B15
        + 5.2 YC1C2 + 11.3 YC1C3 + 4.6 YC1C6 + 1.5 YC2C3 + 23.1 YC2C5
        + 1.1 YC3C4 + 1.8 YC4C5 + 6 YC4C7 + 1.5 YC5C8 + 7.5 YC6C8 + 5.9 YC6C10
        + 3.2 YC7C12 + YC8C9 + 1.3 YC9C11 + 12.1 YC10C12 + 1.5 YC11C12
        + 17.9205 XN <= 1795.6
 row55: XN - XA11 >= 0
 row56: XN - XB15 >= 0
 row57: XN - XC12 >= 0
 row58: 22.5 YA1A2 + 26.1 YA1A3 + 3 YA1A4 + 16.65 YA2A5 + 2.7 YA4A7
        + 2.25 YA3A10 + 35.55 YA5A6 + 25.35 YA6A8 + 18.9 YA7A10 + 23.7 YA9A11
        + 15.45 YA10A11 <= 325
 row59: 6 YA1A2 + 6.96 YA1A3 + 0.8 YA1A4 + 4.44 YA2A5 + 0.72 YA4A7 + 0.6 YA3A10
        + 9.48 YA5A6 + 6.76 YA6A8 + 5.04 YA7A10 + 6.32 YA9A11 + 4.12 YA10A11
        <= 75
 row60: 2.55 YB1B2 + 1.05 YB1B5 + 12.15 YB1B8 + 0.45 YB2B3 + 0.9 YB2B6
        + 10.95 YB3B4 + 7.65 YB3B7 + 11.25 YB4B14 + 16.65 YB5B9 + 5.1 YB5B10
        + 4.8 YB6B7 + 14.7 YB6B8 + 3.45 YB7B11 + 6.75 YB8B14 + 3.6 YB9B12
        + 1.95 YB10B13 + 9.45 YB11B14 + 1.95 YB12B15 + 0.6 YB13B15
        + 0.9 YB14B15 <= 125
 row61: 0.68 YB1B2 + 0.28 YB1B5 + 3.24 YB1B8 + 0.12 YB2B3 + 0.24 YB2B6
        + 2.92 YB3B4 + 2.04 YB3B7 + 3 YB4B14 + 4.44 YB5B9 + 1.36 YB5B10
        + 1.28 YB6B7 + 3.92 YB6B8 + 0.92 YB7B11 + 1.8 YB8B14 + 0.96 YB9B12
        + 0.52 YB10B13 + 2.52 YB11B14 + 0.52 YB12B15 + 0.16 YB13B15
        + 0.24 YB14B15 <= 25
 row62: 7.8 YC1C2 + 16.95 YC1C3 + 6.9 YC1C6 + 2.25 YC2C3 + 34.65 YC2C5
        + 1.65 YC3C4 + 2.7 YC4C5 + 9 YC4C7 + 2.25 YC5C8 + 11.25 YC6C8
        + 8.85 YC6C10 + 4.8 YC7C12 + 1.5 YC8C9 + 1.95 YC9C11 + 18.15 YC10C12
        + 2.25 YC11C12 <= 200
 row63: 2.08 YC1C2 + 4.52 YC1C3 + 1.84 YC1C6 + 0.6 YC2C3 + 9.24 YC2C5
        + 0.44 YC3C4 + 0.72 YC4C5 + 2.4 YC4C7 + 0.6 YC5C8 + 3 YC6C8
        + 2.36 YC6C10 + 1.28 YC7C12 + 0.4 YC8C9 + 0.52 YC9C11 + 4.84 YC10C12
        + 0.6 YC11C12 <= 50
 row64: 22.5 YA1A2 + 26.1 YA1A3 + 3 YA1A4 + 16.65 YA2A5 + 2.7 YA4A7
        + 2.25 YA3A10 + 35.55 YA5A6 + 25.35 YA6A8 + 18.9 YA7A10 + 23.7 YA9A11
        + 15.45 YA10A11 + 2.55 YB1B2 + 1.05 YB1B5 + 12.15 YB1B8 + 0.45 YB2B3
        + 0.9 YB2B6 + 10.95 YB3B4 + 7.65 YB3B7 + 11.25 YB4B14 + 16.65 YB5B9
        + 5.1 YB5B10 + 4.8 YB6B7 + 14.7 YB6B8 + 3.45 YB7B11 + 6.75 YB8B14
        + 3.6 YB9B12 + 1.95 YB10B13 + 9.45 YB11B14 + 1.95 YB12B15
        + 0.6 YB13B15 + 0.9 YB14B15 + 7.8 YC1C2 + 16.95 YC1C3 + 6.9 YC1C6
        + 2.25 YC2C3 + 34.65 YC2C5 + 1.65 YC3C4 + 2.7 YC4C5 + 9 YC4C7
        + 2.25 YC5C8 + 11.25 YC6C8 + 8.85 YC6C10 + 4.8 YC7C12 + 1.5 YC8C9
        + 1.95 YC9C11 + 18.15 YC10C12 + 2.25 YC11C12 <= 650
 row65: 6 YA1A2 + 6.96 YA1A3 + 0.8 YA1A4 + 4.44 YA2A5 + 0.72 YA4A7 + 0.6 YA3A10
        + 9.48 YA5A6 + 6.76 YA6A8 + 5.04 YA7A10 + 6.32 YA9A11 + 4.12 YA10A11
        + 0.68 YB1B2 + 0.28 YB1B5 + 3.24 YB1B8 + 0.12 YB2B3 + 0.24 YB2B6
        + 2.92 YB3B4 + 2.04 YB3B7 + 3 YB4B14 + 4.44 YB5B9 + 1.36 YB5B10
        + 1.28 YB6B7 + 3.92 YB6B8 + 0.92 YB7B11 + 1.8 YB8B14 + 0.96 YB9B12
        + 0.52 YB10B13 + 2.52 YB11B14 + 0.52 YB12B15 + 0.16 YB13B15
        + 0.24 YB14B15 + 2.08 YC1C2 + 4.52 YC1C3 + 1.84 YC1C6 + 0.6 YC2C3
        + 9.24 YC2C5 + 0.44 YC3C4 + 0.72 YC4C5 + 2.4 YC4C7 + 0.6 YC5C8
        + 3 YC6C8 + 2.36 YC6C10 + 1.28 YC7C12 + 0.4 YC8C9 + 0.52 YC9C11
        + 4.84 YC10C12 + 0.6 YC11C12 <= 150
 row66: XB4 - XA3 >= 0
Bounds
 0 <= YA1A2 <= 1.7
 0 <= YA1A3 <= 5.7
 0 <= YA1A4 <= 2.5
 0 <= YA2A5 <= 1.1
 0 <= YA4A7 <= 3.5
 0 <= YA3A10 <= 5
 0 <= YA5A6 <= 7.8
 0 <= YA6A8 <= 4.2
 0 <= YA7A10 <= 1.8
 0 <= YA9A11 <= 2.2
 0 <= YA10A11 <= 1
 0 <= YB1B2 <= 9.7
 0 <= YB1B5 <= 1.7
 0 <= YB1B8 <= 22.8
 0 <= YB2B3 <= 17.7
 0 <= YB2B6 <= 8.9
 0 <= YB3B4 <= 8.7
 0 <= YB3B7 <= 3.6
 0 <= YB4B14 <= 8.3
 0 <= YB5B9 <= 3.4
 0 <= YB5B10 <= 8.5
```

```
 0 <= YB6B7 <= 0.8
 0 <= YB6B8 <= 3.3
 0 <= YB7B11 <= 4.8
 0 <= YB8B14 <= 16.6
 0 <= YB9B12 <= 6
 0 <= YB10B13 <= 6.8
 0 <= YB11B14 <= 18.1
 0 <= YB12B15 <= 15
 0 <= YB13B15 <= 14.5
 0 <= YB14B15 <= 2.4
 0 <= YC1C2 <= 1.9
 0 <= YC1C3 <= 14.6
 0 <= YC1C6 <= 11.1
 0 <= YC2C3 <= 2.6
 0 <= YC2C5 <= 10.2
 0 <= YC3C4 <= 4
 0 <= YC4C5 <= 13.9
 0 <= YC4C7 <= 10.5
 0 <= YC5C8 <= 9.5
 0 <= YC6C8 <= 4.2
 0 <= YC6C10 <= 11.7
 0 <= YC7C12 <= 1.5
 0 <= YC8C9 <= 3.6
 0 <= YC9C11 <= 3.5
 0 <= YC10C12 <= 1.7
 0 <= YC11C12 <= 15.2
      YA8A9 = 0
End
```

APPENDIX C: INITIAL MASTER PROBLEM FORMULATION

```
\Problem name: m01.lp

Minimize
 OBJ: 67.9205 XN + 10000 dev
Subject To
 row2:  - 20.2 alpha1 + 61.5 beta1 >= 0
 row3:  XN - 86.1 alpha1 >= 0
 row4:  XN - 119.3 beta1 >= 0
 row5:  XN - 129.2 gamma1 >= 0
 row6:  17.9205 XN - dev <= 1795.6
 row7:  <= 650
 row8:  <= 150
 row9:  alpha1  = 1
 row10: beta1  = 1
 row11: gamma1  = 1
End
```

where

alpha1 represents the proposal from Project A in the first iteration, x_A^1,

beta1 represents the proposal from Project B in the first iteration, x_B^1,

gamma1 represents the proposal from Project C in the first iteration, x_C^1,

and M = 10,000.

APPENDIX D: INITIAL SUBPROBLEM FORMULATIONS

Subproblem 1: PROJECT A

```
\Problem name: a01.lp

Minimize
 OBJ: 15 YA1A2 + 17.4 YA1A3 + 2 YA1A4 + 11.1 YA2A5 + 1.8 YA4A7 + 1.5 YA3A10
 + 23.7 YA5A6 + 16.9 YA6A8 + 12.6 YA7A10 + 15.8 YA9A11 + 10.3 YA10A11
Subject To
 row2:  - XA1 + XA2 + YA1A2 >= 3
 row3:  - XA1 + XA4 + YA1A4 >= 4.4
 row4:  - XA1 + XA3 + YA1A3 >= 20.2
 row5:  - XA2 + XA5 + YA2A5 >= 2.8
 row6:  - XA4 + XA7 + YA4A7 >= 11.1
 row7:  - XA3 + XA10 + YA3A10 >= 26
 row8:  - XA5 + XA6 + YA5A6 >= 32.8
 row9:  - XA6 + XA8 + YA6A8 >= 27.8
 row10: - XA7 + XA10 + YA7A10 >= 8.1
 row11: - XA8 + XA9 + YA8A9 >= 13.8
 row12: - XA9 + XA11 + YA9A11 >= 5.9
 row13: - XA10 + XA11 + YA10A11 >= 1.5
 row14: 15 YA1A2 + 17.4 YA1A3 + 2 YA1A4 + 11.1 YA2A5 + 1.8 YA4A7 + 1.5 YA3A10
        + 23.7 YA5A6 + 16.9 YA6A8 + 12.6 YA7A10 + 15.8 YA9A11 + 10.3 YA10A11
        <= 504.6
 row15: 22.5 YA1A2 + 26.1 YA1A3 + 3 YA1A4 + 16.65 YA2A5 + 2.7 YA4A7
        + 2.25 YA3A10 + 35.55 YA5A6 + 25.35 YA6A8 + 18.9 YA7A10 - 0 YA8A9
        + 23.7 YA9A11 + 15.45 YA10A11 <= 325
 row16: 6 YA1A2 + 6.96 YA1A3 + 0.8 YA1A4 + 4.44 YA2A5 + 0.72 YA4A7 + 0.6 YA3A10
        + 9.48 YA5A6 + 6.76 YA6A8 + 5.04 YA7A10 - 0 YA8A9 + 6.32 YA9A11
        + 4.12 YA10A11 <= 75
Bounds
 0 <= YA1A2 <= 1.7
 0 <= YA1A3 <= 5.7
 0 <= YA1A4 <= 2.5
 0 <= YA2A5 <= 1.1
 0 <= YA4A7 <= 3.5
 0 <= YA3A10 <= 5
 0 <= YA5A6 <= 7.8
 0 <= YA6A8 <= 4.2
 0 <= YA7A10 <= 1.8
      YA8A9 = 0
 0 <= YA9A11 <= 2.2
 0 <= YA10A11 <= 1
End
```

Subproblem 2: PROJECT B

```
\Problem name: b01.lp

Minimize
 OBJ: 1.7 YB1B2 + 0.7 YB1B5 + 8.1 YB1B8 + 0.3 YB2B3 + 0.6 YB2B6 + 7.3 YB3B4
 + 5.1 YB3B7 + 7.5 YB4B14 + 11.1 YB5B9 + 3.4 YB5B10 + 3.2 YB6B7 + 9.8 YB6B8
 + 2.3 YB7B11 + 4.5 YB8B14 + 2.4 YB9B12 + 1.3 YB10B13 + 6.3 YB11B14
 + 1.3 YB12B15 + 0.4 YB13B15 + 0.6 YB14B15
Subject To
 row2:  - XB1 + XB2 + YB1B2 >= 27.1
 row3:  - XB1 + XB8 + YB1B8 >= 35.4
 row4:  - XB1 + XB5 + YB1B5 >= 2.5
 row5:  - XB2 + XB3 + YB2B3 >= 25.3
 row6:  - XB2 + XB6 + YB2B6 >= 9
 row7:  - XB3 + XB4 + YB3B4 >= 9.1
 row8:  - XB3 + XB7 + YB3B7 >= 25.5
 row9:  - XB4 + XB14 + YB4B14 >= 32.7
 row10: - XB5 + XB9 + YB5B9 >= 6.2
 row11: - XB5 + XB10 + YB5B10 >= 24.8
 row12: - XB6 + XB7 + YB6B7 >= 3.3
 row13: - XB6 + XB8 + YB6B8 >= 15.6
 row14: - XB7 + XB11 + YB7B11 >= 5.3
 row15: - XB8 + XB14 + YB8B14 >= 27.4
 row16: - XB9 + XB12 + YB9B12 >= 14
 row17: - XB10 + XB13 + YB10B13 >= 18.1
 row18: - XB11 + XB14 + YB11B14 >= 32.7
 row19: - XB12 + XB15 + YB12B15 >= 21
 row20: - XB13 + XB15 + YB13B15 >= 15.4
 row21: - XB14 + XB15 + YB14B15 >= 3.4
 row22: 1.7 YB1B2 + 0.7 YB1B5 + 8.1 YB1B8 + 0.3 YB2B3 + 0.6 YB2B6 + 7.3 YB3B4
        + 5.1 YB3B7 + 7.5 YB4B14 + 11.1 YB5B9 + 3.4 YB5B10 + 3.2 YB6B7
        + 9.8 YB6B8 + 2.3 YB7B11 + 4.5 YB8B14 + 2.4 YB9B12 + 1.3 YB10B13
        + 6.3 YB11B14 + 1.3 YB12B15 + 0.4 YB13B15 + 0.6 YB14B15 <= 703.3
 row23: 2.55 YB1B2 + 1.05 YB1B5 + 12.15 YB1B8 + 0.45 YB2B3 + 0.9 YB2B6
        + 10.95 YB3B4 + 7.65 YB3B7 + 11.25 YB4B14 + 16.65 YB5B9 + 5.1 YB5B10
        + 4.8 YB6B7 + 14.7 YB6B8 + 3.45 YB7B11 + 6.75 YB8B14 + 3.6 YB9B12
        + 1.95 YB10B13 + 9.45 YB11B14 + 1.95 YB12B15 + 0.6 YB13B15
        + 0.9 YB14B15 <= 125
 row24: 0.68 YB1B2 + 0.28 YB1B5 + 3.24 YB1B8 + 0.12 YB2B3 + 0.24 YB2B6
        + 2.92 YB3B4 + 2.04 YB3B7 + 3 YB4B14 + 4.44 YB5B9 + 1.36 YB5B10
        + 1.28 YB6B7 + 3.92 YB6B8 + 0.92 YB7B11 + 1.8 YB8B14 + 0.96 YB9B12
        + 0.52 YB10B13 + 2.52 YB11B14 + 0.52 YB12B15 + 0.16 YB13B15
        + 0.24 YB14B15 <= 25
Bounds
 0 <= YB1B2 <= 9.7
 0 <= YB1B5 <= 1.7
 0 <= YB1B8 <= 22.8
 0 <= YB2B3 <= 17.7
 0 <= YB2B6 <= 8.9
 0 <= YB3B4 <= 8.7
 0 <= YB3B7 <= 3.6
 0 <= YB4B14 <= 8.3
 0 <= YB5B9 <= 3.4
 0 <= YB5B10 <= 8.5
 0 <= YB6B7 <= 0.8
 0 <= YB6B8 <= 3.3
 0 <= YB7B11 <= 4.8
 0 <= YB8B14 <= 16.6
 0 <= YB9B12 <= 6
 0 <= YB10B13 <= 6.8
 0 <= YB11B14 <= 18.1
 0 <= YB12B15 <= 15
 0 <= YB13B15 <= 14.5
 0 <= YB14B15 <= 2.4
End
```

Subproblem 3: PROJECT C

```
\Problem name: c01.lp

Minimize
 OBJ: 5.2 YC1C2 + 11.3 YC1C3 + 4.6 YC1C6 + 1.5 YC2C3 + 23.1 YC2C5 + 1.1 YC3C4
 + 1.8 YC4C5 + 6 YC4C7 + 1.5 YC5C8 + 7.5 YC6C8 + 5.9 YC6C10 + 3.2 YC7C12
 + YC8C9 + 1.3 YC9C11 + 12.1 YC10C12 + 1.5 YC11C12
Subject To
 row2:  - XC1 + XC2 + YC1C2 >= 20.3
 row3:  - XC1 + XC3 + YC1C3 >= 32.5
 row4:  - XC1 + XC6 + YC1C6 >= 24.2
 row5:  - XC2 + XC3 + YC2C3 >= 2.7
 row6:  - XC2 + XC5 + YC2C5 >= 19.2
 row7:  - XC3 + XC4 + YC3C4 >= 25.8
 row8:  - XC4 + XC5 + YC4C5 >= 16.2
 row9:  - XC4 + XC7 + YC4C7 >= 18.9
 row10: - XC5 + XC8 + YC5C8 >= 9.6
 row11: - XC6 + XC8 + YC6C8 >= 21.2
 row12: - XC6 + XC10 + YC6C10 >= 28.7
 row13: - XC7 + XC12 + YC7C12 >= 2.4
 row14: - XC8 + XC9 + YC8C9 >= 9.5
 row15: - XC9 + XC11 + YC9C11 >= 6.8
 row16: - XC10 + XC12 + YC10C12 >= 2.4
 row17: - XC11 + XC12 + YC11C12 >= 28.8
 row18: 5.2 YC1C2 + 11.3 YC1C3 + 4.6 YC1C6 + 1.5 YC2C3 + 23.1 YC2C5 + 1.1 YC3C4
        + 1.8 YC4C5 + 6 YC4C7 + 1.5 YC5C8 + 7.5 YC6C8 + 5.9 YC6C10
        + 3.2 YC7C12 + YC8C9 + 1.3 YC9C11 + 12.1 YC10C12 + 1.5 YC11C12
        <= 587.7
 row19: 7.8 YC1C2 + 16.95 YC1C3 + 6.9 YC1C6 + 2.25 YC2C3 + 34.65 YC2C5
        + 1.65 YC3C4 + 2.7 YC4C5 + 9 YC4C7 + 2.25 YC5C8 + 11.25 YC6C8
        + 8.85 YC6C10 + 4.8 YC7C12 + 1.5 YC8C9 + 1.95 YC9C11 + 18.15 YC10C12
        + 2.25 YC11C12 <= 200
 row20: 2.08 YC1C2 + 4.52 YC1C3 + 1.84 YC1C6 + 0.6 YC2C3 + 9.24 YC2C5
        + 0.44 YC3C4 + 0.72 YC4C5 + 2.4 YC4C7 + 0.6 YC5C8 + 3 YC6C8
        + 2.36 YC6C10 + 1.28 YC7C12 + 0.4 YC8C9 + 0.52 YC9C11 + 4.84 YC10C12
        + 0.6 YC11C12 <= 50
Bounds
 0 <= YC1C2 <= 1.9
 0 <= YC1C3 <= 14.6
 0 <= YC1C6 <= 11.1
 0 <= YC2C3 <= 2.6
 0 <= YC2C5 <= 10.2
 0 <= YC3C4 <= 4
 0 <= YC4C5 <= 13.9
 0 <= YC4C7 <= 10.5
 0 <= YC5C8 <= 9.5
 0 <= YC6C8 <= 4.2
 0 <= YC6C10 <= 11.7
 0 <= YC7C12 <= 1.5
 0 <= YC8C9 <= 3.6
 0 <= YC9C11 <= 3.5
 0 <= YC10C12 <= 1.7
 0 <= YC11C12 <= 15.2
End
```

APPENDIX E: MPS PROGRAM CODE

Visual Basic for MS Excel V5.0 Code

```
Option Base 1
Option Explicit

Sub MakeMPS()

    Dim filename, problem_name, signcol, MPS_data, obj_row As String
    Dim num_MPS_data, i, r, c As Integer

    c = 1
    r = 2
    filename = InputBox("Enter a valid path and filename for the file (*.mps):")
    Open filename For Output As #1

    MPS_data = InputBox("Enter the number of the data sheet to be used:")
    num_MPS_data = CInt(MPS_data)
    problem_name = InputBox("Type an identifier (name) for the problem:")
    signcol = InputBox("Enter the column of number of the signs column:")
    obj_row = InputBox("Enter the row number of the objective coefficients row:")

    Print #1, "NAME"; Tab(15); problem_name
    Print #1, "ROWS"
    Print #1, Tab(2); "N"; Tab(5); "OBJ"
    Do While (Worksheets(num_MPS_data).Cells(r, CInt(signcol)).Value) <> ""
        Print #1, Tab(2); Worksheets(num_MPS_data).Cells(r, CInt(signcol)).Value; _
            Tab(5); "row" & CStr(r)
        r = r + 1
    Loop
    Print #1, "COLUMNS"
    Do While (Worksheets(num_MPS_data).Cells(1, c).Value) <> ""
        If (Worksheets(num_MPS_data).Cells(CInt(obj_row), c).Value) <> "0" Then
            Print #1, Tab(5); Worksheets(num_MPS_data).Cells(1, c).Value; Tab(15); "OBJ";
_
                Tab(25); Worksheets(num_MPS_data).Cells(CInt(obj_row), c).Value
        End If
        For i = 2 To r - 1
            If Worksheets(num_MPS_data).Cells(i, c).Value <> "" Then
                Print #1, Tab(5); Worksheets(num_MPS_data).Cells(1, c).Value; Tab(15); _
                    "row" & CStr(i); Tab(25); Worksheets(num_MPS_data).Cells(i, c).Value
            End If
        Next i
        c = c + 1
    Loop
    r = 2
    Print #1, "RHS"
    Do While (Worksheets(num_MPS_data).Cells(r, CInt(signcol + 1)).Value) <> ""
        Print #1, Tab(5); "RHS"; Tab(15); "row" & CStr(r); Tab(25); _
            Worksheets(num_MPS_data).Cells(r, CInt(signcol + 1)).Value
        r = r + 1
    Loop
    Print #1, "BOUNDS"
    Do While (Worksheets(num_MPS_data).Cells(r, 1).Value) <> ""
        Print #1, Tab(2); "LO"; Tab(5); "BOUNDSET"; Tab(15); _
            Worksheets(num_MPS_data).Cells(r, 3).Value; Tab(25); _
            Worksheets(num_MPS_data).Cells(r, 1).Value
        Print #1, Tab(2); "UP"; Tab(5); "BOUNDSET"; Tab(15); _
            Worksheets(num_MPS_data).Cells(r, 3).Value; Tab(25); _
            Worksheets(num_MPS_data).Cells(r, 5).Value
        r = r + 1
    Loop
    Print #1, "ENDATA"
    Close #1
End Sub
```

Example Spreadsheet

XA1	XA2	XA3	XA4	XA5	XB1	XB2	XB3	XB4		
-1	1								G	3
-1			1						G	4.4
-1		1							G	20.2
	-1			1					G	2.8
			-1						G	11.1
		-1							G	26
				-1					G	32.8
		-1						1	G	0
					-1	1			G	27.1
					-1				G	35.4
					-1				G	2.5
						-1	1		G	25.3
						-1			G	9
							-1	1	G	9.1
							-1		G	25.5
								-1	G	32.7
0	<=	XA1	<=	1.7						
0	<=	XA2	<=	5.7						
0	<=	XA3	<=	2.5						
0	<=	XA4	<=	1.1						
0	<=	XA5	<=	3.5						
0	<=	XB1	<=	5						
0	<=	XB2	<=	7.8						
0	<=	XB3	<=	4.2						
0	<=	XB4	<=	1.8						
1	2	3	4	5	1	2	3	4		

APPENDIX F: MATHCAD TEMPLATE

For each project, the program constraint matrix A_t was constructed. These matrices were used in calculating the new project objective functions at each iteration. See below for a partial view of each matrix.

A =

	14	15	16	17	18	19	20	21	22	23
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	2	11.1	1.8	1.5	23.7	16.9	12.6	0	15.8	10.3
6	3	16.65	2.7	2.25	35.55	25.35	18.9	0	23.7	15.45
7	0.8	4.44	0.72	0.6	9.48	6.76	5.04	0	6.32	4.12

B =

	15	16	17	18	19	20	21	22	23	24
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	-1	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	1.7	0.7	8.1	0.3	0.6	7.3	5.1	7.5	11.1
6	0	2.55	1.05	12.15	0.45	0.9	10.95	7.65	11.25	16.65
7	0	0.68	0.28	3.24	0.12	0.24	2.92	2.04	3	4.44

C =

	12	13	14	15	16	17	18	19	20	21
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	-1	0	0	0	0	0	0	0	0	0
5	0	5.2	11.3	4.6	1.5	23.1	1.1	1.8	6	1.5
6	0	7.8	16.95	6.9	2.25	34.65	1.65	2.7	9	2.25
7	0	2.08	4.52	1.84	0.6	9.24	0.44	0.72	2.4	0.6

The shadow prices from the Master Problem were entered as a column vector at each iteration and the new project objective function

determined. See below for an example of a shadow price column and the equations for determining the objective function coefficients.

$$\pi_2 := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 172972.9205 \\ -10000 \\ 0 \\ 0 \end{bmatrix}$$

$$c_A^T - \pi_2^T \cdot A \qquad c_B^T - \pi_2^T \cdot B \qquad c_C^T - \pi_2^T \cdot C$$

Once the subproblems were solved, the extreme points generated, if beneficial to the program, were converted into columns for the Master Program. See below for examples of this conversion.

$$\left(A \cdot X_{A4}^T\right)^T = (-20.2 \; -75.04135 \; 0 \; 0 \; 187.500005 \; 281.250007 \; 75.000002) \blacksquare \quad c_A \cdot X_{A4}^T = 187.500005 \blacksquare$$

$$\left(B \cdot X_{B4}^T\right)^T = (65.5 \; 0 \; -99.2 \; 0 \; 6.75 \; 10.125 \; 2.7) \blacksquare \quad c_B \cdot X_{B4}^T = 6.75 \blacksquare$$

$$\left(C \cdot X_{C4}^T\right)^T = (0 \; 0 \; 0 \; -93.4 \; 49.6 \; 74.4 \; 19.84) \blacksquare \quad c_C \cdot X_{C4}^T = 49.6 \blacksquare$$

APPENDIX G: DANTZIG-WOLFE MASTER PROBLEM RESULTS

TABLE G1

DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 1

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$52,060,613,286
XN	129.2
x_{A_1}	1
x_{B_1}	1
x_{C_1}	1

TABLE G2

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 1

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0		
XA2	3		
XA3	20.2		
XA4	4.4		
XA5	5.8		
XA6	38.6		
XA7	15.5		
XA8	66.4		
XA9	80.2		
XA10	46.2		
XA11	86.1		
XB1	0		
XB2	27.1		
XB3	52.4		
XB4	61.5		
XB5	2.5		
XB6	36.1		
XB7	77.9		
XB8	51.7		
XB9	8.7		
XB10	27.3		
XB11	83.2		
XB12	22.7		
XB13	45.4		
XB14	115.9		
XB15	119.3		
XC1	0		
XC2	20.3		
XC3	32.5		
XC4	58.3		
XC5	74.5		
XC6	24.2		
XC7	77.2		
XC8	84.1		
XC9	93.6		
XC10	52.9		
XC11	100.4		
XC12	129.2		

TABLE G3

DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 2

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$36,597,786,445
XN	119.3
$X_{A_1}^1$	1
X_{B_1}	1
X_{C_2}	.182797
X_C	.817203

TABLE G4

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 2

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0		
XA2	3		
XA3	20.2		
XA4	4.4		
XA5	5.8		
XA6	38.6		
XA7	15.5		
XA8	66.4		
XA9	80.2		
XA10	46.2		
XA11	86.1		
XB1	0		
XB2	27.1		
XB3	52.4		
XB4	61.5		
XB5	2.5		
XB6	36.1		
XB7	77.9		
XB8	51.7		
XB9	8.7		
XB10	27.3		
XB11	83.2		
XB12	22.7		
XB13	45.4		
XB14	115.9		
XB15	119.3		
XC1	0	(C1,C3)	3.6
XC2	24.4		
XC3	28.9	(C3,C4)	3.3
XC4	51.4	(C4,C5)	11.4
XC5	56.2	(C5,C8)	7.8
XC6	24.2		
XC7	73.5		
XC8	58.1	(C8,C9)	2.9
XC9	64.6	(C9,C11)	2.9
XC10	52.9		
XC11	68.6	(C11,C12)	12.4
XC12	84.9		

TABLE G5

DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 3

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$59,648,472
XN	86.1
$x_{A_1}^1$	1
x_{B_3}	.165097
x_{B_1}	.834903
x_{C_3}	.132797
x_C	.867203

TABLE G6

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 3

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0		
XA2	3		
XA3	20.2		
XA4	4.4		
XA5	5.8		
XA6	38.6		
XA7	15.5		
XA8	66.4		
XA9	80.2		
XA10	46.2		
XA11	86.1		
XB1	0	(B1,B2)	8.1
XB2	19.0	(B2,B3)	14.8
XB3	29.5	(B3,B7)	3.0
XB4	48.4		
XB5	18.1		
XB6	35.6		
XB7	52.0	(B7,B11)	4.0
XB8	51.2		
XB9	38.6		
XB10	42.9		
XB11	53.3	(B11,B14)	1.3
XB12	52.6		
XB13	61.0		
XB14	84.7	(B14,B15)	2.0
XB15	86.1		
XC1	0		
XC2	28.5		
XC3	32.5	(C3,C4)	3.5
XC4	54.8	(C4,C5)	12.0
XC5	59.0	(C5,C8)	8.2
XC6	24.2		
XC7	77.1		
XC8	60.3	(C8,C9)	3.1
XC9	66.7	(C9,C11)	3.0
XC10	52.9		
XC11	70.5	(C11,C12)	13.2
XC12	86.1		

TABLE G7

DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 4

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$56,504,200
XN	79.5
$x_{A_4}^1$	.406340
x_{A_3}	.593660
x_{B_3}	1
x_{C_4}	.997488
x_C	.002512

TABLE G8

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 4

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0	(A1,A2)	1.0
XA2	2.0	(A2,A5)	0.7
XA3	20.2		
XA4	4.4		
XA5	4.1	(A5,A6)	1.1
XA6	35.8	(A6,A8)	2.5
XA7	15.5		
XA8	61.1		
XA9	74.9	(A9,A11)	1.3
XA10	46.2		
XA11	79.5		
XB1	0	(B1,B2)	9.7
XB2	17.4	(B2,B3)	17.7
XB3	25.0	(B3,B7)	3.6
XB4	45.8		
XB5	21.2		
XB6	35.5		
XB7	46.9	(B7,B11)	4.8
XB8	51.1		
XB9	44.5		
XB10	46.0		
XB11	47.4	(B11,B14)	1.6
XB12	58.5		
XB13	64.1		
XB14	78.5	(B14,B15)	2.4
XB15	79.5		
XC1	0		
XC2	29.8		
XC3	32.5	(C3,C4)	4
XC4	54.3	(C4,C5)	13.9
XC5	56.6	(C5,C8)	9.5
XC6	24.2		
XC7	77.1		
XC8	56.7	(C8,C9)	3.6
XC9	62.6	(C9,C11)	3.5
XC10	52.9		
XC11	65.9	(C11,C12)	15.2
XC12	79.5		

TABLE G9

DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 5

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$56,414,740
XN	79.5
$X_{A_5}^1$	.286404
$X_{A_5}^5$	.713596
X_{B_3}	1
X_{C_4}	.997488
X_C^*	.002512

TABLE G10

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 5

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0	(A1,A2)	1.2
XA2	1.8	(A2,A5)	0.8
XA3	20.2		
XA4	4.4		
XA5	3.8		
XA6	36.6	(A6,A8)	3.0
XA7	15.5		
XA8	61.4		
XA9	75.2	(A9,A11)	1.6
XA10	46.2		
XA11	79.5		
XB1	0	(B1,B2)	9.7
XB2	17.4	(B2,B3)	17.7
XB3	25.0	(B3,B7)	3.6
XB4	45.8		
XB5	21.2		
XB6	26.4		
XB7	46.9	(B7,B11)	4.8
XB8	42.0		
XB9	44.5		
XB10	46.0		
XB11	47.4	(B11,B14)	1.6
XB12	58.5		
XB13	64.1		
XB14	78.5	(B14,B15)	2.4
XB15	79.5		
XC1	0		
XC2	29.8		
XC3	32.5	(C3,C4)	4
XC4	54.3	(C4,C5)	13.9
XC5	56.6	(C5,C8)	9.5
XC6	24.2		
XC7	77.1		
XC8	56.7	(C8,C9)	3.6
XC9	62.6	(C9,C11)	3.5
XC10	52.9		
XC11	65.9	(C11,C12)	15.2
XC12	79.5		

TABLE G11

DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 6

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$56,390,249
XN_5	79.5
x_{A_6}	.588294
x_{A_5}	.411706
x_{B_3}	1
x_{C_4}	.997488
x_{C_5}	.002512

TABLE G12

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 6

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0	(A1,A2)	1.7
XA2	1.3	(A2,A5)	1.1
XA3	20.2		
XA4	4.4		
XA5	3		
XA6	35.8	(A6,A8)	2.5
XA7	15.5		
XA8	61.1		
XA9	74.9	(A9,A11)	1.3
XA10	46.2		
XA11	79.5		
XB1	0	(B1,B2)	9.7
XB2	17.4	(B2,B3)	17.7
XB3	25.0	(B3,B7)	3.6
XB4	45.8		
XB5	21.2		
XB6	26.4		
XB7	46.9	(B7,B11)	4.8
XB8	42.0		
XB9	44.5		
XB10	46.0		
XB11	47.4	(B11,B14)	1.6
XB12	58.5		
XB13	64.1		
XB14	78.5	(B14,B15)	2.4
XB15	79.5		
XC1	0		
XC2	29.8		
XC3	32.5	(C3,C4)	4
XC4	54.3	(C4,C5)	13.9
XC5	56.6	(C5,C8)	9.5
XC6	24.2		
XC7	77.1		
XC8	56.7	(C8,C9)	3.6
XC9	62.6	(C9,C11)	3.5
XC10	52.9		
XC11	65.9	(C11,C12)	15.2
XC12	79.5		

TABLE G13

DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 7

Dantzig-Wolfe Variables	Non-zero Values
Objective Function (z)	$56,380,285
XN^5	79.5
X_{A_7}	.372638
X_{A_5}	.627362
X_{B_3}	1
X_{C_4}	.997488
X_{C}	.002512

TABLE G14

TRANSLATION OF DANTZIG-WOLFE MASTER PROBLEM RESULTS - ITERATION 7

Event Node	Time of Event	Activity crashed	Amount Crashed
XA1	0	(A1,A2)	1.7
XA2	1.3	(A2,A5)	1.1
XA3	20.2		
XA4	4.4		
XA5	3		
XA6	35.8	(A6,A8)	1.6
XA7	15.5		
XA8	62.0		
XA9	75.8	(A9,A11)	2.2
XA10	46.2		
XA11	79.5		
XB1	0	(B1,B2)	9.7
XB2	17.4	(B2,B3)	17.7
XB3	25	(B3,B7)	3.6
XB4	45.8		
XB5	21.2		
XB6	26.4		
XB7	46.9	(B7,B11)	4.8
XB8	42		
XB9	44.5		
XB10	46.0		
XB11	47.4	(B11,B14)	1.6
XB12	58.5		
XB13	64.1		
XB14	78.5	(B14,B15)	2.4
XB15	79.5		
XC1	0		
XC2	29.8		
XC3	32.5	(C3,C4)	4
XC4	54.3	(C4,C5)	13.9
XC5	56.6	(C5,C8)	9.5
XC6	24.2		
XC7	77.1		
XC8	56.7	(C8,C9)	3.6
XC9	62.6	(C9,C11)	3.5
XC10	52.9		
XC11	65.9	(C11,C12)	15.2
XC12	79.5		

APPENDIX H: ACTIVITY "EXTENDING" DANTZIG-WOLFE MASTER PROBLEM RESULTS

Introduction

The inclusion of activity "extending" in the Base Model involves the introduction of some additional variables in the Base Model. These additional variables are incorporated into the Subproblems of the Dantzig-Wolfe Decomposition Model. The Dantzig-Wolfe Decomposition algorithm operates in the same manner as that described in Chapter III. To develop the Extended Base Model, the assumptions, notation, and general form of the Base Model are retained. This Appendix describes the additional assumptions, introduces the new variables, and presents the results of the Dantzig-Wolfe Master Problem on this Extended Base Model.

Extended Model Assumptions

The following assumption has been added to the Base Model Assumptions of Chapter III.

5) Activity "extending" is allowed up to an upper bound (N_{ij}) for each activity (i,j)

6) The time/cost tradeoff function for activity "extending" will be modeled as a linear relation at this aggregate level

While "extending" is allowed as a possibility for all activities, it may be applied to only a subset of activities is operationally required. Such a restriction would have the added benefit of reducing the model size.

Extended Model Notation

To incorporate activity "extending," the following variables are introduced. Let:

Z_{ij} = number of time units activity (i, j) extended

κ_{ij} = cost per time unit activity (i, j) extended (Z_{ij})

ϑ_{ijl} = personnel months per time unit activity (i, j) extended (Z_{ij})

N_{ij} = upper bound on Z_{ij}

Extended Model

Objective Function. The extended model's new variables change only the form of the minimum cost objective function in the Base Model. The second objective function form, minimizing program duration, remains unchanged. Introducing the Extended Model variables into the Base Model objective function gives the following:

$$z = (OH + eb) \cdot X_n + \sum_i \sum_j K_{ij} Y_{ij} - \sum_i \sum_j \kappa_{ij} Z_{ij}$$

The negative coefficient in front of the activity "extending" variables represent the program being rewarded for using less than the normal amount of resources for an activity. This reward is not a justification for the extension of a program, rather it allows a program decision maker to take resources away from non-critical path activities and apply those resources to critical path activities. In this model, a critical path activity would be extended only if the cost savings gained by extending the activity outweighed any potential additional overhead and bonus opportunity loss incurred by lengthening the overall program duration.

Budget Constraints. The budget constraints, both project and program level, for the Extended Model include the same summation term introduced in the Objective Function section above. Again, the negative coefficient represents the additional funds being made available to the project and program budgets. The budget constraints then have the following forms for project and program, respectively:

$$\sum_i \sum_j K_{ij} Y_{ij} - \sum_i \sum_j \kappa_{ij} Z_{ij} \leq B_k - \sum_i \sum_j D_{ij} \quad \forall k \text{ and } (i,j) \in I_k$$

$$\sum_i \sum_j K_{ij} Y_{ij} - \sum_i \sum_j \kappa_{ij} Z_{ij} \leq B - \sum_i \sum_j D_{ij} \quad \forall (i,j) \in I$$

Note that the Y_{ij} and Z_{ij} columns form dependent vectors in the constraint matrix.

Personnel-Month Constraints. The changes in the personnel-month constraints reflect the additional months made available by not applying the normal personnel-month requirements to "extended" activities. The equations for the personnel-month constraints for the project and the program, respectively, are given as follows:

$$\sum_i \sum_j \sum_l S_{ijl} Y_{ij} - \sum_i \sum_j \sum_l \vartheta_{ijl} Z_{ij} \leq PH_{kl}, \quad \forall k,l \text{ and } (i,j) \in I_k$$

$$\sum_i \sum_j \sum_l S_{ijl} Y_{ij} - \sum_i \sum_j \sum_l \vartheta_{ijl} Z_{ij} \leq PH_l, \quad \forall l \text{ and } (i,j) \in I$$

Due Date Constraints. The due date constraint of the Base Model remains unchanged. However, it now represent the absolute date that the program can not be extended beyond. If the activity extension variables, Z_{ij}, are not all bounded above by the N_{ij}, then this constraint acts as the limiting factor as to how long an activity can be extended.

Precedence Constraints. The precedence constraints now reflect the additional activity extension variables. If an activity is extended, then the number of time units extended is added to the normal duration time of the activity. The precedence relation constraints for the project and the program, respectively, are as follows:

$$- X_i + X_j + Y_{ij} - Z_{ij} \geq T_{ij}, \quad \text{for } (i,j) \in I_k \ \forall k.$$

$$- X_i + X_j + Y_{ij} - Z_{ij} \geq T_{ij}, \quad \text{for } (i,j) \in I.$$

Dantzig-Wolfe Decomposition Problem Formulation

Extended Model Master Problem. The Dantzig-Wolfe Master Problem follows the development discussed in Chapter 3 and is of the same form as given in Chapter 3. The changes to the program level constraints presented in this Appendix are reflected in the coupling constraint coefficient matrix variables, a_k, introduced in Chapter 3.

Extended Model Subproblems. The Dantzig-Wolfe Subproblems for the Extended Model reflect the changes made to the constraints of the Base Model as would be as follows:

Minimize $z = (c_k - \pi a_k)x_k$

subject to

$$\sum_i \sum_j K_{ij}Y_{ij} - \sum_i \sum_j \kappa_{ij}Z_{ij} \leq B_k - \sum_i \sum_j D_{ij} \quad \forall k \text{ and } (i,j) \in I_k$$

$$\sum_i \sum_j \sum_l S_{ijl}Y_{ij} - \sum_i \sum_j \sum_l \vartheta_{ijl}Z_{ij} \leq PH_{kl}, \quad \forall k,l \text{ and } (i,j) \in I_k$$

$$-X_i + X_j + Y_{ij} - Z_{ij} \geq T_{ij}, \quad \text{for } (i,j) \in I_k \ \forall k.$$

$$0 \leq Y_{ij} \leq M_{ij} \qquad \text{for } (i,j) \in I_k$$

$$0 \leq Z_{ij} \leq N_{ij} \qquad \text{for } (i,j) \in I_k$$

where

c_k = the objective function coefficients associated with project k

a_k = the coupling constraint coefficient matrix associated with project k

x_k = the variables associated with project k

π = the shadow prices associated with the common constraints received from the Master Problem

Extended Example

The illustrative example presented in Chapter 4 serves as the Base Model for the Extended Model. The coefficients for the activity "extending" costs are half of the coefficients for activity "crashing" $\left(\kappa_{ij} = 0.5 \cdot K_{ij}\right)$. The personnel-months coefficients are similar $\left(\vartheta_{ijl} = 0.5 \cdot S_{ijl}\right)$. The upper bound on the activity "extending" variables is twice the upper bound of the "crashing" variables $\left(N_{ij} = 2 \cdot M_{ij}\right)$. Of course, actual cost and savings would be utilized in an operational setting.

Results

The results from the Dantzig-Wolfe Decomposition Algorithm are similar to those presented in Chapter 4. In the solving of the subproblems, the activities are either "crashed", "extended", or left alone. However, the initial Master Problem iteration results provide convex combinations of activities that may have activities being both "crashed" and "extended." This occurs until an initial basic feasible solution is found in the Master Problem (remember we used a deviational variable to relax the budget constraint). See Figures 10 to see the Master Problem Iteration Results and Figure 11 to see the minimum personnel-months requirement at each iteration of the Dantzig-Wolfe Master Problem. Clearly, extending can be applied to the type of formulation, if it is justified by the operational setting.

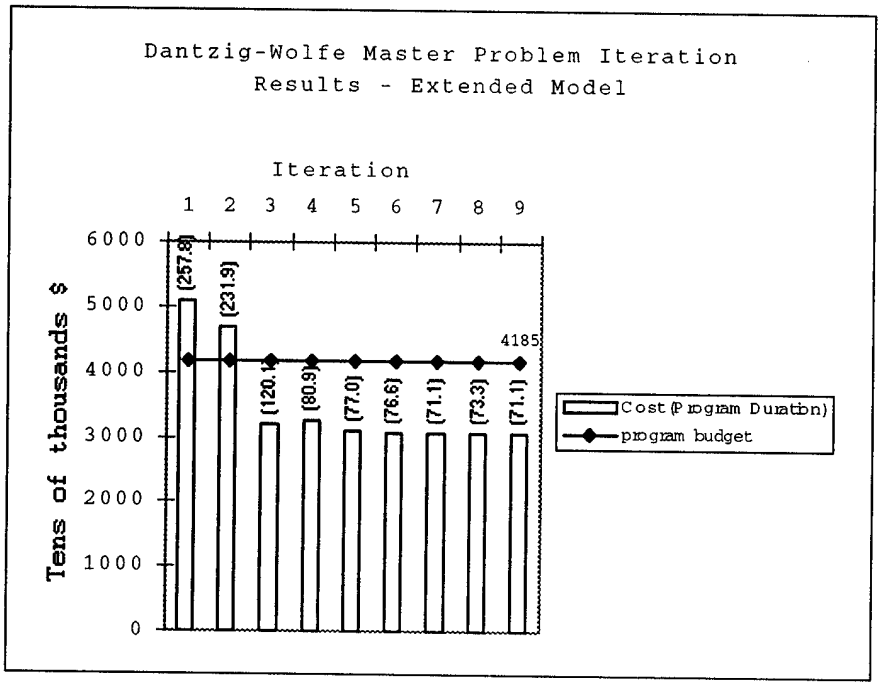

Figure 10. Dantzig-Wolfe Master Problem Iteration Results - Extended Model

Figure 10 lists the cost (vertical bars) and the program duration (in parentheses) at each iteration. However, the first two iterations bases contain the deviational variable introduced previously. At iterations 3, 4 and 5, the program schedules generated by the master program contain convex combinations of dependent vectors; therefore, these program schedules are meaningless in the extended model solution space. However, in iterations 6 to 9, the master problem solutions correspond to feasible program schedules in the extended model solution space.

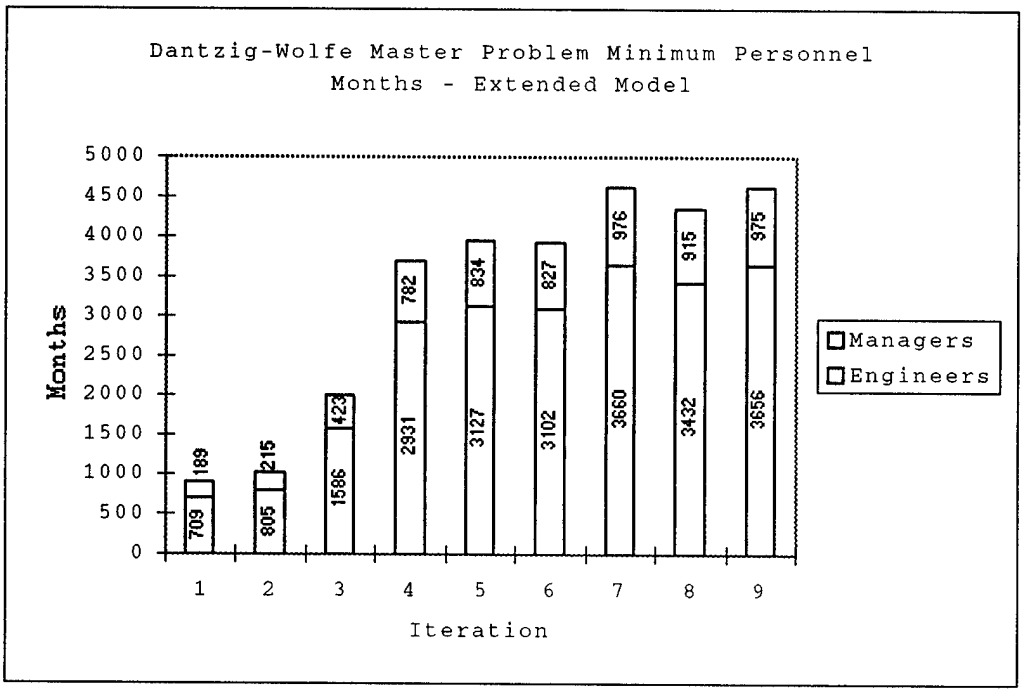

Figure 11. Dantzig-Wolfe Master Problem Minimum Personnel Months - Extended Model

Bibliography

1. Aardal, Karen and Aysen Ari. "On the resemblance between the Kornai-Liptak and Cross Decomposition Techniques for Block-Angular Linear Programs", European Journal of Operational Research, 46: 393-398 (1990).

2. Bates, Donald L. and David L. Eldridge. Strategy and Policy: Analysis, Formulation, and Implementation. Dubuque: Wm. C. Brown Company Publishers, 1980.

3. Baumol, William J. and Tibor Fabian. "Decomposition, Pricing for Decentralization and External Economies," Management Science, 11: 1-32 (1964).

4. Bitran, Gabriel R. and Devanath Tirupati. "Hierarchical Production Planning" in Handbooks in Operations Research and Management Science Volume 4: Logistics of Production and Inventory. Ed. Graves, S.C., A. H. G. Rinnooy Kan, and P.H. Zipkin. Amsterdam, The Netherlands: Elsevier Science Publishers B.V., 1993.

5. Candea, Dan. Issues of Hierarchical Planning in Multi-Stage Production Systems: Technical Report No. 134. Contract N00014-75-C-0556. Cambridge, MA: Operations Research Center, July 1977 (AD-A042559).

6. Charette, Wilfred and Walter Halverson. "Tools of Project Management", in The Implementation of Project Management: The Professional's Handbook. Ed. Linn C. Stuckenbruck, Ph.D. Reading, MA: Addison-Wesley Publishing Company, 1981.

7. Dantzig, George B. and Philip Wolfe. "Decomposition Principle for Linear Programs," Operations Research, 8: 101-111 (1960).

8. Deckro, RF, JE Hebert, and WA Verdini. "Project Scheduling with Work Packages," OMEGA International Journal of Management Science, 20: 169-182 (1992).

9. Deckro, Richard F. and John E. Hebert. "Modeling Diminishing Returns in Project Resource Planning", Article - Pre-Publication Draft. June 1995.

10. Deckro, Richard F, E.P. Winkofsky, John E. Hebert, and Roger Gagnon. "A Decomposition Approach to Multi-Project Scheduling," European Journal of Operational Research, 51: 110-118 (1991)

11. Deckro, RF and JE Hebert. "Resource Constrained Project Crashing," OMEGA International Journal of Management Science, 17: 69-79 (1989).

12. Demeulemeester, Erik, Willy Herroelen, Wendell P. Simpson, Sami Baroum, James H. Patterson, and Kum-Khiong Yang. "On a paper by Christofides et al. for Solving the Multiple-Resource Constrained, Single Project Scheduling Problem", European Journal of Operational Research, 76: 218-228 (1994).

13. Demeulemeester, Erik and Willy Herroelen. "A Branch-and-Bound Procedure for the Multiple Resource-Constrained Project Scheduling Problem", Management Science, 38: 1803-1818 (1992).

14. Department of the Air Force. Course handout, Acquisition Fundamentals. Air Training Command, 3400 Technical Training Wing, 3440 Technical Training Group, Lowry AFB CO, June 1992.

15. Department of Defense. Defense Acquisition. DOD Directive 5000.1. Washington: USD(A), 23 February 1991.

16. Department of Defense. Defense Acquisition Management Policies and Procedures. DOD Instruction 5000.2. Washington: USD(A), 23 January 1991.

17. Dincerler, Abdurrezak. Project Scheduling in Project-Oriented Production Systems: ORC-85-11. Contract N00014-76-C-0134. Berkeley, CA: Operations Research Center, September 1985 (AD-A159980).

18. Dowden, Simon B. "The Project Manager in Pharmaceutical R&D", PM Network, 9: 15-22 (May 1995).

19. Edmunds, T. A. and J. F. Bard. "A Decomposition Technique for Discrete Time Optimal Control Problems with an Application to Water Resources Management", Mathematical Computer Modelling, 13: 61-78 (1990).

20. Erengue, S. Selcuk, Suleyman Tufecki, and Christopher J. Zappe. "Solving Time/Cost Trade-Off Problems with Discounted Cash Flows Using Generalized Benders Decomposition", Naval Research Logistics, 40: 25-50 (1993).

21. Geoffrion, Arthur M. "Elements of Large-Scale Mathematical Programming", Management Science, 16: 652-691 (1970).

22. Geoffrion, Arthur M. "The Purpose of Mathematical Programming is Insight, Not Numbers", Interfaces, 7: 81-92 (1976).

23. Geoffrion, A. M. and G. W. Graves. "Multicommodity Distribution System Design by Benders' Decomposition", Management Science, 20: 822-844 (1974).

24. Haurie, A. "Time Scale Decomposition in Production Planning for Unreliable Flexible Manufacturing Systems", European Journal of Operational Research, 82: 339-358 (1995).

25. Ho, J. K. and E. Loute. "Computational Experiences with Advanced Implementation of Decomposition Algorithm for Linear Programming", Mathematical Programming, 27: 283-290 (1983).

26. Ho, James K. and Etienne Loute. "An Advanced Implementation of the Dantzig-Wolfe Decomposition Algorithm for Linear Programming", Mathematical Programming, 20: 303-326 (1981).

27. Holmberg, Kaj. "Cross Decomposition Applied to Integer Programming Problems: Duality Gaps and Convexification in Parts", Operations Research, 42: 657-668 (1994).

28. Holmberg, Kaj. "Linear Mean Value Cross Decomposition: A Generalization of the Kornai-Liptak Method", European Journal of Operational Research, 62: 55-73 (1992).

29. Iyer, A. V., John J. Jarvis, and H. Donald Ratliff. Network Aggregation Concepts: PDRC 85-04. Contract N00014-83-K-0147 and N00014-84-C-0578. Atlanta, GA: School of Industrial and Systems Engineering, May 1985 (AD-A168247).

30. Kelley, James E. "Critical-Path Planning and Scheduling: Mathematical Basis*", Operations Research, 9: 296-320.

31. Kerzner, Harold. Project Management: A Systems Approach to Planning, Scheduling, and Controlling (Fifth Edition). New York: Van Nostrand Reinhold, 1995.

32. Kurtulus, I. and E. W. Davis. "Multi-Project Scheduling: Categorization of Heuristic Rules Performance", Management Science, 28: 161-172 (1982).

33. Lasdon, Leon S. Optimization Theory for Large Systems. New York: MacMillan Publishing Co, Inc., 1970.

34. Lee, Choong Y. "A Cross Decomposition Algorithm for a Multiproduct-Multitype Facility Location Problem", Computers and Operations Research, 20: 527-540 (1993).

35. Lee, Choong Y. "Application of a Cross Decomposition Algorithm to a Location and Allocation Problem in Distributed Systems", Computer Communications, 18: 367-377 (1995).

36. Michelon, Philippe and Nelson Maculan. "Lagrangean Decomposition for Integer Nonlinear Programming with Linear Constraints", Mathematical Programming, 52: 303-313 (1991).

37. Muther, Richard, PE, CMC, ScD(hc). High Performance Planning. Kansas City: Management and Industrial Research Publications, 1988.

38. Nagurney, Anna and Dae-Shik Kim. "Parallel Computation of Large-Scale Dynamic Market Network Equilibria via Time Period Decomposition", Mathematical Computer Modelling, 15: 55-67 (1991).

39. Patterson, James H. "A Comparison of Exact Approaches for Solving the Multiple Constrained Resource, Project Scheduling Problem", Management Science, 30: 854-867 (1984).

40. Pritsker, A. Alan B., Lawrence J. Watters, and Philip M. Wolfe. "Multiproject Scheduling with Limited Resources: A Zero-One Programming Approach", Management Science, 16: 93-108 (1969).

41. Reinoso, Hernaldo and Nelson Maculan. "Lagragean Decomposition in Integer Linear Programming: A New Scheme", Information Systems and Operational Research, 30: 1-5 (1992).

42. Sharda, Ramesh. "Linear Programming Solver Software for Personal Computers: 1995 Report", OR/MS Today: 49-57 (October 1995).

43. Steiner, George A. "The Systems Approach in Planning", in Long-Range Planning for Management (Third Edition). Ed. David W. Ewing. New York: Harper & Row Publishers, 1972.

44. Stuckenbruck, Linn C. "The Job of the Project Manager: Systems Integration", in The Implementation of Project Management: The Professional's Handbook. Ed. Linn C. Stuckenbruck, Ph.D. Reading, MA: Addison-Wesley Publishing Company, 1981.

45. Sweeney, Dennis J. and Richard A. Murphy. "A Method of Decomposition for Integer Programs", Operations Research, 27: 1128-1141 (1979).

46. Vercellis, Carlo. "Constrained Multi-Project Planning Problems: A Lagrangean Decomposition Approach", European Journal of Operational Research, 78: 267-275 (1994).

47. Weist, Jerome D. and Ferdinand K. Levy. A Management Guide to PERT/CPM with GERT/PDM/DCPM and other Networks (Second Edition). Englewood Cliffs: Prentice-Hall, Inc., 1977.

48. Winston, Wayne L. Operations Research: Applications and Algorithms (Third Edition). Belmont: Wadsworth Publishing Company, 1994.

Vita

Capt Victor D. Wiley [redacted] He graduated from Douglas MacArthur High School in 1987 and entered undergraduate studies at Texas A&M University in College Station, Texas. He graduated with a Bachelor of Science degree in Applied Mathematical Sciences on 16 August 1991 and received his commission on that same date. His first assignment was at Wright-Patterson AFB as a program management officer. In August 1994, he entered the School of Engineering, Air Force Institute of Technology.

Permanent Address: [redacted]

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Mar 1996	Final

4. TITLE AND SUBTITLE
Optimization Analysis for Design and Planning of Multi-Project Programs

5. FUNDING NUMBERS

6. AUTHOR(S)
Victor D. Wiley, Capt, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
AFIT/ENS

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GOR/ENS/96M-18

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unlimited distribution

12b. DISTRIBUTION CODE
A

13. ABSTRACT *(Maximum 200 words)*

Program Management concerns the long-term planning, coordination, and control of major technological, engineering, scientific, and/or developmental activities. In general, programs tend to be exceptionally large, consisting of several parallel or sequential projects or groups of projects. While many of the approaches developed for projects have been used in program management, there are critical differences between programs and projects. A key difference is the magnitude of programs, both in program duration and number of tasks to be coordinated. In addition, project parameters are driven by program decisions.

While a large number of modeling efforts have focused at the project level, very little optimization literature deals directly with aspects of initial program design and development. This thesis effort looks at the application of optimization techniques to the initial design and development of multi-project programs. The classic work breakdown structure is used as a framework for Dantzig-Wolfe Decomposition to provide an aggregate model for investigating the effects of funding levels, resource allocation, and program durations.

14. SUBJECT TERMS
Program Management, Program Planning, Program Analysis, Decomposition

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements***.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	**PR**	- Project
G	- Grant	**TA**	- Task
PE	- Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- **DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- **DOE** - See authorities.
- **NASA** - See Handbook NHB 2200.2.
- **NTIS** - Leave blank.

Block 12b. Distribution Code.

- **DOD** - Leave blank.
- **DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- **NASA** - Leave blank.
- **NTIS** - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.