Semidefinite Cuts and Partial Convexification Techniques with Applications to Continuous Nonconvex Optimization, Stochastic Integer Programming, and Facility Layout Problems

Barbara M. P. Fraticelli

Dissertation Submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Hanif D. Sherali, Chair
Ebru Bish
Terry Herdman
C. Patrick Koelling
Subhash Sarin

April 6, 2001
Blacksburg, Virginia

Keywords: Reformulation-Linearization Technique (RLT), semidefinite programming (SDP), stochastic programming, mixed-integer programming, facility layout problem (FLP), disjunctive models.

Semidefinite Cuts and Partial Convexification Techniques with Applications to Continuous Nonconvex Optimization, Stochastic Integer Programming, and Facility Layout Problems

Barbara M. P. Fraticelli

(ABSTRACT)

Despite recent advances in convex optimization techniques, the areas of discrete and continuous nonconvex optimization remain formidable, particularly when globally optimal solutions are desired. Most solution techniques, such as branch-and-bound, are enumerative in nature, and the rate of their convergence is strongly dependent on the accuracy of the bounds provided, and therefore, on the tightness of the underlying formulation. This research develops both general and problem-specific procedures to be used in conjunction with the Reformulation-Linearization Technique (RLT) for generating tight model formulations for challenging nonconvex optimization problems. These problems include the general classes of nonlinear and integer programs, as well as specific applications within these areas. We begin by deriving a new class of cutting planes, called *semidefinite cuts*, for enhancing the solution of nonconvex optimization problems. While these cuts can be generally applied to either discrete or continuous nonconvex problems, we specifically demonstrate their effectiveness in solving quadratic optimization problems. We then focus on the important class of mixed-integer programming (MIP) problems, and develop a new decomposition technique. This methodology is particularly well-suited to solve stochastic integer programming problems, arguably, the most difficult class of discrete problems. Finally, we address a specific MIP application, known as the facility layout problem, that has defied exact solution methods, and which subsumes the notorious quadratic assignment problem. We significantly advance the state-of-the-art in solving these problems by developing substantially improved models and algorithms through outer-linearization techniques and concepts from disjunctive programming.

Our first contribution proposes a mechanism to tighten RLT-based relaxations for general problems in nonconvex optimization by importing concepts from semidefinite programming (SDP), leading to a new class of *semidefinite cutting planes*. Given an RLT relaxation, the usual nonnegativity restrictions on the matrix of RLT product variables is replaced by a suitable positive semidefinite constraint. Instead of relying on specific SDP solvers, the positive semidefinite stipulation is re-written to develop a semi-infinite linear programming representation of the problem, and an approach is developed that can be implemented using traditional optimization software. Specifically, the infinite set of constraints is relaxed, and members of this set are generated as needed via a separation routine in polynomial time. In essence, this process yields an RLT relaxation that is augmented with valid inequalities, which are themselves classes of RLT constraints that we call *semidefinite cuts*. We illustrate the use of this strategy by applying it to the case of optimizing a nonconvex quadratic objective function over a simplex. Several implementation variants of this basic concept are delineated and computationally explored. The results indicate that the cutting plane algorithm provides a significant tightening of the lower bound obtained by using RLT alone. Moreover, when used within a branch-and-bound framework, the proposed lower bound substantially reduces the effort

required to obtain globally optimal solutions. On average, the semidefinite cuts have reduced the number of nodes in the branch-and-bound tree by a factor of 37.6, while decreasing solution time by a factor of 3.4. The semidefinite cuts have also led to a significant reduction in the optimality gap at termination for larger problem instances, in some cases producing optimal solutions for problems that could not be solved using RLT alone within the allowable size and time limits. We have also proposed a method for generating semidefinite cuts to enhance higher order levels of RLT, thus enabling the semidefinite concept to be extended to orders higher than two for the first time in the literature.

Next, we consider a modification of Benders' decomposition method, using concepts from the Reformulation-Linearization Technique (RLT) and lift-and-project cuts, in order to develop an approach for solving discrete optimization problems that yield integral subproblems, such as those that arise in the case of two-stage stochastic programs with integer recourse. We first demonstrate that if a particular convex hull representation of the problem's constrained region is available when binariness is enforced on only the second-stage (or recourse) variables, then the regular Benders' algorithm is applicable. The proposed procedure is based on sequentially generating a suitable partial description of this convex hull representation as needed in the process of deriving valid Benders' cuts. We also show how this procedure can be applied even more efficiently to the case of stochastic programs, by exploiting the dual angular structure that they possess. The key idea is to design an RLT or lift-and-project cutting plane scheme for solving the subproblems where the cuts generated have right-hand sides that are functions of the first-stage variables. Hence, we are able to re-use these cutting planes from one subproblem solution to the next simply by updating the values of the first-stage decisions. The proposed Benders' cuts also recognize these RLT or lift-and-project cuts as functions of the first-stage variables, and are hence shown to be globally valid, thereby leading to an overall finitely convergent solution procedure. An illustrative example is provided to elucidate the proposed approach. The focus is on developing a first comprehensive finitely convergent extension of Benders' methodology for problems having 0-1 mixed-integer subproblems, as in the aforementioned context of two-stage stochastic programs with integer recourse.

Finally, we develop a substantially improved mixed-integer programming (MIP) modeling and algorithmic approach for the facility layout problem. Given a rectangular building, and area requirements along with aesthetic ratios for each department, the problem is to determine the dimensions and location of each (rectangular) department within the building in order to minimize the total travel cost (number of trips times the distance) between all departments. The distance between departments is measured as the rectilinear distance separating their respective centroids. Although the facility layout problem can be stated rather simply, it is extremely difficult to solve to optimality, even for small problem instances. The difficulty arises from the nonlinear area constraints for each department and the disjunctive constraints that no two departments can overlap. Existing models for this problem have been unable to even capture an adequately accurate linearized representation of the nonlinear area constraints that would yield a tractable model formulation. Motivated by this dearth, we focus on developing several model enhancements for producing more accurate solutions while also decreasing the solution effort required. In order to represent the nonlinear area constraints, we begin by strengthening the bounds on the departmental dimensions, and then derive a novel polyhedral outer approximation scheme that can provide as accurate a representation as desired.

We also develop and evaluate the performance of several classes of valid inequalities, as well as alternative methods for reducing problem symmetry. Finally, we explore the construction of partial convex hull representations for the disjunctive constraints that are used to prohibit the overlapping of departments. These proposed enhancements have been evaluated using an AMPL interface with CPLEX, and compared with published results to gauge their effectiveness. The results indicate a *substantial* increase in the accuracy of the layout produced, while at the same time, providing a *dramatic* reduction in computational effort. Overall, the maximum error in department size was reduced from over 6% to nearly zero, while solution time decreased by a factor of 110. Previously unsolved test problems from the literature that had defied even approximate solution methods have been solved to exact optimality using our proposed approach.

Acknowledgements

First and foremost, I wish to extend my heartfelt thanks to Dr. Hanif Sherali for all of the help and guidance he has provided me over these past four years. He is truly a role model in teaching excellence, research excellence, humility, and kindness. I have learned so much from working with him and from observing the way he lives his life. Similarly, I express appreciation to the remaining members of my committee, all of whom have helped me significantly throughout this process. In particular, I would like to thank Dr. Ebru Bish, Dr. Pat Koelling, and Dr. Subhash Sarin for all your help and guidance from the prelim stage right through my final defense. I would also like to thank Dr. Stanley Suboleski for serving on my committee until his retirement, and Dr. Terry Herdman for graciously agreeing to take his place and join the committee in midstream. I value the insights that the committee members have given me with respect to research and life in academia, as well as the relationships we have developed.

I thank Ms. Lovedia Cole for helping me navigate through all the required paperwork along the way, and for always lending a kind, listening ear to me. Lovedia is a great asset to the entire ISE department, and graduate students in particular. In addition, I would like to thank Burak Ozdaryal for being a wonderful friend and for proofreading countless papers over the past few years, and Cole Smith for teaching me everything there is to know about CPLEX (and a few other things along the way). Thanks also to my good friends Greg Beskow, Elise Caruso, Felipe Helo, Qing Li, Laurent Matthey, Ian Rehmert, Mardi Russell, Mukund Venkatesan, and Fernando Vittes for making this whole experience a lot of fun.

I also would like to thank the members of my family who have been supportive of all my efforts. Namely, Frederick S. Priebe (father), Patricia G. Weikert (mother), Pamela and Jose Cowen (sister and brother-in-law), and Thomas M. and Theresa Fraticelli (mother- and father-in-law). I know the 5-hour drive to visit us in Virginia wasn't always a lot of fun, and I appreciate your understanding in accepting our less frequent visits to see you.

Finally, there is no way to adequately thank my husband, Thomas D. Fraticelli, for all he has done to make this venture a success. From moving here and starting a new job, to understanding my need to work in the evenings and weekends, to keeping our house clean and our kitties fed. I've told you before, but I thought I'd put it here in writing so you have physical proof, you are the best husband in the world and I am grateful for your love every day of my life.

Contents

List of Figures

List of Tables

Chapter 1: Introduction and Motivation

While efficient solution techniques have been developed for certain types of convex optimization problems, particularly linear programming problems, there are few efficient algorithms for discrete or continuous nonconvex optimization problems. The most typical solution techniques are enumerative in nature, including the typical branch-and-bound strategy and its variants, and their performance is highly dependent on the strength of the bounding mechanisms employed. In order to derive accurate problem bounds, it is essential to develop tight model formulations, through the use of both general-purpose and problem-specific strategies. This dissertation focuses on developing both general-purpose and problem-specific strategies to strengthen model formulations for continuous nonconvex optimization, stochastic integer programming, and facility layout problems.

1.1 Motivation

As evident from several studies in the literature, there is a clear need for deriving tight formulations for nonconvex optimization problems in order to develop effective solution methods. Our focus in this dissertation will be to build upon existing methods and concepts for obtaining such tight reformulations. Solution techniques for solving nonconvex optimization problems typically involve a reformulation step of relaxing some of the complicating constraints, but then augmenting this with a set of additional suitable restrictions that are implied for any feasible solution and that tighten the resulting formulation. The goal of problem relaxation is to obtain a formulation that is significantly easier to solve, and yet provides a sufficiently accurate approximation for the original problem. Since some of the restrictions on the problem have been eliminated, any solution to the relaxation gives a best-case bound for the original problem. A feasible solution to the original problem provides a worst-case bound, and these two bounds can be used in conjunction to search for globally optimal solutions. It is essential to have tight bounds if this search is to be computationally effective. In this dissertation, we rely heavily on the relaxation strategy known as the Reformulation-Linearization Technique (RLT) (see Sherali and Adams (1990, 1994, 1999)), and we develop several extensions and specializations of this technique to tighten model formulations through general and problem-specific insights.

There are many commonly used techniques for obtaining relaxations for nonconvex optimization problems. In discrete optimization, for instance, a basic strategy is to relax the integrality restrictions to produce a linear programming relaxation. While such a relaxation is easy to solve, it is typically not a tight formulation, and therefore provides weak bounds and leads to a significant effort in the overall search process. An alternative relaxation strategy relies on semidefinite programming (SDP). SDP is similar to linear programming, except that the vector of variables is replaced by a matrix, and the non-negativity restrictions are replaced by the restriction that the matrix of variables should be positive semidefinite. SDP has been used to provide relatively tight relaxations for discrete and continuous optimization problems, but their solution requires specialized SDP solvers. On the other hand, the RLT approach is amenable to

the use of standard LP solvers, while providing a significant tightening beyond the basic LP relaxation. The RLT strategy, which is a unifying approach for solving discrete and continuous nonconvex optimization problems (see Sherali and Adams (1999) for a comprehensive exposition), is to suitably multiply appropriate constraints by nonnegative bound-factors, constraint-factors, or simply variables in a reformulation phase, and then to replace the products of original variables by new variables in order to derive a higher-dimensional lower bounding linear programming (LP) relaxation for the original problem. This RLT process can actually generate a hierarchy of tighter relaxations, depending on the types of factor products employed in the reformulation phase. In practice, however, the lowest-level RLT relaxation (as dictated by the nature of the terms in the original problem) is most frequently implemented in order to control the size of the resulting relaxation, although higher-level relaxations have been successfully used in certain special applications. In order to close the gap between these lower level RLT relaxations and the convex hull of feasible solutions, it is often helpful to incorporate additional classes of RLT constraints.

In deriving effective solution strategies, it is essential to strengthen the relaxation employed by including suitable additional restrictions in order to obtain tighter bounds. However, including a large number of additional restrictions can significantly increase the size of the problem, thereby making it more difficult to solve. As a compromise between tightening the bound and increasing the problem size, many solution strategies iteratively solve the relaxation and then add to it cutting planes or valid inequalities that are implied for any feasible solution but are violated by the current solution of the relaxation. These inequalities are used to "cut off" the current solution in an attempt to drive the best-case bound to more closely approximate the true solution value. Some types of cutting planes, such as Gomory's fractional cuts for discrete optimization problems, are generic to entire classes of problems, while others have been derived for specific applications. Among the general-purpose techniques, we will discuss applications of disjunctive programming and RLT (or lift-and-project) cuts. These cuts are used to derive tight approximations for the convex hull of feasible solutions. In addition, we will develop a new general class of RLT cuts, based upon concepts from semidefinite programming, that are valid for any nonconvex optimization problem. We will also explore the use of some valid inequalities that are problem-specific, particularly in the case of the facility layout problem.

The focus of this dissertation is to develop tight problem formulations in order to lead to improved solution techniques for general and specific applications of nonconvex optimization. Specifically, we concentrate on three applications of nonconvex optimization: continuous nonconvex problems, stochastic programs with integer recourse, and mixed-integer programming (MIP) formulations for the facility layout problem. In each of these problems, there is a particular need for developing tight model formulations in order to enhance algorithmic performance. For continuous nonconvex optimization problems, the search for a global optimum is typically obtained through branch-and-bound enumeration. Without tight bounds, the enumeration tree would continue to explore non-improving areas of the solution space and thereby dramatically increase the overall computational effort. In the context of stochastic programs, typical solution strategies involve solving the subproblems repeatedly, for many different realizations of the first-stage variables. When these subproblems involve integer variables, the overall solution effort can become prohibitive unless we are able to approximate the convex hull of feasible solutions for the subproblems. Finally, in the case of facility layout

problems, the nature of these MIP problems has made it difficult to solve even moderately sized instances to optimality. Moreover, existing model formulations provide relatively poor approximations to the underlying nonlinear problem. We demonstrate that improved formulations can enhance both the accuracy and solvability of these problems, thereby facilitating the derivation of optimal or good provable quality solutions to larger instances of this class of problems.

1.2 Research Goals

We state the goals of this research in terms of the three areas discussed above.

The first portion of this dissertation focuses on using concepts from semidefinite programming to develop a new class of cutting planes that can be used to enhance general RLT relaxations. We explore several alternative cut generation strategies and evaluate their performance to identify the most effective techniques for deriving tighter bounds. We also study their effectiveness in determining globally optimal solutions within a branch-and-bound framework. The effectiveness of these cuts is demonstrated for a class of problems in which a nonconvex quadratic objective function is minimized over a simplex.

The next endeavor is concerned with developing a modified Benders' partitioning strategy to solve discrete optimization problems that decompose into discrete subproblems. In particular, we develop a finitely convergent algorithm for solving the subproblems via an RLT-based cutting plane approach, while using these subproblem cuts to obtain valid Benders' cuts, and lifting them to enable their re-use for subsequent visits to the subproblem. The design of this approach is particularly motivated by the case of stochastic programs with integer recourse.

In the final portion of this dissertation, we develop a new, more accurate mixed-integer programming formulation for the facility layout problem, and design a series of enhancements to this model by tightening relaxations through symmetry-breaking considerations, valid inequalities, and partial convex hull constructions. The effectiveness of these proposed enhancements is evaluated in comparison with previously published approaches using several test problems that have been addressed in the literature.

1.3 Organization of the Dissertation

The remainder of this dissertation is organized as follows. Chapter 2 provides a review of the literature relevant to this research, beginning with a discussion on the need for deriving tight formulations. This leads to a description of several popular solution techniques for solving nonconvex optimization problems, including the Reformulation-Linearization Technique (RLT), semidefinite programming (SDP), cutting planes, and Benders' decomposition. Elements of each of these are included in the development of the solution techniques presented in the ensuing chapters. The remainder of Chapter 2 reviews the literature for some specific applications of nonconvex optimization, including stochastic programming and facility layout problems.

Chapter 3 uses concepts from semidefinite programming to create a new class of cuts that enhance RLT relaxations for general nonconvex optimization problems. Based on the fact that the traditional matrix of second-order (or more generally, even-ordered) RLT variables can be restricted to be positive semidefinite, we develop a polynomial-time framework to determine whether this matrix satisfies the stated positive semidefiniteness requirement, and if not, we generate valid linear inequalities that delete the current approximating solution. This leads to an iterative process whereby the semidefinite relaxation of a problem is solved via a series of linear programming problems, beginning with an initial RLT relaxation. We have proposed several variations for generating these SDP cuts, and we demonstrate their use on a class of continuous nonconvex quadratic optimization problems. Computational results are presented to exhibit the strong effectiveness of the proposed cuts in providing tighter bounds, and in substantially decreasing the overall solution effort within the context of an exact branch-and-bound solution strategy.

As opposed to the general-purpose model strengthening procedures developed in Chapter 3, Chapters 4 and 5 design problem-specific techniques for tightening problem formulations. Chapter 4 presents a modified Benders' partitioning strategy for solving discrete optimization problems that yield discrete subproblems. This methodology is particularly motivated by the class of stochastic integer programs with mixed-integer recourse. The proposed procedure solves the resulting subproblems through a series of cutting planes that approximate the convex hull of solutions, and these cutting planes are then used to derive valid Benders' cuts. In order to enhance algorithmic performance, we also develop various lifting techniques that render the generated cuts to be globally valid for all subsequently solved subproblems. In addition, we develop a specialization for stochastic programming problems that exploits the dual angular structure that they possess. The proposed solution strategy is the first comprehensive extension of Benders' methodology for problems having 0-1 mixed-integer subproblems.

Chapter 5 addresses the facility layout problem, a discrete optimization problem that has proven difficult to solve to optimality, even for moderately sized problem instances. We review a published mixed-integer programming formulation for this problem, and then propose a series of enhancements that are designed to provide more accurate solutions to the underlying nonlinear, nonconvex problem, as well as decrease solution effort. We begin by deriving a novel polyhedral outer approximation scheme that can provide as accurate a representation as desired for the nonlinear area requirements for each department. We also develop and evaluate the performance of several classes of valid inequalities, alternative methods for reducing problem symmetry, and certain partial convex hull constructions for the disjunctive constraints that are used to prohibit the overlapping of departments. The results indicate a substantial increase in the accuracy of the layout produced, as well as a dramatic reduction in computational effort. Finally, Chapter 6 provides a summary and conclusion, along with recommendations for future research.

Chapter 2: Literature Review

This chapter contains a summary of literature that is relevant to the topics covered in the remainder of the dissertation. Section 2.1 discusses the importance of deriving tight formulations for nonconvex optimization problems, particularly within the context of branch-and-bound or other enumerative approaches. Section 2.2 reviews some of the more common solution techniques for nonconvex optimization problems, beginning with traditional methods (cutting planes, enumerative methods, and partitioning strategies) and concluding with the more modern approaches of disjunctive programming, the Reformulation-Linearization Technique, and semidefinite programming. Section 3.3 reviews the literature on two areas (namely, stochastic programming and facility layout problems) for which we will later propose new conceptual approaches and techniques.

2.1 Model Formulations

From the onset of operations research as a field, the key to obtaining meaningful results has relied upon formulating a mathematically correct model that accurately characterizes the situation being studied. For most problems, there are many representations that constitute mathematically correct models, and in some cases (for example moderately sized linear programming problems), most of these equivalent representations can be solved within a reasonable amount of time given sophisticated solver routines. In the case of nonconvex optimization, however, the problems become substantially more challenging to solve, and the model formulation can have a significant impact on the effort required to solve the problem to optimality. As we will discuss throughout this dissertation, most solution techniques for nonconvex optimization problems rely heavily on solving successive (typically linear programming based) approximations to the underlying problem. The tightness of these approximations directly affects how accurately they reflect the original problem and, in turn, influences the amount of effort required to solve the problem to global optimality. In order to develop such tight problem relaxations, it is essential to seek *good* models to represent these problems, rather than just models that are mathematically correct.

Sherali and Driscoll (2000) provide an excellent discussion on the importance of tight formulations for discrete optimization problems, which they illustrate with the following fixed-charge location problem example, in which up to *m* supply facilities (having capacities $s_1, s_2, \ldots, s_m$) are to be constructed. The objective is to minimize the cost of construction plus the cost of shipping from the constructed supply centers to the set of *n* customers having demands $d_1, d_2, \ldots, d_n$. The variables $x_{ij} \geq 0$ represent the amount shipped from facility *i* to customer *j*, and y_i is a binary variable that equals one if facility *i* is constructed and zero otherwise. If facility *i* is *not* constructed ($y_i = 0$), the model must then ensure that $x_{ij} = 0 \ \forall j$. As a first-step

in formulating this restriction, consider the constraint $\sum_{j=1}^{n} x_{ij} \le s_i y_i$ for $i = 1, \ldots, m$. If $y_i = 0$, this constraint (coupled with $x_{ij} \ge 0$) forces $x_{ij} = 0 \ \forall j$, and if $y_i = 1$, it simply enforces that the total amount shipped out of facility *i* is at most its supply s_i. Although this formulation is mathematically correct, it can be substantially strengthened by including the constraints $0 \le x_{ij} \le y_i \min\{s_i, d_j\} \ \forall i, j$. These constraints are clearly satisfied in the discrete sense, since they force $x_{ij} = 0 \ \forall j$ when $y_i = 0$ and otherwise limit the amount shipped between supply *i* and customer *j* to be within the logically implied bounds. Although these additional restrictions are implied for the discrete case, they significantly tighten the linear programming relaxation by including variable upper bounds on each individual x_{ij}. This tightening of the relaxation has been amply demonstrated to significantly improve the solvability of the model.

The foregoing example illustrates how formulations can be strengthened by making logical inferences based on the problem structure. In addition, in the case of discrete optimization, it is sometimes possible to examine the problem data and alter some of the constraint coefficients to provide even tighter approximations through a technique known as *coefficient reduction* (see Nemhauser and Wolsey (1998) for example). In other cases (see Sherali and Smith (1999), for example) a natural symmetry in the problem can produce a series of solutions that, while appearing to be different numerically, each represent an equivalent set of decisions. In such cases, a solver could spend a great deal of time examining these solutions, not recognizing that they in fact represented the same situation. In order to reduce such inherent problem symmetry and thus speed the solution process, a set of *symmetry breaking constraints*, or *hierarchical constraints* as proposed by Sherali and Smith, can often be developed. We will apply these conceptual techniques in our study of a facility layout problem as described in Chapter 5.

2.2 Solution Techniques for Nonconvex Optimization Problems

There are several general techniques that have been developed for nonconvex optimization. We now review some of the relevant solution methods, beginning with the classical techniques of cutting planes, enumeration, and Benders' decomposition. Following this, we address some of the more recent developments, including disjunctive programming, the Reformulation-Linearization Technique (RLT), and semidefinite programming. We note that rather than relying solely on one of these specific techniques, it is typically more effective to combine these strategies, along with problem specific insights, while designing solution algorithms. For more on the importance of using hybrid algorithms of this type, we refer the reader to Hoffman and Padberg (1985, 1991) and Padberg and Rinaldi (1987).

2.2.1 Cutting Planes

As mentioned previously, the basic technique for solving nonconvex optimization problems relies on solving a sequence of relaxations that produce tighter and tighter

approximations to the original problem. (In this dissertation, the relaxations that we consider, although possibly derived in a higher-dimensional space, will typically be linear programming representations.) Since any relaxation has weakened the set of constraints imposed by the original problem, and thereby expanded the feasible region for the problem, the solution to the relaxed problem provides a best-case bound on the original problem. If the solution to the relaxed problem satisfies all of the relaxed restrictions, it is feasible and therefore optimal to the original problem as well. If the solution violates some of these relaxed constraints, we need to include additional restrictions to force the relaxed solution towards an optimal solution for the original problem.

Cutting planes, or *valid inequalities*, are such additional restrictions that are satisfied by every feasible solution to the original constraints, but are violated by the current solution to the relaxed problem. These constraints are said to "cut off" the current solution to the relaxed problem and force the feasible region of the relaxed problem to more closely approximate that of the original problem. The revised relaxation would then be solved, and if its solution continues to violate the constraints of the original problem, a new cutting plane would be derived. This procedure would then be repeated until the original problem was solved. Cutting planes for discrete optimization problems were introduced in the 1960's by Gomory (1960), and several other cutting plane schemes were developed in the same era. Recently, several stronger types of cuts have been developed based upon the concepts of disjunctive programming (see Balas (1974), Balas and Jeroslow (1975) and Sherali and Shetty (1980)) and the Reformulation-Linearization Technique (see Sherali and Adams (1990, 1994, 1999) and Sherali *et al.* (1998)). Although cutting planes are not always particularly effective in solving a problem to optimality in and of themselves, they have recently experienced a resurgence in attention due to their effectiveness when implemented within an enumerative framework such as branch-and-bound.

2.2.2 Enumerative Methods

Enumerative methods, such as *branch-and-bound*, successively partition the solution space of the original problem into smaller and smaller regions in the search for a globally optimal solution. The concept of branch-and-bound was developed by Land and Doig (1960) and further refined by Dakin (1965), and it remains one of the most widely used techniques in nonconvex optimization. This type of search is typically characterized by an enumeration tree, beginning with a root node that represents some base-level relaxation of the original problem. The root node is further partitioned into successor or children nodes that represent more restricted problems, with each branch of the tree detailing some set of additional restrictions on the variables. The specific restrictions at any particular node are all those listed on the branches along the path to the root node. At each node, the relaxation is solved to obtain a best-case bound, and a feasible solution to the original problem is sought in order to determine a worst-case bound. The goal is to shrink the gap between these two bounds in order to find an exact solution to the problem associated with each node. If these two bounds are different, the current node is partitioned into two (or more) new nodes that are more restricted. Whenever a node is infeasible or its best-case bound is worse than some previously obtained worst-case solution, or if the node subproblem is solved to optimality, we remove the node from any further consideration, or *fathom* the node. Throughout the process, we track the best known (or *incumbent*) solution, and the search ends when the bounds indicate that the incumbent solution

cannot be improved upon, and thus that all active nodes have been fathomed.

Most implementations of branch-and-bound solve some linear programming outer approximation or relaxation to obtain the best-case bound at each node, and a worst-case bound is found by performing a local search in the neighborhood of the relaxation solution. (If the solution to the relaxed problem is feasible to the original problem at that node, then this local search would be unnecessary as an optimal solution would already be at hand.) Since branching occurs whenever the bounds for a node are not sufficiently close to each other, the number of branches can become quite large unless good bounding strategies are employed. Thus, in order to prevent the enumeration tree from becoming prohibitively large, it is essential to have linear programming relaxations that closely approximate the original problem. Cutting planes are often employed within the branch-and-bound framework in order to tighten the bounds obtained at each node. While the cutting planes derived at a particular node are inherently valid for any subsequent node on the same branch, there has recently been increased attention on making these cuts globally valid for any node in the branch-and-bound tree. Such a solution technique, introduced by Padberg and Rinaldi (1987), is known as *branch-and-cut*, and is one of the most popular solution techniques in practice today. In the approach that we develop in Chapter 4, we will discuss one such technique for deriving globally valid cutting planes based upon the solution of a particular subproblem.

2.2.3 Benders' Decomposition

Benders' decomposition has proven to be a powerful technique for solving large-scale linear (and integer) programs since its introduction in 1962. The main idea behind this approach is to group the variables in such a way as to partition the problem into components that are easier to solve. This is accomplished by transforming the problem to create inner and outer optimization problems. The outer optimization, or *master problem*, captures the implicit projection of the original problem onto the space of the "complicating variables" via a set of Benders' constraints or cuts. Only a subset of these constraints is maintained at any stage to produce a *relaxed master problem*, and violated members of this set of cuts are sequentially generated as needed by solving the inner optimization problems, or *subproblems*. By fixing the complicating variables at values determined by the relaxed master problem, the subproblems can be solved with relative ease. The solution procedure, in essence, determines an optimal solution for the current relaxed master problem and solves a subproblem to determine whether or not the prospective solution violates any of the omitted constraints. (As detailed in Chapter 4, practical implementations, however, do not require the master program to be solved to optimality at each stage, and are designed to generate Benders' cuts in the spirit of applying a branch-and-cut procedure on the master program.) The subproblems can provide two types of cuts for the master problem, feasibility and optimality cuts. *Feasibility cuts* are used to eliminate any master program decisions that can produce infeasible inner optimization subproblems, while *optimality cuts* are used to approximate the inner optimization problem's objective value function. Both of these types of cuts can be generated using the dual solution to the subproblem. If no violated constraints are found, then the aforementioned prospective solution is determined to be an optimal solution to the original problem. Otherwise, a (most) violated Benders' cut is generated and this process is reiterated. More details on the Benders' partitioning strategy can be found in several linear or integer programming books (see, for example, Bazaraa *et al*. (1993) or Parker

and Rardin (1988)). In Section 2.3.1, we show the details of how Benders' partitioning can be used to solve stochastic programs, and in Chapter 4 we will present a specific application of this technique for discrete problems.

2.2.4 Disjunctive Programming

Many modern techniques for nonconvex optimization are based on generating polyhedral approximations for the original problem, and the theory and algorithms supporting these approaches can often be viewed in the context of disjunctive programming. Disjunctive programming problems are optimization problems in which the constraints of the problem are given as logical functions of two or more clauses, and typically these clauses are assumed to be linear with respect to the problem variables. Sherali and Shetty (1980) provide a thorough overview of disjunctive programming, as does Balas (1974, 1998), demonstrating how this field subsumes several classes of nonconvex optimization problems. In order to express the logical relationships between two clauses *A* and *B*, the following operations have been defined. A *conjunction*, denoted by $A \wedge B$, indicates that both clauses *A* and *B* must be true, while a *disjunction*, denoted $A \vee B$, indicates that either of the two clauses (or both) must be true. In many cases, the constraints of a disjunctive program restrict solutions to satisfy at least one of the relationships $A^h x \geq b^h, x \geq 0$ for some $h \in H$, where *H* is an index set over the family of restrictions. In this case, the feasible region of the disjunctive program, *F*, can be stated as a union of sets by $F = \bigcup_{h \in H} \{x : A^h x \geq b^h, x \geq 0\}$. The *disjunctive cut principle* facilitates linear constraints to be developed to enforce such logical restrictions on the variables. The forward part of the disjunctive cut principle, due to Balas (1974, 1975), associates a set of nonnegative multipliers, λ^h, with each set of constraints, $A^h x \geq b^h$, and surrogates this set of constraints into the single inequality, $(\lambda^h)^T A^h x \geq (\lambda^h)^T b^h$, for each $h \in H$. This set of surrogate constraints is then reduced to a single constraint by taking the pointwise supremum of the left-hand side and the infimum of the right-hand side of these constraints to yield $\left[\sup_{h \in H}[(\lambda^h)^T A^h]\right] x \geq \inf_{h \in H}[(\lambda^h)^T b^h]$ as a valid inequality for *F*. Jeroslow (1977) showed that the converse of this statement, known as the reverse part of the disjunctive cut principle, is also true. (Glover (1974) also provided similar results in a different context.) This indicates that any valid inequality for *F* can be uniformly dominated by a disjunctive cut of the above form, implying that the convex hull of *F* can theoretically be obtained by selecting appropriate λ-values.

Much attention has been given to the class of disjunctive programs known as *facial disjunctive programs* (FDP), which are detailed clearly and concisely in Sherali (1999). A facial disjunctive program is generally represented as: minimize $\{ c^T x : x \in X \cap Y \}$, where *X* is a nonempty polytope, and *Y* is given in *conjunctive normal form*, i.e., as a conjunction of disjunctions. More precisely, we have $Y = \bigcap_{h \in H} \left[\bigcup_{i \in Q_h} \{x : a_i^h x \geq b_i^h\} \right]$, where for each $h \in H \equiv \{1, \ldots, \hat{h}\}$, the corresponding disjunction requires that at least one of the inequalities $a_i^h x \geq b_i^h$ be satisfied for some $i \in Q_h$. The term *facial* implies that for each $i \in Q_h, h \in H$, $X \cap \{x : a_i^h x \geq b_i^h\}$ defines a face of *X*. The class of 0-1 mixed-integer programming problems,

for instance, can be viewed as a facial disjunctive program by taking *X* as the linear programming relaxation of the original problem, *H* as the index set of the binary variables, and $Y = \bigcap_{h \in H} \left[(x_h \le 0) \vee (x_h \ge 1) \right]$. As shown by Balas (1998), it is possible to construct the convex hull of feasible solutions for FDPs in an iterative fashion through a hierarchy of tighter relaxations $K_0, K_1, \ldots, K_{\hat{h}}$, starting with K_0 as the linear programming relaxation, $K_0 = X$. At each step of the process, Balas has shown how to inductively determine $K_h = \text{conv}\left[\bigcup_{i \in Q_h} (K_{h-1} \cap \{x : a_i^h x \ge b_i^h\}) \right]$ for $h = 1, \ldots, \hat{h}$, with $K_{\hat{h}} = \text{conv}(X \cap Y)$. Recent work in disjunctive programming has focused on generating deep disjunctive cuts within a branch-and-cut framework. (Earlier ideas in this vein were proposed by Sherali and Shetty (1980).) Toward this end, Balas *et al*. (1993) have developed *lift-and-project cuts* for 0-1 mixed integer programs by taking $K_h = \text{conv}\left[(K_{h-1} \cap \{x : x_h \le 0\}) \cup (K_{h-1} \cap \{x : x_h \ge 1\}) \right]$. In this process, each constraint in K_{h-1} is multiplied by the factors x_h and $(1 - x_h)$, and the resulting problem is linearized by replacing each product of variables as a single variable. As shown in the next section, however, this process can also be viewed as a direct application of the Reformulation-Linearization Technique (1990).

2.2.5 Reformulation-Linearization Technique (RLT)

The Reformulation-Linearization Technique (RLT) of Sherali and Adams (1990, 1994, 1999), provides a unifying approach for discrete and continuous nonconvex optimization problems. This approach can be used to generate a hierarchy of relaxations for nonconvex optimization problems that can lead to the convex hull of feasible solutions. The initial purpose of RLT was to address the class of 0-1 (mixed) integer linear and polynomial programming problems (Sherali and Adams (1990)), but it has since been extended to continuous nonconvex programming problems (Sherali and Tuncbilek (1992)). The main construct of RLT begins by multiplying problem constraints by a group of factors, known to be nonnegative for any feasible solution, where each factor is defined in terms of the original problem variables. Following this step, product terms of variables are replaced by a set of new variables in order to re-linearize the problem. This yields a relaxation derived in a higher dimensional space. The most basic factors used in the multiplication process, known as *bound-factors*, are based upon the premise that for any feasible solution x_j for $j \in N \equiv \{1, \ldots, n\}$, we have $(x_j - l_j) \ge 0$ and $(u_j - x_j) \ge 0$, where l_j and u_j are, respectively, the given (or implied) lower and upper bounds for the variable x_j. These individual terms can then be used to construct nonnegative bound-factors of the form $\prod_{j \in J_1} (x_j - l_j) \prod_{j \in J_2} (u_j - x_j)$, where J_1 and J_2 are appropriate index sets. In the case of continuous nonconvex optimization problems, each of the variables are used within these bound-factor products, and moreover, indices might repeat within the sets J_1 and J_2. However, in the context of 0-1 mixed-integer optimization problems, only the integer-restricted variables need be considered, and J_1 and J_2 are subsets of *N* with $J_1 \cap J_2 = \varnothing$. Specifically, focusing on 0-1 problems, for any level *d* in the hierarchy of relaxations produced by RLT, $0 \le d \le n$, the

bound-factors of order *d* are given as $F_d(J_1, J_2) = \left[\prod_{j \in J_1}(x_j - l_j)\right]\left[\prod_{j \in J_2}(u_j - x_j)\right]$, for all $J_1, J_2 \subseteq N$ such that $J_1 \cap J_2 = \varnothing$ and $|J_1 \cup J_2| = d$.

The level-*d* relaxation of the original problem is obtained by multiplying each of the constraints by every bound-factor of order *d*. The process for constructing a level-*d* RLT relaxation is comprised of two basic steps, a reformulation step and a linearization step, as summarized below.

> **Step 1 (Reformulation Step):** Multiply each inequality in the original problem (including upper and lower bounds on the variables) by each factor $F_d(J_1, J_2)$. In the case of 0-1 integer programming, we may tighten the formulation by noting that $x_j^2 \equiv x_j$ and $x_j(1 - x_j) \equiv 0$ for any binary x_j.
>
> **Step 2 (Linearization Step):** Linearize the resulting formulation in a higher dimensional space by defining a new variable to replace each distinct term that represents the product of original variables.

For the case of 0-1 mixed integer programming problems, common notation gives the binary variables x_j for $j = \{1, \ldots, n\}$, while the continuous variables are represented by y_k for $k = \{1, \ldots, m\}$. Given this notation, we linearize the resulting product terms using the variable substitutions $w_J \equiv \prod_{j \in J} x_j$ and $v_{Jk} \equiv y_k \prod_{j \in J} x_j$ for $k = \{1, \ldots, m\}$. In the case of continuous problems, however, all of the variables are assumed to be given as x_j for $j = \{1, \ldots, n\}$, and each of these variables is used to create the bound-factors. In this case, the typical substitution is given as $X_J \equiv \prod_{j \in J} x_j$, where *J* might have replicated indices from *N*.

In the case of 0-1 (mixed) integer programming problems with binary variables, Sherali and Adams have shown that a hierarchy of RLT relaxations can be obtained as *d* varies from 0 to *n*, starting with the linear programming relaxation with $d = 0$, and ending with the convex hull of feasible solutions for $d = n$. We note, however, that the process at level *n* involves multiplying each constraint by 2^n such bound-factors, which increases the size of the problem exponentially. The RLT process is by no means limited to the realm of 0-1 discrete optimization problems. As detailed in Sherali and Adams (1996, 1999), the RLT process has also been used to solve general integer programs, continuous polynomial programming problems, 0-1 quadratic programs, continuous and discrete bilinear programming problems, and indefinite quadratic programs. In all of these cases, the same conceptual two-step process is applied with appropriate specializations, and tighter formulations can be derived by applying higher levels of RLT.

Due to the tremendous growth in problem size at higher levels of RLT, levels greater than one or two are rarely used in practice. Sherali and Adams, however, have shown that lower level RLT applications, even the most basic level-one application, have proven very effective in

tightening problem relaxations for many classes of problems. In addition, several other strategies have been developed in order to tighten the lower level RLT relaxations, including the use of projected implications from higher level applications. One such tactic at level-one itself is to also multiply the original constraints by *constraint factors*, $(\alpha x - \beta) \geq 0$, for any structural inequalities $\alpha x \geq \beta$ that are implied by the original constraints, and then to apply the traditional linearization strategies. This can be particularly effective with constraints containing special structures. In addition, the concepts of RLT can be used to generate cutting planes to sequentially create tight representations of the problem in the vicinity of optimal solutions, rather than to develop *a priori* a tight representation of the entire feasible region during the modeling phase itself. A new tightening strategy, based upon incorporating concepts from semidefinite programming into RLT, will be presented in Chapter 3 of this dissertation.

2.2.6 Semidefinite Programming (SDP)

Semidefinite programming (SDP) offers a related relaxation strategy to RLT for solving certain types of nonconvex programming problems. Semidefinite programs are similar to LPs, except that the vector of variables is replaced by a matrix of appropriate variables, a special product operation is defined in lieu of the usual matrix-vector operations, and the matrix of variables is restricted to be positive semidefinite (**PSD**), in contrast with the nonnegativity constraints on the variables in linear programming. SDP has been receiving increased attention from the mathematical programming community since its inception over the past 5-10 years. Part of the reason for its popularity, as pointed out by Vandenberghe and Boyd (1996), is that SDP unifies several areas of mathematical programming (including linear and quadratic programming) from a theoretical point of view. Active set methods (similar to the simplex method in LP) were originally employed to solve SDP problems, but more recently, as shown by Alizadeh (1995), many interior point methods for solving linear programs can be directly modified and used to solve semidefinite programs in polynomial time. For a detailed overview of SDP, see Vandenberghe and Boyd (1996) or Alizadeh (1995). For articles that address theoretical results as well as various specific applications pertaining to SDP, see also: Wolkowicz *et al*. (2000), Todd (1998), Bertsimas and Zhang (1998), Ramana and Pardalos (1996), Ramana and Goldman (1995), and Goemans and Williamson (1995).

In general, semidefinite programming is the minimization of a linear function of symmetric matrices, subject to the constraint that an affine combination of these matrices is positive semidefinite. Recall that the following are equivalent for a symmetric n x n matrix U:

1. U is positive semidefinite (PSD), denoted as $U \succeq 0$.
2. $z^T U z \geq 0$ for all nonzero $z \in R^n$.
3. All eigenvalues $\lambda_j(U)$, $j = 1, \ldots, n$, of U are nonnegative.

The definition for a positive definite (PD) matrix is the same, but with strict inequalities. A common form of an SDP is given by:

$$\begin{array}{lll} \textbf{SDP}: & \text{minimize} & C \bullet X \\ & \text{subject to} & A_i \bullet X = b_i, \quad i = 1, \ldots, m \end{array}$$

$$X \succeq 0,$$

where $X, C, A_1, \ldots, A_m \in R^{nxn}$ and $b \in R^m$. The dot product of matrices *A* and *B*, denoted as $A \bullet B$, is defined as the trace of the matrix $A^T B$. That is, $A \bullet B = \sum_i \sum_j A_{ij} B_{ij}$. SDP is nonlinear and nonsmooth, but it is a convex optimization problem (see Vandenberghe and Boyd (1996) for a proof). Semidefinite programming shares the concepts of duality and complementary slackness, as well as some well-known theorems in linear programming such as weak duality. For a review of these theorems, see Vandenberghe and Boyd (1996). A semidefinite program can also be represented as a semi-infinite linear program, which is defined as a linear program having a finite number of variables and an infinite number of constraints. This is clear from the definition of PSD, since $X \succeq 0$ implies that $z^T Xz \geq 0$ for all $z \in R^n$.

Semidefinite programming is often used to obtain lower bounds for nonconvex optimization problems. A common strategy for developing an SDP relaxation commences by modifying the problem (if necessary) to create constraints containing the term xx^{T}. The substitution $X = xx^{\mathrm{T}}$ is next used, noting that *X* is PSD and rank-one by construction. In order to relax the problem, the constraint $X = xx^{\mathrm{T}}$ is then replaced by $X \succeq 0$, or more strongly by $X \succeq xx^T$. Note that the latter constraint may be expressed as $\begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0$.

Interior point methods are usually used to solve semidefinite programs. Alizadeh (1995) has shown that although several variations of these methods have been proposed, they have a similar structure, the same worst-case behavior, and similar performance in practice. The solution procedures generally solve one or two least-squares problems to determine a primal and dual search direction, as well as to compute a suitable step length. These two calculations comprise the majority of computation time per iteration. Vandenberghe and Boyd (1996) state that, in theory, the number of iterations required to solve an SDP to a specified accuracy grows no faster than the square root of the problem size. In practice, however, the algorithms converge much faster. In most cases, the number of iterations required is about 5-50, with almost no regard to problem size.

Semidefinite programming has been used, for example, to provide relaxations for the max-cut problem, in which the task is to select a maximally weighted set of arcs that separate the nodes of a graph into two disjoint sets. After some manipulation, this problem can be formulated as the optimization of a quadratic function over a hypercube, which happens to be NP-hard. This problem may be stated as follows:

MC:	maximize	$x^T Lx$
	subject to	$x \in \{-1,1\}^n$.

Observe that the constraint simply requires x_i^2 to equal 1 for all *i*. To obtain an SDP relaxation of this problem, we can define $X = xx^{\mathrm{T}}$, constrain each diagonal element to equal one, and relax *X*

to be any PSD matrix. This gives the following, noting that $x^T Lx \equiv L \bullet (xx^T)$.

$$\begin{array}{lll} \textbf{SDP(MC)}: & \text{maximize} & L \bullet X \\ & \text{subject to} & \text{diag}(X) = e \\ & & X \succeq 0 . \end{array}$$

Upon solving SDP(MC), the resultant matrix *X* must be transformed back into the vector *x* to derive a solution for the original problem. Goemans and Williamson (1995) have developed a "*randomized algorithm*" for this procedure, and they have proven that their solution has an expected maximum error of 13.8%.

Although the bounds provided by Goemans and Williamson are promising, it is shown that a branch-and-bound routine using only SDP relaxations cannot solve large problems. For this reason, Helmberg and Rendl (1998) have developed a solution procedure, combining SDP relaxation with cutting planes, that is both fast and robust. After an exact solution to the SDP relaxation has been found, Helmberg and Rendl transform *X* into the vector *x* by rounding each row of *X* to a {-1,1} vector. Next they vary the signs of the elements until there is no improvement in the objective function. The best of these rows is taken as the max-cut, and in many cases, this rounding procedure produces an optimal cut. If it does not, they use several criteria to try to generate an inequality that is violated by the current solution, and they append this to the problem. Since adding constraints adds more dimensions (and more work) to finding the search direction, the authors recommend adding only several of the strongest inequalities to the problem, even though many violated constraints may be detected. After these constraints are added to the problem, the authors restart their primal-dual algorithm. In addition to adding inequalities after the solution of the SDP relaxation (called a large-add), they also append some constraints during the process of solving the current relaxation (small-add). The overall procedure constructs the SDP relaxation of the problem and iteratively performs a large-add followed by 10 small-adds, terminating when the gap between the upper bound and the best known solution falls to within a pre-specified range. The results of their computational experience in combining SDP with cutting planes in this fashion are promising, albeit at a high computational cost. Helmberg and Rendl noted that the first round of adding inequalities typically yielded significant improvements while the improvement from later iterations was less dramatic. The authors hence recommend using one phase of adding the inequalities (1 large-add, 10 small-adds) within a branch-and-bound framework. It is worth noting that in their computational experiments, the relatively small problems having fewer than 50 nodes typically required no branching, being solved at the root node itself.

Benson, Ye, and Zhang (1998) have also addressed quadratic optimization problems using SDP relaxations. They have applied a polynomial-time dual-scaling algorithm to an SDP relaxation and combined it with heuristic procedures to achieve results for test problems of dimension 800 to 10,000. The problem considered by Benson *et al*. is of the form:

$$\begin{array}{lll} \textbf{QP1}: & \text{minimize} & \hat{C} \bullet (vv^T) + \hat{c}^T x \\ & \text{subject to} & \hat{A}_i \bullet (vv^T) + \hat{a}_i{}^T x = b_i, \quad \forall i = 1, \ldots, m \\ & & x \geq 0 , \end{array}$$

where $\hat{C}$ and $\hat{A}_i$ are given symmetric matrices, $\hat{a}_i$ and $\hat{c}$ are given column vectors, and *v* and *x* are the unknown variable vectors. Several combinatorial and optimization problems, including graph partitioning problems and box-constrained quadratic problems, can be put into this general form. Typically $\hat{A}_i$ is a sparse matrix of rank one, $\hat{C}$ is sparse, and $\hat{a}_i$ is either null or equal to the i^{th} unit vector. The authors make the standard substitution, $X = vv^T$, and then relax *X* to be any positive semidefinite matrix. Their computational experience has shown that the parameter values that work well at one point may be very different from the ones that work well at another point. For this reason, their dual-scaling algorithm computes four dual step directions by using four different input parameter values. If none of the four directions yields an improving solution, the input parameter is reverted to a multiple of the value that was used at the previous iteration. Five types of problems have been solved using a software package (DSDP) that contains their dual-scaling algorithm along with the aforementioned randomized algorithm. Their solution method was the first study to solve SDP relaxations of combinatorial problems having over 1000 variables.

Kojima and Tuncel (1999) have used the SDP approach to provide successive convex relaxations for problems having nonconvex feasible regions. They have developed two methods, the Successive Semidefinite Relaxation (SSDP) Method and the Successive Semi-Infinite Linear Program (SSILP) Relaxation Method. The SSILP is similar to the Reformulation-Linearization Technique (RLT) for continuous polynomial programs as developed by Sherali and Tuncbilek (1995). Kojima and Tuncel focus on problems having a linear objective function maximized over a nonconvex region that is described by a finite number of quadratic inequalities. They develop a procedure known as discretization to approximate an infinite number of semi-infinite SDPs (or LPs) by a finite number of standard SDPs (LPs) using a finite number of linear inequalities. A second technique, known as localization, is used when only an upper bound is required on the objective value for a particular objective function. This effort concentrates on finding a convex hull representation only in a suitable local neighborhood. Kojima and Takeda (1999) have performed a complexity analysis for the convex relaxation scheme proposed by Kojima and Tuncel. They found that even though the successive relaxations involve a finite number of problems having a finite number of constraints, the problem size still grows rapidly when higher accuracy is required, making the solution procedure impractical. Takeda *et al*. (1999) further reduced the problem to obtain an implementation containing a reasonable number of constraints. Their research focuses on an implementation of the Discretized-Localized version of SSILP. Their computational experience (on six types of test problems) shows that this method provides better approximations as compared with algorithms that use a single application of semidefinite programming or semi-infinite linear programming relaxations.

More recently, there has been an impetus of research related to reformulating and solving SDPs as ordinary nonlinear programs. Vanderbei and Benson (2000) propose a smooth, convex, finite nonlinear programming representation of a given positive semidefinite constraint $X \succeq 0$, by noting that a symmetric matrix *X* is PSD if and only if it can be factored as $X = LDL^T$, where *L* is a unit lower triangular matrix, and *D* is a diagonal nonnegative matrix . Denoting $d_j(X),\ j = 1,\ldots,n$, as the diagonal elements of *D* for a given $n \times n$ symmetric matrix *X*,

Vanderbei and Benson show that each $d_j(X)$ is a concave function of the elements of *X*, and moreover, is twice continuously differentiable on the set of PSD matrices. Accordingly, they replace $X \succeq 0$ by the nonlinear, smooth constraints $d_j(X) \geq 0$ for $j = 1,\ldots,n$, and develop a specialized interior-point algorithm for solving the underlying semidefinite program. Burer and Monteiro (2000) consider linear semidefinite programs in the standard form to

$$\text{minimize } \{C \bullet X : A_i \bullet X = b_i \text{ for } i = 1,\ldots,m,\ X \succeq 0\},$$

where *C* and A_i, $i = 1,\ldots,m$ are symmetric $n \times n$ matrices. They show that this problem can be solved as a nonlinear program in which *X* is replaced by a low-rank factorization RR^T, where *R* is an $n \times r$ matrix, with *r* taken as $\lceil \sqrt{2m} \rceil$. An augmented Lagrangian approach is then proposed to solve this resulting problem, using a limited-memory BFGS scheme for the inner-loop minimization process. However, the authors note that several local minima might exist, and offer no theoretical proof of convergence, although encouraging empirical results are presented.

Shor (1998) develops an alternative nondifferentiable optimization approach to semidefinite programming based on incorporating the nonsmooth convex constraint that restricts the smallest eigenvalue of *X* to be nonnegative. Given a symmetric $n \times n$ matrix *X*, if we denote the *n* real eigenvalues of *X* arranged in nondecreasing order by $\lambda_j(X)$, $j = 1,\ldots,n$, then $X \succeq 0$ is equivalent to the condition that $\lambda_1(X) \geq 0$. Moreover, if we denote $\alpha^j \equiv \alpha^j(X)$, $j = 1,\ldots,n$, as the set of linearly independent normalized eigenvectors corresponding to $\lambda_j(X)$, $j = 1,\ldots,n$, then noting that $\lambda_j(X) = (\alpha^j)^T X \alpha^j \ \forall j = 1,\ldots,n$, we have that $X \succeq 0 \Leftrightarrow \lambda_j(X) \geq 0$ for $j = 1,\ldots,n \Leftrightarrow (\alpha^j)^T X \alpha^j \geq 0$ for $j = 1,\ldots,n$. It is interesting to note that as a function of symmetric matrices *X*, $\lambda_1(X)$ is a concave, but nondifferentiable, function (see Shor (1998), for example), although as demonstrated by Vanderbei and Benson (2000), the remaining eigenvalue functions $\lambda_j(X)$ for $j = 2,\ldots,n$, do not necessarily enjoy this concavity property. Furthermore, by the Raleigh-Ritz formula (which can be readily verified via the normalized eigen-basis diagonalization process), we have that

$$\lambda_1(X) = \underset{\|\alpha\|=1}{\text{minimum}}(\alpha^T X \alpha).$$

Observe that as a function of *X*, λ_1 is hereby characterized as the minimum of a family of linear functions, and is therefore concave with a set of subgradients that can be characterized in terms of the normed eigenvectors α^* associated with $\lambda_1(X)$, where $\lambda_1(X) = \alpha^{*T} X \alpha^*$ for each such α^*. Accordingly, Shor (1998) incorporates the nonsmooth convex constraint $\lambda_1(X) \geq 0$ in the model formulation, in lieu of $X \succeq 0$, and proposes a nondifferentiable optimization strategy.

2.3 Some Relevant Application Areas of Nonconvex Optimization

In Chapters 4 and 5 of this dissertation, we will focus on two particular types of nonconvex optimization problems, namely stochastic programming and facility location problems. We therefore review some of the relevant literature in these areas in the following two sections.

2.3.1 Stochastic Programming Problems

Stochastic programs are mathematical programs where some of the problem parameters are not known with certainty, but rather, their values are known to follow some probabilistic distributions. Dantzig and Beale independently proposed the basic concepts of stochastic programming in 1955, with Dantzig calling the area "Linear Programming Under Uncertainty" and Beale labeling it "Linear Programming with Random Coefficients." The application areas of stochastic programming can be as varied as those of linear programming, but applications in production, financial planning, airplane scheduling, power generation, and vehicle routing are among the most common. The literature on stochastic programs focuses largely on two-stage stochastic programs with recourse. In theory, multi-stage programs can be handled in a similar fashion via a nested approach, but in practice, this process is cumbersome to implement. In these problems, the first-stage decisions must be made before the relevant random components of the environment are realized, and then, a set of second-stage (or recourse) variables is used to compensate for the ensuing effect of the environment. In the context of production planning, for example, the first-stage variables might include the number of worker-hours required to meet customer demand, where the latter is not known with certainty at the time of scheduling. If the actual customer demand is not met exactly by the first-stage decision, recourse actions (such as using overtime, underutilizing the workforce, or laying off workers) may be used, but they generally incur a penalty cost. The goal of the stochastic program is to optimize the first-stage costs plus the expected recourse costs. Some notable applications of stochastic programming include scheduling (Birge and Dempster, 1996), financial planning (Carino *et al*., 1994), power generation (Murphy *et al*., 1982), facility location (Laporte *et al*., 1994), and vehicle routing (Laporte *et al*., 1992). For more information on stochastic programming in general, we refer the reader to recent books on stochastic programming by Ermoliev and Wets (1988), Kall and Wallace (1994), and Birge and Louveaux (1997).

There are several popular methods for solving two-stage stochastic LPs with recourse, and most of these rely on the underlying principle of Benders' decomposition. The inherent structure of these problems lends itself to a natural partitioning of the variables. The first-stage investment, resource acquisition or location-type decisions, represent the complicating variables, while the subproblems determine the best recourse actions for each realization of the environment, given any first-stage decisions. A common practice is to approximate continuous distributions with discrete ones, which allows the expected recourse function to be calculated as a simple weighted sum. In the case of stochastic programs with integer recourse, Schultz (1995) has shown that, under mild conditions, discrete distributions can effectively approximate continuous ones to any given accuracy. Consequently, assume that there are L possible environments, $\tilde{\xi}^l$, $l = 1,...,L$, each occurring with a respective probability of p_l. The set of

constraints that couples the first- and second-stage decisions, $x \in R^n$ and $y \in R^m$, respectively, is generally expressed as

$$W^l y^l = h^l - T^l x,$$

where the (technology) matrix T^l and the (resource) vector h^l are known for each possible environment $\tilde{\xi}^l$, $l = 1,...,L$. The matrix W^l (which is often assumed to be fixed in order to yield an exploitable subproblem structure, but in general, could be stochastic as well) is known as the recourse matrix, and it determines the set of recourse actions, y^l, that are governed by the net outcome $h^l - T^l x$. Given this notation, a typical Benders' decomposition for the two-stage stochastic program with recourse would view the given problem in the form

$$\begin{aligned} \textbf{SP:} \quad & \text{minimize} \quad cx + \sum_{l=1}^{L} p_l Q(x, \xi^l) \\ & \text{subject to} \quad x \in X, \end{aligned}$$

$$\text{where} \quad Q(x,\xi^l) = \min \{q^l y^l : W^l y^l = h^l - T^l x,\ y^l \geq 0\} \ \text{ for } l = 1,\ldots,L,$$

and where X is some nonempty polytope in R^n, with approximations for the optimal value functions $Q(x,\xi^l), l = 1,\ldots,L$ being generated via Benders' cuts. The term *fixed recourse* refers to the situation where the recourse matrix is non-stochastic, that is $W^l = W$, $\forall l = 1,\ldots L$. In the special case of *complete recourse*, we have that the recourse problem remains feasible for *any* given realization of the first-stage variables. The weaker assumption of *relatively complete recourse* implies that for every *feasible* first-stage decision, i.e. $\{x \mid x \in X\}$, the recourse problems $Wy^l = h^l - T^l x, y^l \geq 0$ are feasible for all $l = 1,...,L$. In practice, it is difficult to recognize *a priori* whether or not a particular problem has relatively complete recourse. The most basic type of stochastic programs possess *simple recourse*, in which the recourse variables directly equal the net outcome $h^l - T^l x$. In other words, we have $W^l = [I, -I]$, and the constraints of each recourse problem simplify to $y^{l^+} - y^{l^-} = h^l - T^l x$. Clearly, simple recourse problems also exhibit complete recourse. In the cases of complete or relatively complete recourse, the subproblems encountered in the Benders' partitioning strategy are all feasible, and in such cases, only optimality cuts are generated.

We note that stochastic programs with recourse can also be modeled as large-scale linear programs, assuming that the random outcomes follow a discrete distribution. The LP equivalent of SP is given as:

$$\begin{aligned} \textbf{SLP:} \quad & \text{minimize} \quad & & c^T x + p_1 (q^1)^T y^1 + p_2 (q^2)^T y^2 + \quad \ldots \quad + p_L (q^L)^T y^L & \\ & \text{subject to} \quad & & Ax & = b \\ & & & T^1 x + \quad Wy^1 & = h^1 \\ & & & T^2 x \quad + Wy^2 & = h^2 \end{aligned}$$

$$\begin{array}{lll} & \ddots & \\ T^L x & & +Wy^L = h^L \\ & & x, y^l \geq 0. \end{array}$$

These two formulations are equivalent in the sense that they have the same set of solutions over x, and the optimal values of $y^l, l = 1,...,L$ for the SLP are the solutions to the second stage problem of P, given an optimal set of first-stage decisions x. Note that when $T^l = T \ \forall l$ (i.e. T is non-stochastic), the structure of SLP simplifies significantly to the staircase structure shown below.

$$\begin{array}{cccccccc} c & p_1q^1 & p_2q^2 & \cdots & & p_Lq^L & & \\ A & & & & & & = & b \\ T & W & & & & & = & \hat{h}^1 \\ & -W & W & & & & = & \hat{h}^2 \\ & & -W & W & & & = & \hat{h}^3 \\ & & & & \cdots & & & \\ & & & & -W & W & = & \hat{h}^L \end{array}$$

When this type of problem structure exists, special solution techniques can be used to take advantage of it. Similarly, the dual structure provides an alternative method for solving the LP equivalent of SP. Consider the dual of SLP as:

$$\begin{aligned} \textbf{SLD':} \ \text{maximize} \quad & b^T\sigma + \sum_{l=1}^{L} (h^l)^T \hat{\pi}^l \\ \text{subject to} \quad & A^T\sigma + \sum_{l=1}^{L} (T^l)^T \hat{\pi}^l \leq c \\ & W^T \hat{\pi}^l \leq p_l q^l, \qquad l = 1,...,L. \end{aligned}$$

If we let $\pi^l = \hat{\pi}^l / p_l$, we arrive at the following equivalent formulation:

$$\begin{aligned} \textbf{SLD:} \ \text{maximize} \quad & b^T\sigma + \sum_{l=1}^{L} p_l (h^l)^T \pi^l \\ \text{subject to} \quad & A^T\sigma + \sum_{l=1}^{L} p_l (T^l)^T \pi^l \leq c \\ & W^T \pi^l \leq q^l, \qquad l = 1,...,L. \end{aligned}$$

The matrix structure of SLD displayed below can be exploited to generate efficient solution techniques for SLD.

$$
\begin{array}{lllll}
b^T & p_1h^1 & p_2h^2 & \ldots & p_Lh^L \\
A^T & p_1(T^1)^T & p_2(T^2)^T & & p_L(T^L)^T \\
 & W^T & & & \\
 & & W^T & & \\
 & & & W^T & \\
 & & & & \\
 & & & \ldots & \\
 & & & & \\
 & & & & W^T
\end{array}
\qquad
\begin{array}{ll}
 & \\
\leq & c \\
\leq & q^1 \\
\leq & q^2 \\
\leq & q^3 \\
 & \\
 & \\
 & \\
\leq & q^L
\end{array}
$$

In particular, when the recourse problem contains more variables than constraints (which is usually the case) SLD has fewer (unconstrained) variables but a large number of constraints.

The majority of stochastic programming algorithms, however, focus on solving problem SP using decomposition techniques. Most of these methods can be considered as extensions of the L-shaped algorithm that was proposed by Van Slyke and Wets (1969). The L-shaped algorithm is a cutting plane algorithm that uses Benders' decomposition to create an outer linearization of the objective function. The algorithm iterates between a master problem and a series of subproblems. The master problem is shown below.

$$
\begin{array}{lll}
\textbf{MP:} & \text{Minimize} & c^T x + \theta \\
 & \text{subject to} & Ax = b \\
 & & \theta - f(x) \geq 0 \\
 & & x \geq 0,
\end{array}
$$

$$\text{where} \qquad f(x) = E[\min\{q^T y \,|\, Wy = h - Tx, y \geq 0\}].$$

Since $f(x)$ is not known explicitly, it is typically approximated via a set of feasibility and optimality constraints or cuts. This produces the relaxed master problem shown below.

$$
\begin{array}{lll}
\textbf{RMP:} & \text{Minimize} & c^T x + \theta \\
 & \text{subject to} & Ax = b \\
 & & D_k x \geq d_k, \quad k = 1,\ldots,r \\
 & & E_k x + \theta \geq e_k, \quad k = 1,\ldots,s \\
 & & x \geq 0, \ \theta \text{ unrestricted.}
\end{array}
$$

At iteration *v*, we are given a solution, (x^v, θ^v), to the relaxed master problem. We first determine if x^v admits a feasible solution to the recourse problem. To do this, we solve a Phase I problem for the recourse problems. If the optimal solution value for this problem is positive, x^v does not yield a feasible solution to the recourse problem, and so we add to the master problem a feasibility cut that constrains the equivalent dual solution to be non-positive. If x^v is feasible to the recourse problem, we then compare the optimal recourse objective value to the bound θ^v, in

essence verifying whether $f(x^v) \le \theta^v$. If not, we add an optimality cut to the master problem, forcing $f(x^v) = (h - Tx^v)^T \sigma^v \le \theta$. Recall that from duality theory,

$$f(x^v) = \min\{q^T y \,|\, Wy = h - Tx^v, y \ge 0\} = \max\{(h - Tx^v)^T \sigma^v \,|\, W^T \sigma^v \le q\}.$$

Since only a finite number of these constraints exist based on extremal solutions, the overall algorithm converges finitely. In summary, at each iteration of the L-Shaped Algorithm, we solve the relaxed master problem followed by one subproblem for each of the *L* outcomes. If any of the subproblems are infeasible, a feasibility cut is added to the master problem. Otherwise, the optimal dual multipliers for the set of subproblems are used to create a single optimality cut for the master problem. If the cost coefficients of the recourse problem are deterministic and only the right-hand side values are stochastic, we solve *L* linear programs that differ only in their right-hand side values:

$$\begin{aligned} \text{minimize} \quad & w^l = q^T y \\ \text{subject to} \quad & Wy = t^l \\ & y \ge 0, \end{aligned}$$

where $t^l = h^l - T^l x$. In such cases, we can use special techniques such as sifting (discrete parametric analysis) and bunching (basis by basis analysis).

Birge and Louveaux (1988) developed a multicut enhancement to the L-Shaped Algorithm, in which a separate optimality cut is constructed for each subproblem. While the L-shaped method sends a single constraint to the relaxed master problem as an outer linearization of the expected recourse costs, the multicut algorithm sends an outer linearization of the recourse cost for each subproblem. Note that using multiple cuts corresponds to including several columns in a dual procedure (Dantzig-Wolfe decomposition) instead of one aggregate column. The intention is to send more information to the relaxed master problem than a single cut, and in so doing, reduce the number of major iterations, and therefore increase convergence speed of the algorithm. Birge and Louveaux have also developed a simplification for simple recourse problems, stemming from the fact that only two types of optimality cuts can be generated for these problems.

The major limitation of the L-shaped and multicut algorithms is that they require the solution of *L* linear programs at each iteration. An alternative approach to the decomposition-based strategies is to use modified convex optimization techniques such as stochastic quasigradient (SQG) methods. This strategy works with discrete and continuous distributions, and it generates one observation of the random variable at each iteration. The SQG techniques also have limitations, however. Their major drawbacks are the difficulty in determining step lengths and the lack of an estimate of the objective function during the iterative process. Higle and Sen have developed Stochastic Decomposition (1991) and Conditional Stochastic Decomposition (1994) to combine the best aspects of decomposition-based and stochastic approximation algorithms. (See Higle and Sen (1996) for a thorough review of both approaches.) These methods are similar to the L-shaped and multicut algorithms, except that at

each iteration, the subproblem is solved for *one* randomly generated sample point. There are no restrictions on the random variable distributions. The idea is to generate statistically-based approximations for the feasibility and optimality cuts. At later iterations, when more observations of the random variable are available, the previous cuts are updated to reflect the most accurate information. Although Higle and Sen have shown that these methods contain a sequence of iterates that converge to optimality with probability one, practical implementations track the best incumbent solution since the convergent subsequence is difficult to track. For a thorough summary of current decomposition methods for stochastic programs, including some recent advances, see Ruszczynski (1999).

Stochastic integer programs are stochastic programs in which some of the variables are restricted to be integer-valued. The integrality restriction can apply to the first- and/or second-stage variables. When the second-stage (recourse) variables are restricted to be integral, the resulting problem is referred to as a *stochastic program with integer recourse*. In this case, the problem complexity increases significantly, since the subproblem for any random outcome is an integer program whose parameters depend on the first-stage decisions. Moreover, the optimal value recourse objective function now becomes nonconvex and discontinuous in general.

Although some solution strategies have been developed for specific applications of stochastic IPs, relatively few techniques have been developed to solve general stochastic IPs. We comment here on some recent algorithmic advances that employ decomposition techniques. (For a thorough review of recent advances in developing models and algorithms for stochastic integer programming, see Klein Haneveld and van der Vlerk (1999) and Schultz *et al*. (1996).) Laporte and Louveaux (1993) developed the integer L-shaped algorithm (a combination of the L-shaped method and branch-and-bound) to solve stochastic IPs with binary first-stage variables and complete (mixed-integer) recourse. This extension constructs optimality cuts based on independent evaluations of the recourse value function. For efficiency in an enumerative search process, certain lower bounding functionals on this recourse value function are also derived. Caroe and Tind (1998) have used general duality theory to develop a more general extension of the L-shaped decomposition method to solve two-stage stochastic programs with integer recourse, and have shown the integer L-shaped method to be a special case of their more general framework. Previously, Caroe and Tind (1997) had developed a Lagrangian dual approach based on applying variable splitting to the first-stage decisions, and then dualizing the resultant equal-value nonanticipatory constraints. This approach was shown to be equivalent to computing a hull relaxation in the context of disjunctive programming, and was solved using the lift-and-project cutting plane technique of Balas *et al*. (1993). Cuts derived for one subproblem were lifted to derive valid inequalities for other subproblems. However, in order to preserve facetial properties in this lifting process, a separate linear program needed to be solved. We note here that in our approach (presented in Chapter 4), which is geared toward solving the original problem itself (rather than its relaxation), we show how cuts derived for one subproblem can be directly used for other subproblems without any intermediate lifting step or auxiliary problem solution (other than a simple substitution). Moreover, facetial properties are preserved in a manner that induces finite convergence.

For the specific case of simple integer recourse where $W^I = [I, -I]$, and with a fixed technology matrix and discretely distributed right-hand sides, Klein Haneveld *et al*. (1996) have

used theoretical properties of the recourse objective value function to derive a convex hull representation for the problem. They first show that the expected value function is separable and that the mass points of the discrete distribution dictate its structure. They then develop an algorithm based on the premise that the convex hull of the simple integer recourse objective is equal to that of the expected value of a continuous simple recourse problem (under a suitable transformation of variables) plus a constant. Based on this, they next apply a procedure to systematically smooth out "knots," or nondifferentiable points, from an underlying piecewise linear approximation, and use the corners of the resulting smoothed piecewise linear function as the mass points for the transformed variables. The constant term is then determined as a function of the transformed variables, and the overall algorithm is shown to be polynomially bounded in terms of the number of random outcomes. (See Klein Haneveld and ven der Vlerk (1999) for a summary of several other techniques for simple integer recourse problems.)

Caroe and Schultz (1999) have used scenario decomposition and Lagrangian relaxation within a branch-and-bound framework to solve two-stage stochastic IPs, and this approach can readily be extended to multistage stochastic programs. Ahmed *et al*. (2000) consider two-stage stochastic programs having pure integer second-stage variables, but mixed-integer first-stage variables. They employ a transformation that induces a special structure in the discontinuities of the second-stage optimal value function and based on a characterization of this structure, they design a finitely convergent branch-and-bound algorithm for the original problem. Promising computational results are provided on several classes of problems. A specialized approach for two-stage stochastic IPs with mixed-integer recourse that is similar to ours in concept, but uses an alternative sequential convexification process based on a different asymptotically exact cutting plane approach for solving the subproblems for fixed values of the first-stage decisions, has been proposed by Higle and Sen (2000). In a different vein, Schultz *et al*. (1998) have used Grobner basis techniques within an implicit enumeration strategy to address the class of problems having integer recourse. Although Grobner bases are typically expensive to compute, their use becomes relatively more effective when the same problem is re-solved for different right-hand side values, which is the case for recourse problems.

2.3.2 Facility Layout Problems

The facility layout problem is concerned with determining a non-overlapping layout of departments within a designated section of a building, while maintaining certain area restrictions for each department and minimizing the expected cost of flows, taken as the rectilinear distance times the number of trips, between the departments. The literature addresses problem instances that specify fixed dimensions for each of the departments while considering only their relative positions as decision variables, as well instances where both the location and dimensions of each department are variables to be optimized. Similarly, some instances assume fixed grid positions for the departments, while others allow more flexibility. While most applications assume that the flow of material occurs to and from the departmental centroids, some applications consider the placement of a specific input/output station within each department. In the most general sense, facility layout problems are composed of two types of constraints, as noted by Meller and Gau (1996). The first type restricts the area of the departments to be within some prescribed limits, while the second type provides restrictions on departmental locations, such as avoiding departmental overlaps, and requires the departments to remain within the limits of the facility,

and to avoid certain fixed areas of the building. For a detailed survey of recent advances in the facility layout problem, see Meller and Gau (1996).

Several sources in the literature (see, for example, Chittratanawat and Noble (1999) and Georgiadia *et al*. (1999)) have shown that the layout of a facility has a tremendous impact on its operating costs, and is therefore of critical importance. For this reason, the facility layout problem has received a great deal of attention in the operations research community. In the 1970s and 1980s, the most popular approaches to the facility layout problem were graph theoretical approaches. In these approaches, the relative desirability of locating each pair of departments adjacent to one another is specified. These relationships are used to construct an adjacency graph which, ignoring department sizes, specifies a general preference for which departments should be near one another. The dual of this graph is then constructed, and is used to generate a block layout for the facility. Typically, heuristic approaches are used to construct an adjacency graph that is maximally weighted, yet still limited enough to construct its dual with reasonable effort. The truly limiting factor, however, is translating the dual graph to a block layout that specifies each department's shape and size, a task that is typically done by hand.

While most of the research on the facility layout problem has focused on generating good layouts through construction and improvement heuristics, a new trend has also emerged. Within the past decade, several researchers have formulated the facility layout problem as an optimization problem. If we desire to locate equally sized departments within some predetermined grid, the facility layout problem reduces to the quadratic assignment problem, which is itself a very difficult problem for even a moderate number of departments. When adding the complications of unequal areas and varying horizontal and vertical dimensions, it is clear that the facility layout problem is highly challenging to solve to optimality. For this reason, several researchers have considered heuristic approaches for the underlying optimization problem.

In this respect, Montreuil *et al*. (1993) examine several design skeletons (flow graphs, adjacency graphs, cut trees, etc.) from which human designers have traditionally generated good facility layouts. Given such a design skeleton and its graphical representation, the authors solve a linear programming model to generate a layout. This approach can be integrated well with an interactive optimization-based design framework. Delmaire *et al*. (1997) have combined genetic algorithms with linear programming for the problem where all the departments must be located on either side of a main aisle. They use a genetic algorithm to generate the relative positioning of the departments, and then formulate and solve a linear programming model to determine the locations of the input/output stations and the dimensions of the departments, to minimize the cost of the layout. The method can also be extended to the case where the departments are located around a ring-shaped aisle. The results reported are promising, outperforming several available methods for the test cases solved. Chittratanawat and Noble (1999) have developed an integrated approach to address facility layout, including the determination of input/output stations and material handling equipment selection. Due to these added complications, the model requires equal department sizes that are known *a priori*. Their model is a nonlinear mixed-integer program, and is solved using tabu search metaheuristic schemes, including two heuristic procedures for solving the underlying subproblems. Other recent examples of combining submodels with heuristic optimization techniques include the approaches of Banerjee

et al. (1992), Heragu and Kusiak (1991), and Langevin *et al*. (1994).

There have been several attempts, however, to solve the facility layout problem through traditional modeling and optimization techniques, and this is the area that our research will focus on. The main difficulty in these models is finding good approximations for the nonlinear departmental area restrictions, and providing adequate constraints to prevent departments from overlapping. Montreuil (1990) has proposed one such model, a mixed-integer programming formulation called FLP1. This model includes four decision variables for each department *i*; namely, the half-length and half-width (ℓ_i^x, ℓ_i^y), and the centroidal location (c_i^x, c_i^y). For each department *i*, the required area a_i is specified, and a parameter α_i (≥ 1), known as the aspect ratio, is delineated to restrict the maximum permissible ratio between the longest and shortest sides of the department for aesthetic purposes. Using this information, Montreuil relaxes the nonlinear area constraint, $a_i = 4l_i^x l_i^y$, with bounded perimeter constraints, $p_i \leq 4(l_i^x + l_i^y) \leq P_i \ \forall i$, where $p_i = 4\sqrt{a_i}$ and $P_i = 2\sqrt{a_i}(1+\alpha_i)/\sqrt{\alpha_i}$. This formulation, however, is biased in favor of smaller departments and can lead to a significant under-representation of the area. In their model FLP2, Meller *et al*. (1999) develop improved area restriction representations, called surrogate area constraints, by requiring $4(\ell_i^x + \ell_i^y) \geq 3\sqrt{a_i} + f \times 2\ell_i^{\max} \quad \forall i$, where *f* is a parameter that is empirically determined to be 0.95. These constraints are also not very effective in enforcing the area requirements, as we shall exhibit later in Chapter 5, where we will describe a more analytical approach to deriving appropriate area representation approximations. We note that a more complete review of the model FLP2 is presented in Chapter 5, prior to the presentation of our proposed enhancements.

Chapter 3: Enhancing RLT Formulations through Connections with Semidefinite Programming

As discussed in Chapters 1 and 2, it is essential to have tight formulations for nonconvex optimization problems if we are to obtain good lower and upper bounds, and thereby solve the original problem with a reasonable amount of effort. For many classes of problems, lowest-level RLT relaxations have proven effective in deriving tight lower bounding mechanisms. However, this observation is not a uniform experience, and even in the aforementioned cases, the overall process can greatly benefit by incorporating suitable general classes of additional RLT inequalities that serve to further tighten the relaxation, without having to resort to higher-level representations. With this motivation, we explore the generation of particular types of valid inequalities or cutting planes that are in fact generalized RLT constraints derived via semidefinite programming concepts. We call this class of valid inequalities *semidefinite cuts*. For some other classes of effective RLT cuts developed for the special case of quadratic polynomial programs, we refer the reader to Audet *et al.* (2000).

The remainder of this chapter is organized as follows. After discussing the motivation for combining the SDP and RLT methods in Section 3.1, we introduce in Section 3.2 the Problem QP that is used to evaluate the proposed methodology, and discuss a typical SDP relaxation for Problem QP that would then be solved by specific SDP solvers. To illustrate our more general methodology, we present in Section 3.3 an alternative semidefinite relaxation for Problem QP that is more closely associated with the usual RLT process, and which in fact yields a tighter relaxation. This SDP relaxation is then shown to be equivalent to a suitable semi-infinite RLT relaxation. Based on this derivation, we develop a strategy that sequentially augments the first-level relaxation RLT-1(QP) with cutting planes that are automatically generated from the constraints in the semi-infinite representation using a special polynomial-time separation procedure. In Section 3.3.2, several cut generation mechanisms are explored in this context. Thereafter, in Section 3.3.3, we demonstrate that potentially stronger classes of such cutting planes can be generated in a likewise fashion with comparable effort by simply replacing the semidefinite constraint $X \succeq 0$ by the restriction $X \succeq xx^T$, i.e., $\begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0$. A summary of our experimental design is presented in Section 3.4, along with computational results for employing cutting planes based on both types of semidefinite constraints. Section 3.5 examines the extension of the proposed relaxation enhancement procedure to higher-level RLT representations. Finally, Section 3.6 presents conclusions and suggestions for future research.

3.1 Motivation

In this chapter, we integrate the concepts of semidefinite programming and RLT to develop a class of semidefinite cuts that can be used to augment the RLT relaxation for any problem (discrete or continuous, linear or nonlinear) to which the latter technique is applicable. Given an RLT relaxation for any such problem, we show that we can further enhance this relaxation by incorporating an infinite class of particular RLT constraints that are based on semidefinite relationships. Rather than solve the resulting semi-infinite program, which in itself would require a specialized solution approach, we adopt the strategy of generating suitable members from the infinite constraint set as needed through a cutting plane or separation procedure. This separation routine is executed in polynomial time, thereby making the cut generation process efficient. Moreover, each relaxation in this sequential process is a linear program whose solution can be updated using standard mathematical programming software. At termination, this procedure yields a lower bound on the optimal value of the original problem. In addition, an upper bound can be computed by initializing a local search procedure with the solution obtained for the final relaxation. These bounds can be embedded within a branch-and-bound framework to determine a global optimum to the original problem.

Note that this concept of generating cutting planes based on semidefinite restrictions can be used to augment *any* RLT relaxation, *even if* the overall relaxation cannot be cast as a semidefinite program, or if it contains sets of (nonlinear) convex constraints as in Sherali and Tuncbilek (1997). For example, Sherali and Wang (2001) have recently proposed a global optimization approach for solving general nonconvex factorable programs by integrating a polynomial approximation with an RLT scheme. In this context, our proposed approach can be applied identically by augmenting the simple nonnegativity and symmetry restrictions on the even-ordered RLT variables by a stronger positive semidefinite constraint, and then generating valid inequalities to tighten the relaxation in a manner similar to that exposed in the sequel.

3.2 Problem Class QP

As a point of *illustration* of this *general concept*, we will consider a specific example of the class of problems involving the minimization of a nonconvex quadratic objective function over a simplex (denoted **QP** below). This problem is interesting in its own right, and has been extensively studied by Nowak (1998a,b, 1999). It arises, for instance, in the context of finding a maximal weighted clique in an undirected graph.

$$\textbf{QP}: \quad \text{Minimize} \quad \sum_i \sum_j C_{ij} x_i x_j \tag{3.1a}$$

$$\text{subject to} \quad e^T x = 1 \tag{3.1b}$$

$$x \geq 0, \tag{3.1c}$$

where $x \in R^n$ and e is a vector of n ones. Although Problem QP is NP-Hard, it has a simple structure that makes it convenient to *illustrate the essence* of our approach, and extensions to more general problems are readily evident.

The first-level RLT relaxation RLT-1 (see Sherali and Tuncbilek, 1992) for Problem QP would multiply (3.1b) with each variable x_i, for $i = 1,\ldots,n$, and then substitute a nonnegative variable X_{ij} for each term $x_i x_j$ in the problem, where $X_{ij} \equiv X_{ji}\ \forall i, j = 1,\ldots,n$. To write this resulting problem in a specific manner that exposes connections with semidefinite programming and motivates our development, define $X \equiv [X_{ij}]$ to be an $n \times n$ (symmetric) matrix that represents the linearization of xx^T under the foregoing RLT substitution (i.e., $X \equiv [xx^T]_L$, where in general, $[\cdot]_L$ represents the standard linearization operation of RLT; in the present context, this involves the substitution of X_{ij} for the product term $x_i x_j$). Then, we can write the level-one RLT relaxation for QP in the form

$$\textbf{RLT-1(QP)}: \quad \text{minimize} \quad \sum_i \sum_j C_{ij} X_{ij} \tag{3.2a}$$

$$\text{subject to} \quad e^T x = 1 \tag{3.2b}$$

$$Xe = x \tag{3.2c}$$

$$x \geq 0,\ X \geq 0 \text{ and symmetric.} \tag{3.2d}$$

Nowak (1998a,b, 1999) has proposed various SDP approaches for solving Problem QP. To derive a suitable semidefinite relaxation for QP, Nowak first employs the particular RLT constructs of multiplying the constraints $x_i \geq 0$ and $x_j \geq 0$ pairwise and squaring the constraint $e^T x = 1$, to derive the following quadratically constrained quadratic program (QQP).

$$\textbf{QQP}: \quad \text{Minimize} \quad \sum_i \sum_j C_{ij} x_i x_j$$

$$\begin{aligned} \text{subject to} \quad (e^T x)^2 &= 1 \\ x_i x_j &\geq 0, \quad \forall\, 1 \leq i, j \leq n \\ x &\geq 0. \end{aligned}$$

By substituting $X = xx^T$, he then obtains a semidefinite relaxation for this representation as given by

$$\textbf{SDP(QQP)}: \quad \text{minimize} \quad C \bullet X \tag{3.3a}$$

$$\text{subject to} \quad (ee^T) \bullet X = 1 \tag{3.3b}$$

$$X \geq 0 \tag{3.3c}$$

$$X \succeq 0 \tag{3.3d}$$

where $C = [C_{ij}]$ and where for any conformable square matrices $A = [A_{ij}]$ and $B = [B_{ij}]$, the dot product $A \bullet B$ is defined as the trace of $A^T B$, i.e., $A \bullet B = \sum_i \sum_j A_{ij} B_{ij}$. Also, $X \succeq 0$ denotes that X is *symmetric and positive semidefinite*. Nowak next constructs a convex quadratic function, $w(x) = x^T W x$, such that $W \leq C$ and $w(x)$ approximates $C \bullet X = x^T C x$. The matrix W is

found by solving a separate semidefinite program. This produces an approximation for the convex envelope of the objective function, and the optimal solution to this convex program is used to provide an estimate for the global minimum of Problem QP. Nowak has developed several lower bounding schemes for Problem QP, each based upon solving a different SDP problem to find *W*.

3.3 Development of the SDP Cuts

There are several ways to construct a semidefinite relaxation for QP. One such formulation, suggested by Nowak, was presented above in (3.3). Alternatively, rather than squaring the simplex constraint, we can instead multiply it (on the right) by x^T as one would in an RLT approach, and use the substitution $X = xx^T$. Since *X* is symmetric, this yields the constraint $Xe = x$, which we append to the original problem. Relaxing $X = xx^T$ to $X \succeq 0$, we obtain the following semidefinite relaxation of QP. Note that from (3.4b,c), we have $(ee^T) \bullet X \equiv e^T Xe = e^T x = 1$, or that (3.3b) is implied. Hence, formulation (3.4) potentially yields a tighter relaxation of QP than that given by (3.3).

SDP(QP): Minimize $C \bullet X$ (3.4a)

subject to $e^T x = 1$ (3.4b)

$Xe = x$ (3.4c)

$x \geq 0, X \geq 0$ (3.4d)

$X \succeq 0.$ (3.4e)

We will now construct an equivalent semi-infinite linear programming restatement of Problem SDP(QP). This will facilitate the derivation of valid inequalities to augment the first-level RLT relaxation of Problem QP, given by (3.2). Consider the following result.

Proposition 3.1. The problem SDP(QP) given by (3.4) is equivalent to the semi-infinite linear program (SILP(QP)) stated in (3.5) below.

SILP(QP): Minimize $\sum_i \sum_j C_{ij} X_{ij}$ (3.5a)

subject to $e^T x = 1$ (3.5b)

$Xe = x$ (3.5c)

$[(\alpha^T x)^2]_L \geq 0, \quad \forall \alpha \in R^n \ni \|\alpha\| = 1$ (3.5d)

$x \geq 0, X \geq 0$ and symmetric. (3.5e)

Proof. By definition, $X \succeq 0$ is equivalent to requiring that *X* is symmetric and that $\alpha^T X \alpha \geq 0, \quad \forall \alpha \in R^n \ni \|\alpha\| = 1$, noting that any nonzero $\alpha \in R^n$ can be made of unit length. But $\alpha^T X \alpha = [\alpha^T (xx^T)\alpha]_L = [(\alpha^T x)(x^T \alpha)]_L = [(\alpha^T x)^2]_L$. Hence, (3.4e) is equivalent to requiring *X* to be symmetric and such that (3.5d) holds true. This completes the proof. □

Proposition 3.1 reveals a connection between RLT and semidefinite relaxations. Observe that (3.5a,b,c, and e) are respectively identical to (3.2a,b,c, and d) that define the first-level RLT relaxation RLT-1(QP). The constraint set (3.5d) provides a potential strengthening of SDP(QP) or SILP(QP) over RLT-1(QP). The first-level RLT relaxation replaces the nonlinear substitution restriction $X = xx^T$ by simply requiring *X* to be nonnegative and symmetric. On the other hand, the semidefinite relaxation also requires *X* to satisfy the positive semidefiniteness condition associated with the identity $X = xx^T$. But note that as explored in Sherali and Tuncbilek (1997) and Audet *et al*. (2000), for example, aside from the minimal RLT representation constraints stated in (3.2) in the present context, the first-level RLT relaxation can optionally incorporate any other classes of linearized quadratic implied constraints. In particular, enhancing RLT-1(QP) with such implied restrictions of the type (3.5d) yields the semidefinite relaxation SDP(QP) as a special case. We therefore refer to the valid inequalities of the type (3.5d) as *semidefinite cuts* (or *SDP cuts*).

Note that if we denote $\alpha^j \equiv \alpha^j(X),\ j = 1,\ldots,n$, as the set of linearly independent normalized eigenvectors of *X*, then $X \succeq 0$ is equivalent to the condition that $(\alpha^j)^T X \alpha^j \geq 0$ for $j = 1,\ldots,n$. Hence, in the relationships embodied in (3.5d), we could focus on just the α-vectors corresponding to such eigenvectors of *X*, and generate violated members of these constraints in a relaxation framework based on detected negative eigenvalues. The Lanczos algorithm could be used for this purpose (see Paige and Saunders (1975), for example). However, because of the complexity of this approach, given that *X* is a variable in the problem, we will find it more convenient to derive a (polynomial-time) separation mechanism for generating suitable members of (3.5d) in a sequential fashion, based on an LU factorization concept for *X*.

3.3.1 Basic SDP Cut Generation

Rather than solving the semi-infinite program SILP(QP) directly, we adopt the following relaxation approach which leads to a cutting plane generation strategy that can be applied in more general contexts. To begin with, we first solve SILP(QP) with the constraints (3.5d) omitted. Note that this relaxation corresponds precisely to the first-level RLT relaxation of QP as given by (3.2). Let us denote the resulting solution to this problem as $(\hat{x}, \hat{X})$. If $\hat{X} \succeq 0$, then $\hat{X}$ solves Problem SILP(QP) (or SDP(QP)) as well. Otherwise, the solution $\hat{X}$ violates at least one of the constraints (3.5d). The task now is to generate a suitable vector of unit length, $\alpha \in R^n$, for which the constraint $\alpha^T X \alpha \geq 0$ is not satisfied when $X = \hat{X}$. This will then yield a cutting plane of type (3.5d).

In essence, our solution strategy recursively evaluates the entries of $\hat{X}$ to determine whether or not $\hat{X}$ is indeed positive semidefinite. Toward this end, consider the application of a superdiagonalization (or upper triangularization) process to the symmetric matrix $\hat{X}$ (see Bazaraa *et al.*, 1993). In this process, proceeding in the order $i = 1,2,\ldots,n$, we continue to zero out the elements in the i^{th} column under the current i^{th} diagonal element by performing elementary row operations using the i^{th} row, so long as the diagonal elements encountered

remain positive. Starting with $G^1 \equiv \hat{X}$ for $i = 1$, at the i^{th} stage in this process, $i \in \{1,\ldots,n-1\}$, suppose that we have encountered all positive diagonal elements thus far, and that we are examining the reduced submatrix $G^i \in R^{(n-i+1)\text{x}(n-i+1)}$ appearing in rows and columns $i, i+1, \ldots n$. Let us view G^i in its partitioned form, where its first row and column are explicitly displayed as follows

$$G^i \equiv \begin{bmatrix} G^i_{11} & (g^i)^T \\ g^i & G \end{bmatrix}, \tag{3.6}$$

and consider the following result.

Proposition 3.2. Given G^i as in (3.6), suppose that $G^i_{11} > 0$ and define

$$G^{i+1} = G - \frac{g^i (g^i)^T}{G^i_{11}} \quad . \tag{3.7}$$

Then G^i is PSD if and only if G^{i+1} is PSD. Moreover, given any $\alpha^{i+1} \equiv (\alpha_{i+1},\ldots,\alpha_n)^T$, by selecting $\alpha_i = \dfrac{-(\alpha^{i+1})^T g^i}{G^i_{11}}$, we have that $(\alpha^i)^T G^i \alpha^i = (\alpha^{i+1})^T G^{i+1} \alpha^{i+1}$ for $\alpha^i \equiv \begin{pmatrix} \alpha_i \\ \alpha^{i+1} \end{pmatrix}$.

Proof. By simplifying terms, we have from (3.6) and (3.7) that

$$(\alpha^i)^T G^i \alpha^i = G^i_{11} (\alpha_i + \frac{(\alpha^{i+1})^T g^i}{G^i_{11}})^2 + (\alpha^{i+1})^T G^{i+1} \alpha^{i+1}.$$

Clearly, if G^{i+1} is PSD, then so is G^i. Conversely, if G^i is PSD, then noting that by setting

$$\alpha_i \equiv \frac{-(\alpha^{i+1})^T g^i}{G^i_{11}} \text{ gives } (\alpha^i)^T G^i \alpha^i = (\alpha^{i+1})^T G^{i+1} \alpha^{i+1}, \tag{3.8}$$

we must have that G^{i+1} is also PSD. Moreover, in any case, (3.8) holds true. This completes the proof. □

Note that the first part of Proposition 3.2 is based on the superdiagonalization procedure for checking the positive semidefiniteness of $\hat{X}$ (see Bazaraa *et al*. (1993)). The related latter part of the result asserts that if G^i is not PSD, then since G^{i+1} must also not be PSD, we can seek an α^{i+1} such that $(\alpha^{i+1})^T G^{i+1} \alpha^{i+1} < 0$, and accordingly, we will have found an α^i with α_i given by (3.8) such that $(\alpha^i)^T G^i \alpha^i < 0$. We can repeat this process recursively until all components of α are determined. Upon normalizing this α, we will have generated a valid linear inequality of the form (3.5d) that is not satisfied for the current solution $\hat{X}$.

Next, consider the situation addressed by the following result in which for some stage

$i \in \{1,\ldots,n-1\}$, we encounter a submatrix G^i of the type (3.6) for which $G^i_{11} = 0$ *and* $g^i \equiv 0$.

Proposition 3.3. Given G^i as in (3.6), suppose that $G^i_{11} = 0$ and that $g^i \equiv 0$. Then by letting

$$G^{i+1} \equiv G \text{ and } \alpha_i = 0, \tag{3.9}$$

we have that G^i is PSD if and only if G^{i+1} is PSD, and moreover,

$$(\alpha^i)^T G^i \alpha^i = (\alpha^{i+1})^T G^{i+1} \alpha^{i+1} \text{ where } \alpha^i = \begin{pmatrix} \alpha_i \equiv 0 \\ \alpha^{i+1} \end{pmatrix}. \tag{3.10}$$

Proof. Similar to the proof of Proposition 3.2. □

This result asserts again that if G^{i+1} is not PSD and we find an α^{i+1} such that $(\alpha^{i+1})^T G^{i+1} \alpha^{i+1} < 0$, then we can recursively recover an α via (3.8) and (3.10) (according to whether the corresponding diagonal element is positive or zero, noting the condition of Proposition 3.3 in the latter case), such that (3.5d) is violated.

Now, let us consider two cases where for the *first time*, a situation other than the foregoing types arises.

Case (i): $G^i_{11} < 0$ in (3.6).

Suppose that in the foregoing diagonalization process, we encounter for the first time a matrix G^i given by (3.6) having $G^i_{11} < 0$. In this case, we can take $\alpha^i = (\alpha_i,\ldots,\alpha_n)^T = (1,0,\ldots,0)$. Then $(\alpha^i)^T G^i \alpha^i = G^i_{11} < 0$, and we can subsequently compute the full vector α inductively using (3.8) and (3.10).

Case (ii): $G^i_{11} = 0$, but $G^i_{1j} = G^i_{j1} = 0 \neq 0$ for some $j \in \{2,\ldots,n-i+1\}$ in (3.6).

In this case, we know that G^i is not PSD and we can find an α for which $\alpha^T \hat{X} \alpha \geq 0$ is violated as follows. Specifically, consider α^i to be of the form $\alpha^i = (\alpha_i,\ldots,\alpha_n)^T = (\alpha_i,0,\ldots,0,\alpha_{i+j-1},0,\ldots,0)^T$. Let $G^i_{jj} = \phi$, $\xi = (\alpha_i, \alpha_{i+j-1})^T$, and $H = \begin{bmatrix} 0 & \theta \\ \theta & \phi \end{bmatrix}$. We then have that $(\alpha^i)^T G^i \alpha^i = \xi^T H \xi$, and if we can determine a ξ for which $\xi^T H \xi < 0$, we will have obtained an α^i for which $(\alpha^i)^T G^i \alpha^i < 0$. By using (3.8) and (3.10) recursively as before, we could thereby find an α for which $\alpha^T \hat{X} \alpha < 0$. This α could then be normalized to produce a valid inequality of the form (3.5d) that must be satisfied for all feasible solutions *X*. In order to determine such a vector α^i, consider the following result.

Proposition 3.4. Let $\xi = (\alpha_i, \alpha_{i+j-1})^T$ and let $H = \begin{bmatrix} 0 & \theta \\ \theta & \phi \end{bmatrix}$, where $\theta \neq 0$. Then $\xi^T H \xi$ is

minimized, subject to $\|\xi\|^2 = 1$, by selecting

$$\alpha_i = \frac{1}{\sqrt{1+\frac{\lambda^2}{\theta^2}}} \quad \text{and} \quad \alpha_{i+j-1} = \frac{\alpha_i \lambda}{\theta} \quad \text{where} \quad \lambda = \frac{\phi - \sqrt{\phi^2 + 4\theta^2}}{2}. \tag{3.11}$$

Moreover, at the solution (3.11), $\xi^T H \xi \equiv \lambda < 0$.

Proof. By the linear independence constraint qualification, the KKT necessary optimality conditions (see Bazaraa *et al*. (1993)) for the problem of minimizing $\xi^T H \xi$ subject to $\|\xi\|^2 = 1$ yield for some λ,

$$H\xi = \xi\lambda, \quad \|\xi\|^2 = 1. \tag{3.12}$$

This implies that at any KKT solution, we have

$$\xi^T H \xi = \xi^T \xi \lambda = \|\xi\|^2 \lambda = \lambda. \tag{3.13}$$

From (3.12) and (3.13), it follows that the optimal objective value sought equals the minimum eigenvalue λ of *H*, and the corresponding normalized eigenvector yields the optimal solution ξ. To find the minimum eigenvalue for *H*, consider the equation $\det(H - \lambda I) = \lambda^2 - \phi\lambda - \theta^2 = 0$.

Using the quadratic formula, we derive the minimum eigenvalue of *H* as $\lambda = \frac{\phi - \sqrt{\phi^2 + 4\theta^2}}{2}$. The corresponding eigenvector of *H* can be found via the system $(H - \lambda I)\xi = 0$, which gives $\alpha_{i+j-1} = \frac{\lambda \alpha_i}{\theta}$. Since, $\|\xi\|^2 = \alpha_i^2 + \alpha_{i+j-1}^2 = 1$, we have $\alpha_i = \frac{1}{\sqrt{1+\frac{\lambda^2}{\theta^2}}}$, where the positive square-root for computing α_i can be chosen without loss of generality. Furthermore, from (3.13), $\lambda = \xi^T H \xi < 0$ since *X* is not PSD. This completes the proof. □

Remark 3.1. Note that in case of alternative choices of elements pertaining to Case (ii) for which Proposition 3.4 can be applied, we can select one that yields the most negative value of λ. □

Example 3.1. To illustrate, consider the following example. Suppose that the current solution $\hat{X}$ is given as follows:

$$\hat{X} = \begin{bmatrix} 0 & 0.15 & 0.15 \\ 0.15 & 0.2 & 0 \\ 0.15 & 0 & 0.2 \end{bmatrix}.$$

With $i = 1$, we have $G^i_{11} = 0$ and $G^i_{1j} = G^i_{j1} \neq 0$ for $j = 2$ and $j = 3$, indicating that there are two possible values of j that can generate a separating inequality. With j=2, we have $G^i_{12} = G^i_{21} = 0.15$. This yields $\xi = (\alpha_1, \alpha_2)^T$ with $\theta = 0.15$ and $\phi = 0.2$ in the notation of Case (ii) above. From (3.11), $\lambda = \dfrac{0.2 - \sqrt{0.2^2 + 4(0.15)^2}}{2} = -0.08028$, $\alpha_1 = \dfrac{1}{\sqrt{1 + \frac{(-0.08028)^2}{0.15^2}}} = 0.8817$, and $\alpha_2 = \dfrac{(-0.08028)(0.8817)}{0.15} = -0.4719$. Hence, $\alpha = (0.8817,\ -0.4719,\ 0)^T$. Note that $\|\alpha\| = 1$ and that $\alpha^T \hat{X} \alpha = \xi^T H \xi = (0.8817,\ \ -0.4719)\begin{pmatrix} 0 & 0.15 \\ 0.15 & 0.2 \end{pmatrix}\begin{pmatrix} 0.8817 \\ -0.4719 \end{pmatrix} = -0.08028$, which is the value of λ. In a similar fashion, we can calculate the corresponding α for j=3 as $\alpha = (0.8817,\ 0,\ -0.4719)^T$, which also produces λ = -0.08028. Thus, the procedure has found two possible choices of α for which $\alpha^T X \alpha \equiv [(\alpha^T x)^2]_L \geq 0$ is not satisfied for the current solution $\hat{X}$. The corresponding linearized constraints are given as

$$0.7774 X_{11} - 0.8321 X_{12} + 0.2226 X_{22} \geq 0$$

$$\text{and } 0.7774 X_{11} - 0.8321 X_{13} + 0.2226 X_{33} \geq 0 .$$

Since both of these cuts produced the same value of λ, we could arbitrarily choose either cut. □

The foregoing approach establishes an inductive polynomial-time process for generating valid inequalities for the first-level RLT relaxation. Since each recursive step of applying this process to G^i at iteration i is of complexity O(n^2) and we perform at most n such steps, the complexity of the overall separation routine is O(n^3). After obtaining an α for which $\alpha^T \hat{X} \alpha < 0$, $\|\alpha\| = 1$, and generating the corresponding inequality $[(\alpha^T x)^2]_L \geq 0$, we can append this to the current RLT relaxation. This problem could then be re-solved to obtain a new solution $(\hat{x}, \hat{X})$, and the procedure could be repeated until any of the following **termination criteria** is realized: the solution $\hat{X}$ for some relaxed problem turns out to be PSD, or some maximum limit K_1 on the number of LPs solved is attained, or the improvement in the lower bound from one iteration to the next is lesser than a prescribed $\delta > 0$ for some p consecutive iterations. Note that, as described in the sequel, we could generate multiple cuts at each iteration. Hence, we also impose a limit, K_2, on the number of inequalities of type (3.5d) that are generated for any particular solution $\hat{X}$. (In our computations, we used K_1 = 100, K_2 = 100, δ = 0.001, and p = 3.) Figure 3.1 gives a flow-chart for this approximate truncated scheme for solving SILP(QP) by way of augmenting RLT-1(QP) with the proposed SDP cuts.

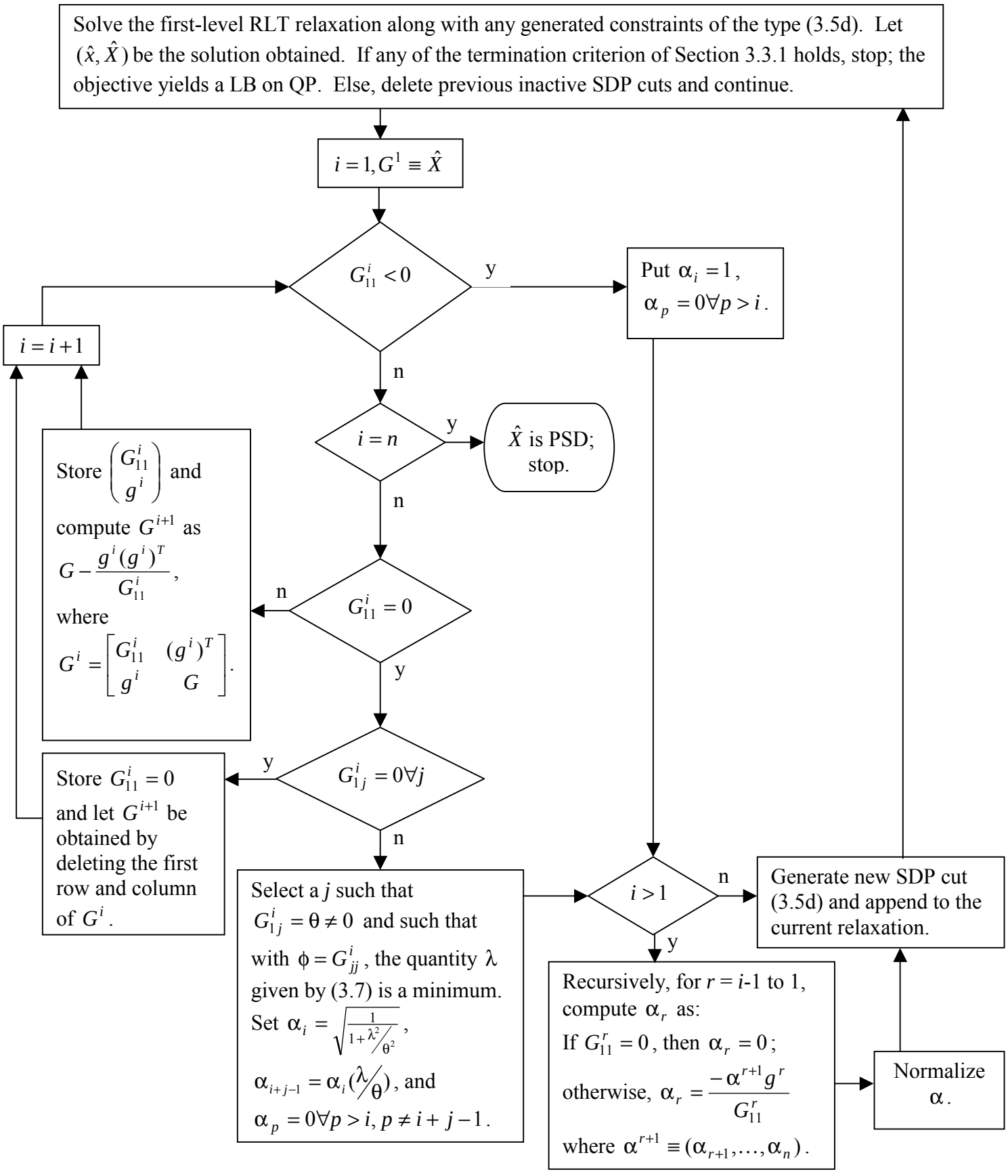

Figure 3.1. Flow-chart for the Fundamental SDP Cut Generation Procedure.

Example 3.2. Suppose that the current solution $\hat{X}$ is given as follows.

$$\hat{X} = \begin{bmatrix} 0.04 & 0.08 & 0.2 \\ 0.08 & 0 & 0 \\ 0.2 & 0 & 0.4 \end{bmatrix}$$

We can see that $\hat{X}$ is not PSD, since $\hat{X}_{22} = 0$ but $\hat{X}_{12} = \hat{X}_{21} = 0.08$. The procedure of Figure 3.1 starts with $i = 1$, $G^1 = \hat{X}$, and examines $G^1_{11} = \hat{X}_{11}$. Since $G^1_{11} > 0$, we store $G^1_{11} = 0.04$, $g^1 = \begin{pmatrix} 0.08 \\ 0.2 \end{pmatrix}$, and we derive the reduced matrix G^2 of Proposition 3.2 via (3.7) as

$$G^2 = \begin{bmatrix} 0 & 0 \\ 0 & 0.4 \end{bmatrix} - \frac{\begin{pmatrix} 0.08 \\ 0.2 \end{pmatrix}(0.08 \quad 0.2)}{0.04} = \begin{bmatrix} -0.16 & -0.4 \\ -0.4 & -0.6 \end{bmatrix}.$$

At $i = 2$, $G^2_{11} = -0.16$ is negative. Hence, we take $\alpha^2 = (\alpha_2, \alpha_3)^T = (1,0)^T$ which gives $(\alpha^2)^T G^2 \alpha^2 = -0.16$. At the final step in Figure 3.1, with $r = 1$, we compute $\alpha_1 = -(\alpha_2, \alpha_3) \cdot g^1 / G^1_{11} = -2$ from Equation (3.8). This yields $\alpha = (-2, \quad 1, \quad 0)^T$ with $\alpha^T \hat{X} \alpha = -0.16$. When we normalize α to $\left(\frac{-2}{\sqrt{5}}, \quad \frac{1}{\sqrt{5}}, \quad 0\right)^T$, we obtain $\alpha^T \hat{X} \alpha = -0.032$. The corresponding SDP cut,

$$[(\alpha^T x)^2]_L = 0.8X_{11} - 0.8X_{12} + 0.2X_{22} \geq 0,$$

is violated for $X = \hat{X}$, since $[(\alpha^T x)^2]_L \equiv \alpha^T \hat{X} \alpha = 0.8(0.04) - 0.8(0.08) + 0.2(0) = -0.032$. □

3.3.2 Enhancing the Basic SDP Cut Generation Strategy

In the cut generation process described above, we have assumed that the matrix $\hat{X}$ is scanned with respect to its i^{th} diagonal element in the order $i = 1, \ldots, n$, and that a single SDP cut is generated once it is revealed that $\hat{X}$ is not PSD. There are several variations to this strategy that we could possibly adopt. One such variation is a *look-ahead feature* for the cut generation process. In this modification, when the matrix under consideration is G^i having dimension $n - i + 1$, we scan the entire diagonal (G^i_{qq} for $q = 1, \ldots, n - i + 1$) to see if any diagonal element is negative. If we find such a negative diagonal element, say $G^i_{QQ} < 0$, we take $\alpha_{i+Q-1} = 1$ and $\alpha_p = 0, \ \forall p \geq i, p \neq i + Q - 1$. As before, we use Equations (3.8) and (3.10) recursively to determine $\alpha_p \ \forall p \leq i - 1$ (if $i \geq 2$). In a similar manner, we can look ahead for cases where there is a diagonal element that equals zero, say, $G^i_{QQ} = 0$, but $G^i_{Qk} \neq 0$ for some $k \in \{1, \ldots, n - i + 1\}$,

and generate a cut based on this revealed violation of positive semidefiniteness. Figures 3.2 and 3.3 provide detailed flow-charts of routines for implementing this look-ahead feature. Here, we use "status = 0" to indicate that we should continue to increment *i* and look for additional cuts. Since, barring a further permutation of rows and columns of $\hat{X}$, it is only valid to increment *i* when either $G^i_{11} > 0$ or when $G^i_{1j} = G^i_{j1} = 0 \; \forall j = 1, \ldots, n-i+1$ we set "status = 1" when either a Case (i) or Case (ii) violation is detected with respect to the leading element G^i_{11} of G^i.

As a second variant of this strategy, whenever the leading element of the current reduced matrix G^i yields a Case (i) or Case (ii) violation, we generate the valid cuts as above, but instead of exiting from the cut generation routine, we examine if any of the other diagonal elements are positive. If so, we permute the rows and columns of G^i to make the most positive diagonal element as the leading element, and continue the cut generation process, taking care to record the appropriate order of the permuted indices for generating future cuts. Let us refer to this technique as the *full permutation strategy*. Since such a permutation strategy can consume significant computational effort, a third variant is developed in order to decrease computational effort while maintaining the benefits of permutation. In this variant, called the *diagonal sort strategy*, we perform an *nlog*(*n*) sort to arrange the diagonal elements in nonincreasing order, and we continue to generate cuts until we encounter a Case (i) or Case (ii) violation from the leading diagonal element. A fourth variant that applies to all of the foregoing strategies adds multiple cuts at each iteration, also using the look-ahead feature. Since there might be several distinct choices of α for composing SDP cuts as revealed during the sequential look-ahead process for the current solution $\hat{X}$, we attempt to generate a bundle of SDP cuts for each such $\hat{X}$ in order to possibly reduce the computational time for the overall solution process. For all variants, we delete previously generated inactive cuts at each iteration. (We also implement an efficient check to avoid the generation of duplicated cuts.) In our experimental analysis, we will investigate both the single and multiple cut implementations, using both the original matrix $\hat{X}$ as well as an augmented matrix that will be considered in Section 3.3.3.

Example 3.3. To illustrate these variants, consider the matrix $\hat{X}$ from Example 3.2:

$$\hat{X} = \begin{bmatrix} 0.04 & 0.08 & 0.2 \\ 0.08 & 0 & 0 \\ 0.2 & 0 & 0.4 \end{bmatrix}.$$

With *i* = 1 and $G^1 = \hat{X}$, we can look-ahead and see that $\hat{X}_{22} = 0$ but $\theta = \hat{X}_{21} = \hat{X}_{12} = 0.08$. Accordingly, we can derive a violated constraint at this point itself, before incrementing *i* and examining G^2. If we take $\xi = (\alpha_2, \alpha_1)^T$, $\theta = \hat{X}_{12} = \hat{X}_{21} = 0.08$, and $\phi = \hat{X}_{11} = 0.04$, we obtain from Proposition 3.4 that $\alpha = (-0.6154, \quad 0.7882, \quad 0)^T$. The corresponding SDP cut is

$$0.3787 X_{11} - 0.9701 X_{12} + 0.6213 X_{22} \geq 0,$$

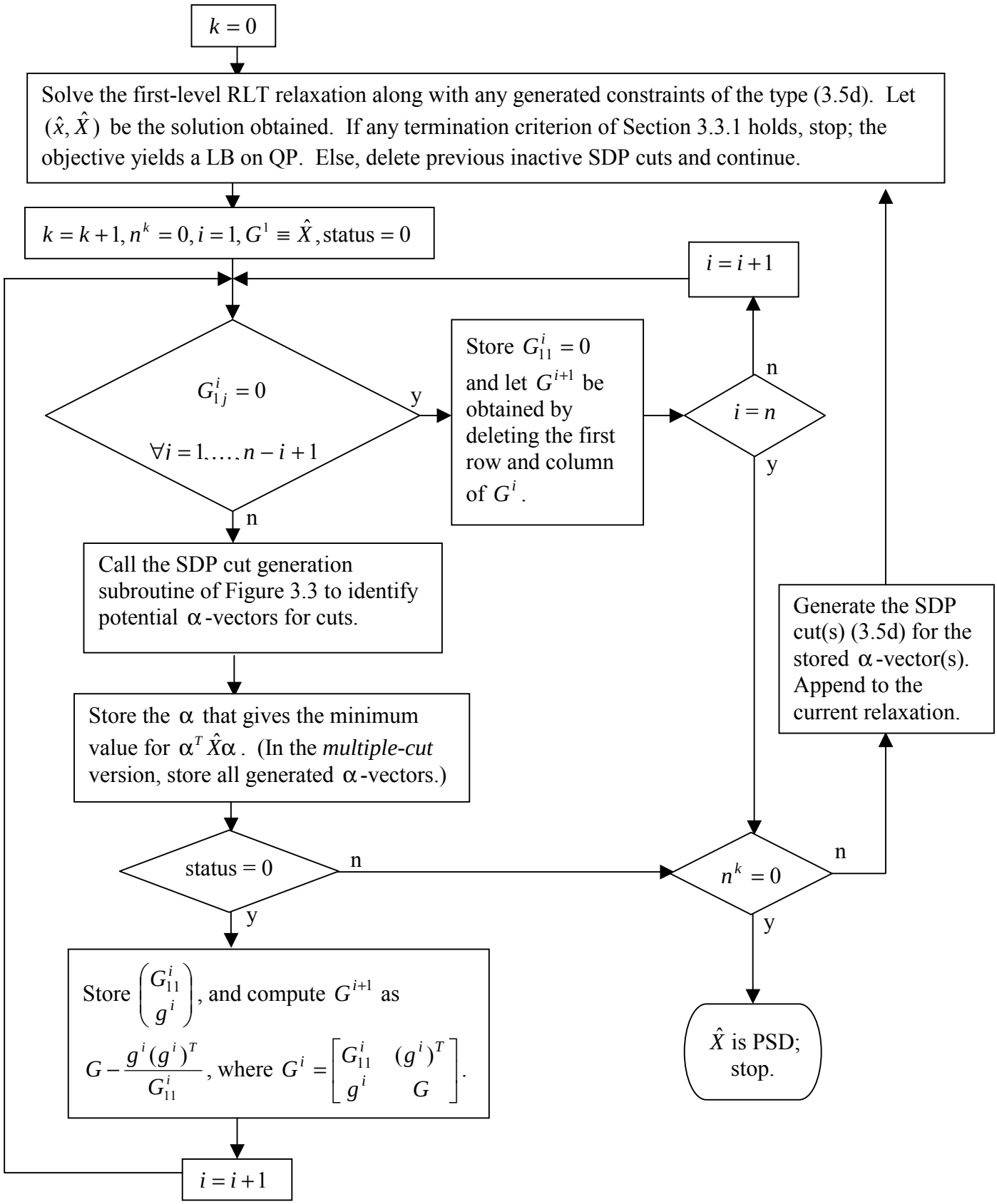

Figure 3.2. Flow-chart for the Look-Ahead SDP Cut Generation Procedure.

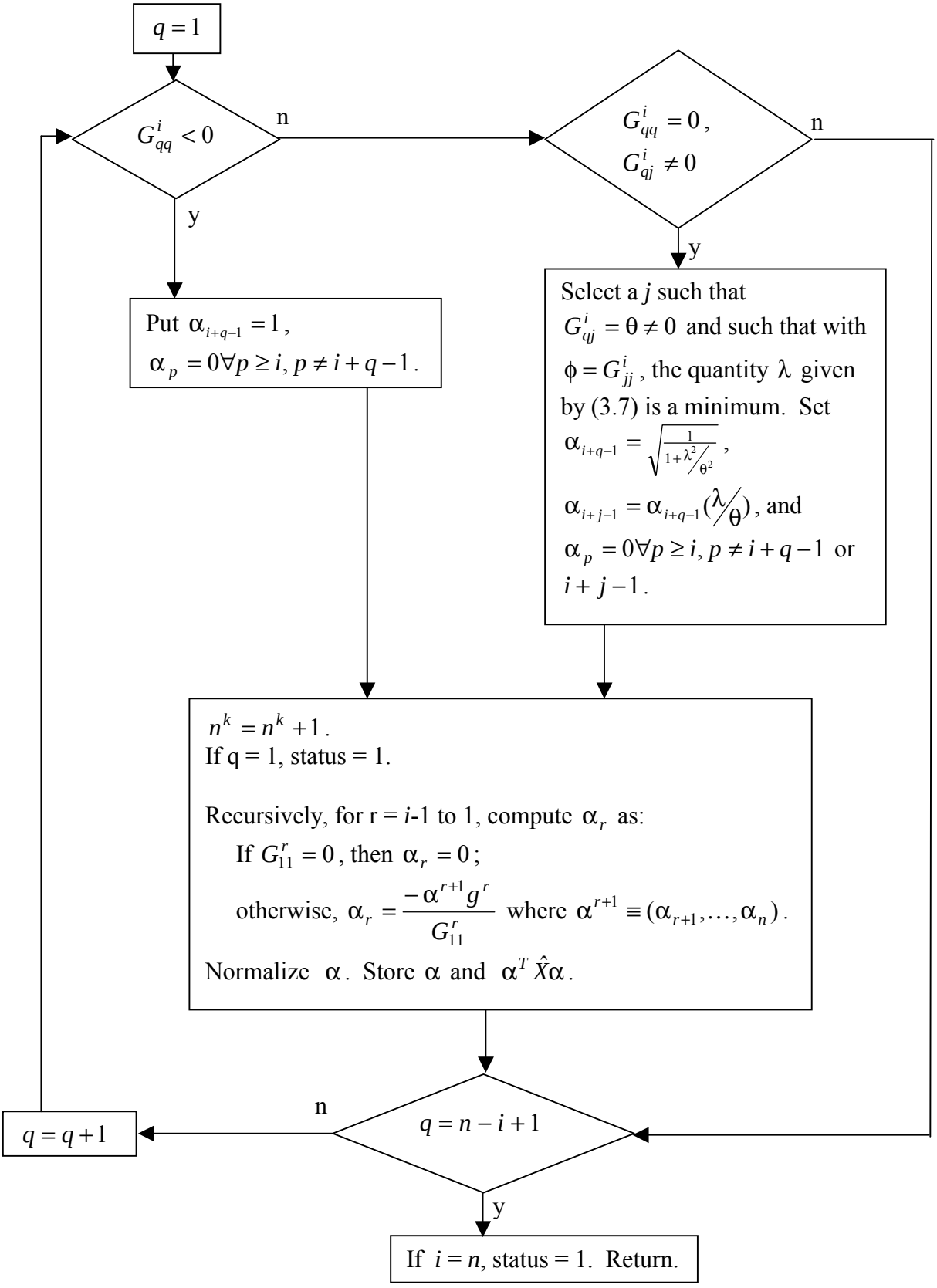

Figure 3.3. Flow-chart for the SDP Cut Generation Subroutine Invoked by the Look-Ahead Procedure of Figure 3.2.

which is currently violated since $\alpha^T \hat{X}\alpha = -0.0625$. Now, since $G_{11}^1 > 0$, we could continue to increment *i* as before and generate the following SDP cut that was obtained in Example 3.2:

$$0.8X_{11} - 0.8X_{12} + 0.2X_{22} \geq 0,$$

with the corresponding $\alpha^T \hat{X}\alpha = -0.032$. Observe that the SDP cut generated by looking ahead had a violation nearly twice as large as the latter cut. In implementing the single-cut option of Figure 3.2, we would only add the first cut since we select the one cut that yields the largest violation. However, when using the multiple cut implementation of Figure 3.2, we would impose both of the above cuts before re-solving the current relaxation. □

Remark 3.2. As a computational expedient, we have adopted the strategy to terminate the above cut generation process when either the resulting solution matrix $\hat{X}$ is PSD or when some practical stopping criterion is attained. In our computations, as indicated above, we set limits on the maximum number of cuts and iterations, as well as on the number of successive iterations performed while obtaining insufficient progress in tightening the lower bound. A question of interest that arises in this context is whether such a process can be induced to attain the ideal termination condition of $\hat{X}$ being PSD, even in an infinite convergence sense, if the other practical stopping criteria are omitted. One approach for attaining such a theoretically convergent process would be to impose a *spacer step*, whereby finitely often, a vector α is generated uniformly distributed on the surface of a unit sphere in R^n. Then, if $\hat{X}^*$ is the limiting matrix for some convergent subsequence of solutions $\hat{X}$ generated in an infinite process, we could not have the situation that there exists an $\bar{\alpha}$ for which $\bar{\alpha}^T \hat{X}^* \bar{\alpha} < 0$, because then there would exist an $\varepsilon-$neighborhood $N_\varepsilon(\bar{\alpha})$ about $\bar{\alpha}$ for which

$$\alpha^T \hat{X}^* \alpha < 0 \quad \forall \alpha \in N_\varepsilon(\bar{\alpha}) \cap \{\alpha : \|\alpha\| = 1\}.$$

This would imply the absence of having generated any α in the latter region which has a nonzero measure on the surface of the unit sphere, a contradiction to the uniform distribution of the generated values of α on the surface of this sphere. □

3.3.3 SDP Cuts Using an Augmented Matrix

The development of the semidefinite cuts in Section 3.3.1 was based on the noting that the identity $X = xx^T$ implies the PSD restriction $X \succeq 0$. Another common tactic in semidefinite programming is to recognize that $X = xx^T$ also implies the stronger condition that $X \succeq xx^T$ (see Nowak (1998a,b, 1999), for example). Note that $X \succeq xx^T$ can equivalently be expressed as $\begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0$. From the viewpoint of RLT constraints (as per Proposition 3.1), $\begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0$ translates to the class of SDP cuts

$$[(\alpha^T x + \alpha_{n+1})^2]_L = \alpha^T X \alpha + 2\alpha_{n+1}(\alpha^T x) + \alpha_{n+1}{}^2 \geq 0 \;\; \forall (\alpha^T, \alpha_{n+1}) \ni \left\| (\alpha^T, \alpha_{n+1}) \right\| = 1 .$$

In terms of the separation routine of the foregoing section, an identical procedure can be implemented on the matrix X^A, where $X^A = \begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix}$. That is, given a solution ($\hat{x}$, $\hat{X}$), we can construct the matrix $\hat{X}^A = \begin{bmatrix} \hat{X} & \hat{x} \\ \hat{x}^T & 1 \end{bmatrix}$ and apply the routine of Section 3.3.1 to the matrix $\hat{X}^A$ in lieu of $\hat{X}$.

Example 3.4. To illustrate the cut generation procedure using the foregoing augmented matrix, consider $\hat{X}$ as given in Example 3.1, and suppose that for all *i*, we have $\hat{x}_i = \sum_j \hat{X}_{ij}$ as required by (3.5c). This leads to the matrix $\hat{X}^A$ as follows:

$$\hat{X}^A = \begin{bmatrix} 0 & 0.15 & 0.15 & 0.3 \\ 0.15 & 0.2 & 0 & 0.35 \\ 0.15 & 0 & 0.2 & 0.35 \\ 0.3 & 0.35 & 0.35 & 1 \end{bmatrix}.$$

Since the upper left portion of the matrix contains $\hat{X}$, we can still derive the two SDP cuts that were obtained in Example 3.1. However, with the additional row and column of $\hat{X}^A$, we also have another possibility for generating an SDP cut inequality. With *i*=1, we have $\hat{X}^A_{11} = 0$, but $\hat{X}^A_{14} = \hat{X}^A_{41} = 0.3 \neq 0$, and so we can apply Proposition 3.4 with $\xi = (\alpha_1, \alpha_4)^T$, $\theta = 0.3$, and $\phi = 1$. From (3.11), we get $\lambda = \dfrac{1 - \sqrt{1^2 + 4(0.3)^2}}{2} = -0.0831$, $\alpha_1 = 0.9637$, and $\alpha_4 = -0.2669$. Hence, $\begin{pmatrix} \alpha \\ \alpha_{n+1} \end{pmatrix} = (0.9637,\ 0,\ 0,\ -0.2669)^T$. Note that $\left\| \begin{pmatrix} \alpha \\ \alpha_{n+1} \end{pmatrix} \right\| = 1$ and that $\begin{pmatrix} \alpha \\ \alpha_{n+1} \end{pmatrix}^T \hat{X}^A \begin{pmatrix} \alpha \\ \alpha_{n+1} \end{pmatrix} = \xi^T H \xi = -0.0831$, which is the value of λ. The corresponding SDP cuts is given by

$$-0.5145 x_1 + 0.9287 X_{11} \geq -0.07125 ,$$

which is currently violated, since we have $-0.5145\hat{x}_1 + 0.9287\hat{X}_{11} = -0.15435$. Thus, the procedure has found an $\begin{pmatrix} \alpha \\ \alpha_{n+1} \end{pmatrix}$ for which $\begin{pmatrix} \alpha \\ \alpha_{n+1} \end{pmatrix}^T X^A \begin{pmatrix} \alpha \\ \alpha_{n+1} \end{pmatrix} \equiv [(\alpha^T x + \alpha_{n+1})^2]_L \geq 0$ is not satisfied for the current solution $\hat{X}^A$. Recall that both of the cuts derived in Example 3.1 had $\alpha^T \hat{X} \alpha = -0.0803$; hence, examining the augmented matrix has produced a cut that is violated to a greater extent than the former cuts. In the single-cut option, we would therefore implement the

cut that was generated by the present example, since it has a larger violation than either of the cuts generated in Example 3.1. In the multiple-cut implementation, we would append all of these generated cuts to the current RLT-1 relaxation before returning to re-solve the next relaxation. □

3.4 Computational Analysis

To gauge the effectiveness of the proposed class of SDP cuts in solving Problem QP, we conducted two types of computational experiments. We first conducted an experiment to evaluate the relative performance of the various cut generation strategies in enhancing the lower bound derived by RLT-1(QP) at the root node within a branch-and-bound framework. Following this analysis, we selected several of the best SDP cut generation strategies and implemented them within a branch-and-bound framework in order to assess their performance in the search for a globally optimal solution.

3.4.1 Root Node Performance

In this section, we compare how the different cut generation strategies compared as to the bound they obtained for the root node problem of a branch-and-bound tree. The first strategy, which serves as a baseline case, uses a single cut per iteration derived from the matrix $\hat{X}$ using no permutations. The remaining six strategies were composed by using each combination of the two matrix types (regular and augmented) with the three permutation types described in Remark 3.2 (no permutation, full permutation, and diagonal sort). Since some preliminary computations indicated that the single cut approach was dominated by the multiple cut implementation, we consider the single cut strategy only in the baseline case. In addition to the stopping criteria mentioned in Section 3.3.1, we also limited each of the strategies to 60 seconds of CPU time per problem. (All computations were executed on a SUN Ultra-1 workstation, with CPLEX 6.5 being used to solve the generated LP relaxations.)

The sizes of the test problems range from 10 variables to 100, by increments of 10. For each problem, the objective coefficients were generated uniformly on the interval [0,10]. The objective coefficients C_{ii} of the terms x_i^2 were always taken to be positive, while the coefficients C_{ij} of the terms $x_i x_j$ were permitted to be positive or negative. In order to vary the problem structure for a given size, the proportion of positive C_{ij} coefficients was varied through four values (0.1, 0.33, 0.66, 0.9), and four problems were generated for each such value, creating a total of 16 problems for each problem size. We obtained a lower bound for each of these 160 problems using each of the seven proposed strategies. The data is summarized in Table 3.1. For each problem, the SDP cut-enhanced bounds were all tighter than the RLT-1 bound, and the improvement was most pronounced with higher proportions of positive C_{ij} coefficients and smaller problem sizes. For instance, for the (four) 10-variable problems having 90% of the C_{ij} coefficients positive, the best SDP cut-enhanced bound improved the RLT-1 bound by an average of 65%; however, for the 100-variable problems having 10% of the C_{ij} coefficients

Table 3.1: Average % Improvement of the Best SDP Cut Bound over the RLT-1 Bound.

		Number of Variables									
		10	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
Proportion	**0.1**	45.63	34.76	25.01	15.94	11.50	7.40	5.23	3.41	1.28	1.35
of Positive	**0.33**	56.72	43.86	29.46	21.48	14.13	10.25	5.69	4.18	2.38	1.43
C_{ij}	**0.66**	59.30	55.44	46.18	35.40	24.20	19.94	14.37	10.25	7.34	5.85
Coefficients	**0.9**	65.18	64.28	59.69	58.10	52.38	44.76	39.61	33.02	28.11	19.14

positive, the SDP cut-enhanced bound only improved the RLT-1 bound by an average of 1.35%. In order to assess the relative performances of the different cut generation strategies, we ranked these methods for each problem size with respect to the bound obtained at the root node, as well as with respect to the CPU time required. For each problem, we computed the best (greatest) lower bound and the best (smallest) CPU time, and then calculated the percentage amount by which each method deviated from the best bound and time for the given problem. Since we have 16 problems of each size being solved using each of the seven strategies, this yields a total of 112 data points for each value of n. These data points pertaining to the bound and time deviations were ranked separately in increasing order for each value of n. In the case of ties, average ranks were assigned so that the sum of the ranks for each n equals $\sum_{i=1}^{112} i = 6328$. Tables 3.2 and 3.3 present the rank-sums for each strategy for each value of n, as well as over the ten problem sizes, for the two respective criteria: lower bounds and CPU times.

The results indicate that the baseline strategy provides significantly worse bounds than its more sophisticated counterparts, but it has a slightly better than average performance with respect to computational time. When used with the regular matrix, the full permutation strategy provides a distinctly better bound than the non-permutation and diagonal-sort strategies, and this trend occurs across all problem sizes. Both permutation strategies (full or diagonal sort) provide a tighter lower bound than the non-permutation strategy when used in combination with the regular matrix, but the effect is less clear when used in combination with the augmented matrix strategy. There are several notable cases where the permutation strategy does not tighten the bounds obtained from the non-permuted method. In general, the augmented matrix strategy provides an improvement in bounds as compared to the regular matrix strategy, particularly as problem size increases. Overall, the rankings indicate that Strategies 3 and 7 provide the best lower bounds, although Strategies 4, 5, and 6 are also competitive. Note that Strategy 3 performs better for smaller problems, while Strategies 5, 6, and 7 tend to perform better as the problem size increases. From Table 3.3, we see that, in general, the methods using the augmented matrix tend to require less computational time, with Strategies 5 and 7 emerging as clearly more time-efficient. Based upon the rankings shown in Tables 3.2 and 3.3, it appears that Strategy 7 provides desirable results in terms of both the quality of the lower bound obtained and the amount of CPU time consumed. In particular, it seems promising that Strategy 7 also performs well in both categories as problem size increases.

In order to determine whether or not the differences in the strategy rankings were significant, we performed a Kruskal-Wallis (rank-sum) test on the data for each n for the seven

Table 3.2: Sum of Lower Bound Rankings.

		Strategy						
		1	**2**	**3**	**4**	**5**	**6**	**7**
	Matrix Type	Reg.	Reg.	Reg.	Reg.	Aug.	Aug.	Aug.
	# Cuts	Single	Multi	Multi	Multi	Multi	Multi	Multi
	Permutation	None	None	Full	Diag. Sort	None	Full	Diag. Sort
	10	1252	1011.5	798.5	803	1070.5	798	594.5
	20	1290.5	1077	305	541.5	1125	945	1044
	30	1153	895.5	293.5	661	936.5	1358	1030.5
	40	1125	978	627	875	625	1410	688
n	**50**	1113.5	1086.5	735.5	992.5	673.5	1014	712.5
	60	1165	1079	798.5	941.5	803.5	664	876.5
	70	1124.5	1058	987.5	1013.5	729	654	761.5
	80	1163	1100	1100	1100	633	673.5	558.5
	90	1090.5	1071.5	1071.5	1071.5	730	502	791
	100	1078.5	1099	1045	1068.5	747.5	487.5	802
	Total	**11555.5**	**10456**	**7762**	**9068**	**8073.5**	**8506**	**7859**

Table 3.3: Sum of CPU Time Rankings.

	Strategy (as defined in Table 3.2)						
n	**1**	**2**	**3**	**4**	**5**	**6**	**7**
10	869	759	1108	930	415	1269.5	977.5
20	636	732	1225.5	1067.5	467.5	1478.5	721
30	869	853	1227	1172	716.5	799.5	691
40	1014	864	1099	1132	823	739	657
50	913	807	1128	1325	590	891.5	673.5
60	928.5	1026	992	1180	496	1149.5	556
70	1233	875	874	1175	559.5	936.5	675
80	579.5	818.5	1617	934	369	1215	795
90	752	934	944.5	1137.5	510	1254	796
100	985	1023.5	892.5	1101	405	1181.5	739.5
Total	**8779**	**8692**	**11107.5**	**11154**	**5351.5**	**10914.5**	**7281.5**

strategies. Table 3.4 indicates that the CPU times were significantly different at the 5% level ($h > \chi^2_{0.05,6} = 12.592$) for each problem size other than for $n = 40$, and the lower bounds were significantly different for all sizes except for $n = 60$ and $n = 70$. We performed an additional Kruskal-Wallis test by analyzing the combined data from all problem sizes. That is, we ranked each of the percentage deviations from 1 through 1120 (= 160 x 7), and performed the Kruskal-Wallis test using a sample size equal to 160 for each strategy. The test statistics for the lower bounds and CPU times were 66.77 and 119.9, respectively, which were much greater than $\chi^2_{0.05,6} = 12.592$, indicating significant differences in the performance of the seven strategies.

As final comparative evidence, we directly display in Table 3.5 the number of problems (out of 160) for which each strategy obtained the best lower bound and CPU time. The strategies that use the augmented matrix have the largest proportion of best lower bounds and best CPU times. Of the strategies based on the regular matrix, the ones that employed the full permutation and the diagonal sort techniques performed significantly better than the one that used no permutation.

Table 3.4: h-Statistic for the Kruskal-Wallis Test.

	h-Statistic	
n	**Lower Bound**	**CPU Time**
10	17.11	26.23
20	43.83	46.55
30	42.49	16.08
40	30.37	11.76
50	12.76	23.34
60	10.65	26.5
70	12.13	21.07
80	25.38	60.25
90	19.18	21.90
100	19.19	24.47

Table 3.5: Number of Problems for which the Best Lower Bounds and CPU Times were Achieved for Each Strategy.

	Strategy (as Defined in Table 3.2)						
	1	**2**	**3**	**4**	**5**	**6**	**7**
Lower Bound	12	16	46	27	48	67	49
CPU Time	18	9	1	3	83	26	22

3.4.2 Overview of the Branch-and-Bound Procedure

Before presenting the results of the branch-and-bound analysis, we first present an overview of the branch-and-bound procedure. Each node of the branch-and-bound tree contains an RLT relaxation of a problem, augmented by a series of SDP cuts. We focus here on developing the RLT representation for each node, given *l* and *u* as the appropriate vectors of upper and lower bounds, respectively, for the original variables. We begin, as before, by multiplying the simplex constraint (3.2b) by each variable. This yields the problem (3.2), with (3.2d) replaced by $l \le x \le u$, $X \ge 0$ and symmetric. We then augment this representation with a set of constraints obtained by multiplying the bound-factors pairwise, as in Sherali and Tuncbilek (1992). We note that all variables (original and RLT) are implicitly bounded between zero and one, with the bounds for the original variables implied by (3.2b) and (3.2d), while the additional constraints (3.2c) imply the same bounds for the RLT variables. We therefore need only include bound-factor product constraints when the corresponding bounds are tighter than the implied bounds of 0 and 1. The pairwise products of bound-factors result in six types of constraints, as outlined in Table 3.6. Types I, III, and V are simply specializations of Types II, IV, and VI, respectively, for the case when $j = i$. The maximum number of each type of constraint is also presented in Table 3.6, giving a total of $2n^2 + n$ potential constraints, where *n* is the number of original variables. We note, however, that several of these constraints may be unnecessary. For example, in the case where $l_i = 0$, the corresponding Type I constraint reduces to a simple nonnegativity constraint on X_{ii}. If, in addition $l_j = 0$, the Type II constraint also reduces to a simple nonnegativity constraint on X_{ij}. Since at the root node we have $l_i = 0$ and $u_i = 1$ for each *i*, all of the bound-product constraints reduce to nonnegativity restrictions on the RLT variables. At subsequent nodes, however, we will not necessarily have $l_i = 0$ and $u_i = 1$ for each *i*, thereby requiring us to generate the appropriate bound-factor products. In summary, then, the initial relaxation at each node (prior to adding SDP cuts) is given as follows:

RLT-1(QP): Minimize $\sum_i \sum_j C_{ij} X_{ij}$ (3.14a)

subject to

$$e^T x = 1 \quad (3.14b)$$

$$Xe = x \quad (3.14c)$$

$$X_{ii} - 2l_i x_i \ge -l_i^2 \quad \forall i \ni l_i > 0 \quad (3.14d)$$

$$X_{ij} - l_j x_i - l_i x_j \ge -l_i l_j, \quad \forall i < j \ni l_i \text{ or } l_j > 0 \quad (3.14e)$$

$$(l_i + u_i) x_i - X_{ii} \ge l_i u_i, \quad \forall i \ni u_i < 1 \quad (3.14f)$$

$$u_j x_i + l_i x_j - X_{ij} \ge l_i u_j, \quad \forall i \ne j \ni u_j < 1 \quad (3.14g)$$

$$X_{ii} - 2u_i x_i \ge -u_i^2, \quad \forall i \ni u_i < 1 \quad (3.14h)$$

$$X_{ij} - u_j x_i - u_i x_j \ge -u_i u_j \quad \forall i < j \ni u_i, u_j < 1 \quad (3.14i)$$

$$l \le x \le u, \quad X \ge 0, \text{ and symmetric.} \quad (3.14j)$$

Upon obtaining the solution to this RLT relaxation in (3.14), we examine the matrix $\hat{X}$ and generate SDP cuts as described previously. Since we are employing the SDP cuts to

Table 3.6: Types of Bound-Factor Constraints.

Type	1st Factor	2nd Factor	Linearized Product	Maximum Number
I	$x_i \geq l_i$	$x_i \geq l_i$	$X_{ii} - 2l_i x_i \geq -l_i^2$	n
II	$x_i \geq l_i$	$x_j \geq l_j$	$X_{ij} - l_j x_i - l_i x_j \geq -l_i l_j$	$\frac{n(n-1)}{2}$
III	$x_i \geq l_i$	$x_i \leq u_i$	$(l_i + u_i)x_i - X_{ii} \geq l_i u_i$	n
IV	$x_i \geq l_i$	$x_j \leq u_j$	$u_j x_i + l_i x_j - X_{ij} \geq l_i u_j$	$n(n-1)$
V	$x_i \leq u_i$	$x_i \leq u_i$	$X_{ii} - 2u_i x_i \geq -u_i^2$	n
VI	$x_i \leq u_i$	$x_j \leq u_j$	$X_{ij} - u_j x_i - u_i x_j \geq -u_i u_j$	$\frac{n(n-1)}{2}$

tighten the bounds within a branch-and-bound framework, we do not necessarily need to solve the SDP relaxation (as given by the SILP representation) to optimality. In our computational analysis, we allowed a maximum of 100 cuts to be generated per iteration, and we limited such sequential rounds of cuts per node to either one or five (as specified). We also included the corresponding SDP cuts that were generated at the nodes on the chain connecting the current node to the root node in the enumeration tree. These cuts are likely to be most effective for the current node subproblem, although the cuts generated elsewhere in the tree are also valid. We took the maximum number of stored cuts as three times the maximum number of cuts that could be generated at any given node (i.e., 300 for the one-round-of-cuts limit and 1500 for the five-rounds-of-cuts case). In case this number exceeded the maximum allowable number of implemented cuts, we overwrote the cuts that were generated the earliest.

Throughout the process, we track the best known solution (incumbent solution) and maintain a list of active nodes listed in order of increasing lower bounds. At the start of the problem, the list contains only the root node with a lower bound of negative infinity and an upper bound of infinity. When a node is selected from the list, the solution to its SDP cut-enhanced problem yields a lower bound on its optimal solution, and since we have linear constraints, the LP solution for each node subproblem also provides an upper bound. In our experimental analysis, we fathomed nodes when the lower bound exceeded $(1-\varepsilon)z_{upper}$, where z_{upper} is the best-known solution value. In our computations, we used $\varepsilon = 0.0001$ for the 10- and 20-variable problems, and we used $\varepsilon = 0.01$ for the 30-variable problems. If the current node cannot be fathomed, we select a branching variable and create two children nodes in which all variable bounds are the same as the parent node, except those corresponding to the branching variable. We select the branching variable, x_p, as given by

$$p \in \arg\max_{i=1,\ldots,n} \{\delta_i = \left| \sum_j C_{ij} (\hat{x}_i \hat{x}_j - \hat{X}_{ij}) \right| \},$$

and we split the current interval $[l_p, u_p]$ at the value $\tilde{x}_p$ in order to derive two children nodes,

where

$$\widetilde{x}_p = \begin{cases} \hat{x}_p, \text{ if } \min\{\hat{x}_p - l_p, u_p - \hat{x}_p\} \geq 0.1(u_p - l_p) \\ \dfrac{l_p + u_p}{2}, \text{ otherwise.} \end{cases}$$

This induces convergence to a global optimum (see Sherali and Tuncbilek (1992)). Since the children nodes are more constrained than their parent node, their lower bounds are potentially tighter and are computed via (3.14), augmented by the appropriate SDP cuts. The parent node is then removed from the list of active nodes, and the new nodes are inserted into the list according to the value of their lower bound. The first node in the list (having the least lower bound) is selected as the current node, and the process is repeated. Whenever we update the incumbent solution value z_{upper}, we fathom (remove) all nodes having lower bounds greater than $(1-\varepsilon)z_{upper}$ from the list. The procedure ends when no nodes remain in the list (the upper and lower bounds for the problem have converged within a tolerance of $\varepsilon \cdot z_{upper}$ or when the maximum number of nodes has been reached. In our analysis, we permitted a maximum of 10,000 nodes for the branch-and-bound routine when using RLT alone, and a maximum of 1,000 nodes for the SDP cut-enhanced procedures.

3.4.3 Branch-and-Bound Results

Based upon the root node analysis, we narrowed our study to exploring the performance of using Strategies 3, 4, 5, 6, and 7 to generate SDP cuts within a branch-and-bound framework. As a benchmark in this comparison, we also implemented the RLT-1 strategy without any cutting planes for computing lower bounds. Tables 3.7 through 3.10 display the results obtained for this branch-and-bound experimentation. Note that in all of these tables, the SDP cut strategies are numbered according to the order shown in Table 3.2, and the baseline RLT strategy using no SDP cuts is referred to simply as RLT. Table 3.7 presents the results obtained for the 10-variable problems, and it shows that for nearly every implementation strategy, the SDP cuts provide a *significant* improvement in the performance of the branch-and-bound algorithm over that using the RLT-1 relaxations alone. The SDP cuts greatly reduce the number of nodes generated as might be expected, but also substantially reduce the overall computational effort. Within the SDP cut- enhanced strategies, using five rounds of SDP cuts per node significantly reduces the number of nodes enumerated as compared with using a single round of SDP cuts; however, the computational time is not consistently reduced. In general, using five rounds of cuts proves most valuable for the relatively more difficult problem instances (lower proportions of positive C_{ij} coefficients), and it does not appear to work well in conjunction with the no permutation strategy. Based upon the results from the 10-variable problems, it was evident that Strategy 5 (augmented matrix, no permutation) would not remain competitive for the more difficult problems, and Strategy 5 was dropped from consideration for the remaining analysis. The results for the 20-variable problems are presented in Tables 3.8 and 3.9. Table 3.8 displays the average time and number of nodes for the various problem types and implementation strategies. Note that in contrast to the results for the 10-variable problems, several problems were not solved to optimality within the allowable number of nodes. In such cases when the gap

Table 3.7: Average Computation Time (in seconds) and Average Number of Nodes for Problems of Size $n = 10$.

		Proportion of Positive C_{ij} Coefficients							
Strategy	**Rounds of Cuts**	**0.1**		**0.33**		**0.66**		**0.9**	
		Time	**Nodes**	**Time**	**Nodes**	**Time**	**Nodes**	**Time**	**Nodes**
RLT	**0**	482.41	5657.5	65.84	1006.5	10.95	239.5	0.74	27
3	**1**	152.24	339	24.15	121.5	3.42	40	0.40	11
	5	135.90	92	32.79	44	4.55	17.5	0.79	7
4	**1**	88.16	195.5	22.77	118	4.00	40.5	0.45	11
	5	62.69	40.5	21.84	34.5	3.94	16.5	0.73	7.75
5	**1**	232.89	422	39.62	162.5	5.89	43.5	0.65	12
	5	419.18	205	109.76	96	9.21	23.5	0.70	3.5
6	**1**	187.61	360.5	31.03	127	7.84	51	0.65	12.5
	5	160.80	95.5	33.79	41.5	5.44	13.5	0.66	3.5
7	**1**	69.43	117	24.46	87	4.53	28.5	0.71	10.5
	5	52.76	32	29.80	26	4.21	9	0.81	3.5

Table 3.8: Average Computation Time (in seconds) and Average Number of Nodes for Problems of Size $n = 20$.

		Proportion of Positive C_{ij} Coefficients							
Strategy	**Rounds of Cuts**	**0.1**		**0.33**		**0.66**		**0.9**	
		Time	**Nodes**	**Time**	**Nodes**	**Time**	**Nodes**	**Time**	**Nodes**
RLT	**0**	6485.25	10001	3917.5	7133.5	86.02	371.5	5.91	51.5
3	**1**	5256	978.5	1304.75	426	25.31	61	3.44	22
	5	5695.75	534	887.75	130	25.04	22.5	3.83	9.5
4	**1**	6025.25	990.5	1162.25	367.5	24.82	60	4.11	27
	5	2775.25	323.5	638.75	96	28.34	24.5	3.37	9.5
6	**1**	5604	1001	2509.5	723.5	44.76	75	5.05	20
	5	10414.5	922	2625	311	95.64	26.5	4.93	7
7	**1**	6683.75	1001	1910	447	52.26	46	4.81	19.5
	5	6478.75	503.5	1450.25	157.5	79.77	21.5	7.78	8

Table 3.9: Average Percentage Optimality Gap at Termination for Problems of Size $n = 20$.

Strategy	Rounds of Cuts	Proportion of Positive C_{ij} Coefficients			
		0.1	**0.33**	**0.66**	**0.9**
RLT	**0**	7.07	0.78	0	0
3	**1**	2.02	0	0	0
	5	0.17	0	0	0
4	**1**	1.26	0	0	0
	5	0	0	0	0
6	**1**	5.71	0.02	0	0
	5	2.01	0	0	0
7	**1**	2.71	0	0	0
	5	0.06	0	0	0

Table 3.10: Average Results for Problems of Size $n = 30$.

		Time		**Nodes**		**% Gap**	
		RLT	**SDP**	**RLT**	**SDP**	**RLT**	**SDP**
Proportion	**0.1**	20574.5	12237	10001	499.5	9.50	0.95
of Positive	**0.33**	17640.5	8554	9613	437.5	4.7	0
C_{ij}	**0.66**	1559.75	380	1370.5	65.5	0	0
Coefficients	**0.9**	125.75	58	209.5	25	0	0

between the best-known solution and least lower bound did not fall below 0.01%, we recorded the percentage gap at termination, and we summarize these results in Table 3.9. Note that although several SDP cut strategies do not significantly decrease the computational effort, they do significantly tighten the optimality gap. Similarly, the use of 5 rounds of cuts generally provides better results than 1 round of cuts across nearly all strategies, either by tightening the optimality gap or by decreasing computational effort. The striking result in Table 3.9 is that one strategy, Strategy 4 (regular matrix, diagonal sort) used in combination with 5 rounds of cuts, obtained the optimal solution (within the allowable number of nodes) for every problem. Although Strategy 7 (augmented matrix, diagonal sort) with 5 rounds of cuts also obtained the global optimum for all problems except one, it did not perform as well with respect to computational effort. Note that Strategy 4, with 5 rounds of cuts, dominated the other strategies in terms of both the average number of nodes enumerated and the average computational effort, particularly for the more difficult set of problems.

Based upon the results obtained for the 20-variable problems, we used only one SDP cut strategy, Strategy 4 with 5 rounds of cuts, to solve the 30-variable problems. The results comparing this strategy with the basic RLT scheme are shown in Table 3.10. For the RLT bounding strategy, seven problems could not be solved to optimality (using a 1% tolerance)

within the 10,000 node limit, while the SDP cut-enhanced strategy failed to solve only one problem to global optimality within 1000 nodes. Furthermore, the SDP cuts drastically decrease the average computational effort as well as the number of nodes enumerated across all problem types. The overall results appear to indicate that the SDP cuts significantly decrease the computational effort and the number of nodes required to solve this class of problems to optimality. Moreover, this relative improvement becomes more pronounced as the degree of difficulty of the problem increases (larger n, smaller proportion of positive C_{ij} coefficients).

3.5 Extensions to Higher Levels of RLT

Observe that the proposed class of SDP cuts can be used in any context where RLT is applied (whether this problem admits an overall SDP formulation or not). This includes problems having polynomial objective and constraint functions, factorable programming problems, or even linear mixed-integer programming problems. In all such cases, SDP cuts can be generated based on the (regular or augmented) matrix of second-order RLT variables. While our focus thus far has been on generating cuts to augment the RLT-1 relaxation for a given problem, it is useful to also consider augmenting higher-level RLT relaxations in a similar manner. For example, consider an RLT relaxation that includes fourth-order RLT variables X_{ijkl} representing the product term $x_i x_j x_k x_l, \forall 1 \le i \le j \le k \le l \le n$. Let $X_{(2)}$ denote the *vector* comprising all distinct $\binom{n+1}{2}$ second-order RLT variables, and let $X^{(4)}$ be a *matrix* comprised of the fourth-order RLT variables structured in the form $X^{(4)} \equiv [X_{(2)} X_{(2)}{}^T]_L$. Since $X^{(4)}$ must be PSD, we can impose a class of SDP cuts in the same spirit as (3.5d) in the form

$$\alpha^T X^{(4)} \alpha \equiv [(\alpha^T X_{(2)})^2]_L \ge 0 \;\; \forall \alpha \in R^{\binom{n+1}{2}} \ni \|\alpha\| = 1. \tag{3.15}$$

Then, given any $\hat{X}^{(4)}$ as part of a solution to the RLT relaxation, we can use the techniques of Section 3.3 identically to derive SDP cuts of the type (3.15) involving the higher dimensional variables.

A second-level RLT relaxation of the problem QP, for example, provides another way to strengthen RLT-1(QP). Such a second-order RLT relaxation RLT-2 could be obtained by multiplying (3.2b) by the quadratic bound-factors $x_j x_k \ge 0 \;\; \forall j \le k$. The variable X_{ijk} would then be defined to linearize the product terms of the form $x_i x_j x_k$. In order to have a unique variable represent the product term $x_i x_j x_k$, regardless of the order in which the variables appear in this term, we would define X_{ijk} only for $i \le j \le k$. Accordingly, for arbitrary indices i, j, and k, let $X_{(ijk)}$ represent the appropriate RLT variable that represents the product of x_i, x_j, and x_k. (This same convention is used on the double-subscripted variables as well.) This leads to the following second-level RLT relaxation of QP.

RLT-2(QP): Minimize $\sum_i \sum_j C_{(ij)} X_{(ij)}$ (3.16a)

subject to $\sum_i x_i = 1$ (3.16b)

$\sum_i X_{(ij)} = x_j, \quad \forall j$ (3.16c)

$\sum_i X_{(ijk)} = X_{jk}, \quad \forall j \le k$ (3.16d)

$(x_i \ \forall i, X_{ij} \ \forall i \le j, X_{ijk} \forall i \le j \le k) \ge 0.$ (3.16e)

Note that Problem (3.16) can be viewed as imposing the additional constraints (3.16d) in terms of the new variables X_{ijk} on Problem (3.2). These additional constraints force each element X_{jk} to equal the summation of several components of a three dimensional symmetric matrix. For example, with $n = 5$ and $(j,k) = (2,3)$, the associated constraint in (3.16d) requires X_{23} to be computable as the sum of the following elements of $[X_{ijk}]$:

$$X_{23} = X_{123} + X_{223} + X_{233} + X_{234} + X_{235}.$$

That is, X_{jk} is the sum of all terms of the three dimensional matrix that contain the subscripts j and k. This type of constraint can be viewed as requiring that the two-dimensional matrix X can be obtained by collapsing a three-dimensional symmetric matrix via a summation process.

The semidefinite relaxation of Problem QP was developed by relaxing the variable substitution constraint $X = xx^T$ to $X \succeq 0$, noting that xx^T is PSD. This concept led to the equivalent class of RLT restrictions $[(\alpha^T x)^2]_L \ge 0$ that were imposed on the first-level RLT relaxation RLT-1(QP). Naturally, this same set of constraints is valid for the level-two relaxation (3.16). Note that in this same spirit, similar additional classes of valid inequalities can be generated to further enhance any odd-level RLT relaxation beyond level-one. For example, consider the third-level RLT relaxation of QP. This relaxation contains all of the constraints in (3.16) along with the constraints obtained by multiplying (3.16b) with the cubic bound-factors $x_j x_k x_l \ge 0 \ \forall j \le k \le l$. The linearization scheme would substitute the variable $X_{(P)}$ for the product term $\prod_{j \in P} x_j$, where (P) orders the indices in P in nondecreasing order. The resulting formulation is given as follows.

RLT-3(QP): Minimize $\sum_i \sum_j C_{(ij)} X_{(ij)}$ (3.17a)

subject to $\sum_i x_i = 1$ (3.17b)

$\sum_i X_{(ij)} = x_j, \quad \forall j$ (3.17c)

$\sum_i X_{(ijk)} = X_{jk}, \quad \forall j \le k$ (3.17d)

$$\sum_i X_{(ijkl)} \quad = \quad X_{jkl}, \quad \forall j \le k \le l \qquad (3.17e)$$

$$(x_i \ \forall i, X_{ij} \ \forall i \le j, X_{ijk} \forall i \le j \le k, X_{ijkl} \forall i \le j \le k \le l) \quad \ge \quad 0. \qquad (3.17e)$$

By defining $A_{ijkl} = \alpha_i \alpha_j \alpha_k \alpha_l$, and denoting $A \bullet X = \sum_i \sum_j \sum_k \sum_l A_{ijkl} X_{ijkl}$, we can validly impose the semidefinite programming types of constraints $A \bullet X \ge 0$ for *any* vector $\alpha \in R^n$. This follows from the fact that $A \bullet X = [(\alpha^T x)^4]_L$. For the first-order and second-order RLT relaxations of QP, we imposed the semidefinite restrictions $(\alpha\alpha^T) \bullet X = [(\alpha^T x)^2]_L \ge 0$, $\forall \alpha \in R^n \ni \|\alpha\| = 1$ in constraint (3.5d). In addition, and in a higher-order extension of (3.5d), we can now enhance the third-level RLT relaxation by dropping the substitution constraint $X_{ijkl} = x_i x_j x_k x_l$ as usual, but instead imposing the implied constraints

$$[(\alpha^T x)^4]_L \ge 0, \ \ \forall \alpha \in R^n \ni \|\alpha\| = 1 \ .$$

This results in augmenting RLT-3(QP) with the following semi-infinite sets of constraints.

$$[(\alpha^T x)^2]_L \quad \ge \quad 0, \quad \forall \alpha \in R^n \ni \|\alpha\| = 1 \qquad (3.18a)$$

$$[(\alpha^T x)^4]_L \quad \ge \quad 0, \quad \forall \alpha \in R^n \ni \|\alpha\| = 1. \qquad (3.18b)$$

A similar procedure could be applied to any general RLT relaxation of level $2v-1$, where the additional constraints would correspond to $[(\alpha^T x)^{2r}]_L \ge 0, \ \forall \alpha \in R^n \ni \|\alpha\| = 1, \ \text{for } r = 1, \ldots, v$. In Section 3.3.1, a polynomial-time procedure was developed to generate an α for which (3.18a) is violated or to verify that none exists. It remains to determine a similar separation routine to systematically generate an α for which (3.18b) or any higher-order variant is violated. We propose this task and related computational studies for future research.

3.6 Conclusions and Extensions

In this chapter, we have explored connections between semidefinite programming (SDP) and the Reformulation-Linearization Technique (RLT), and we have used this insight to develop a new class of cuts to enhance RLT relaxations. This concept has been illustrated on a class of problems involving the minimization of a nonconvex quadratic function over a simplex. The process of closing the gap between a first-level RLT relaxation and a semidefinite relaxation for this problem was shown to yield an equivalent semi-infinite linear program in which the set of infinite constraints comprised a particular class of RLT constraints that we called semidefinite cuts (or SDP cuts). Based on this representation, a relaxation and row generation scheme was devised, leading to a polynomial-time SDP cut generation procedure. Several cut generation strategies, based on using the original or augmented matrix of second-order variables, in natural or specially permuted form, were devised and tested. The SDP cut-enhanced relaxations not

only provided significantly tighter lower bounds, but also resulted in a substantial decrease in both the number of nodes enumerated and in the overall computation effort when embedded within a branch-and-bound framework to determine a global optimal solution (particularly for more challenging problem instances). Of the proposed implementation strategies, the use of multiple cuts clearly dominated the single-cut approach, and the permutation and augmented matrix implementations also provided improved results for some problems. For the most challenging problems, the best combined strategy by far used five rounds of SDP cuts at each node, generated via the regular matrix of second-order RLT variables, rearranged using the diagonal sort permutation strategy. Extensions of this research are readily evident. Future research interests include extending the framework developed here to higher levels of RLT, particularly in the hopes of deriving a cut generation procedure for such higher levels. In addition, experiments should be performed in order to analyze the effectiveness of the proposed algorithm on other classes of discrete or continuous nonconvex optimization problems.

Chapter 4: A Modified Benders' Partitioning Strategy for Discrete Optimization Problems

The focus of this chapter is to develop a Benders' decomposition strategy for discrete optimization problems where both the inner and outer stage decisions might involve 0-1 variables. Although we derive the proposed method for a generic discrete optimization problem, the discussion on stochastic integer programs in Chapter 2 elucidates that the technique is readily applicable to two-stage stochastic programs with (mixed) integer recourse. In particular, due to the large number of subproblems that are encountered therein, the proposed methodology could greatly decrease the effort required to solve stochastic integer programs. Since the technique is also applicable to general discrete optimization problems, however, we use the more generic problem notation throughout the remainder of the chapter. As a point of special interest, we also discuss certain specific modifications for exploiting dual-angular structures, such as those that arise in the aforementioned context of stochastic programs.

This chapter is organized as follows. In Section 4.1, we provide the motivation for developing the method. Section 4.2 contains a preliminary development of the methodology, beginning in Section 4.2.1 with a relatively simpler conceptual case for which a suitable convex hull representation can be constructed that permits a finite regular application of Benders' methodology. Section 4.2.2 provides details for how the approach can be modified to take advantage of dual angular structures, and Section 4.2.3 finishes the development for the case where a complete convex hull representation is available. This lays the groundwork for the more usual case discussed in Section 4.3, where such a representation is only partially generated in a sequential fashion as needed within the context of a Benders' branch-and-cut approach. This viewpoint facilitates the generation of valid inequalities during the solution of any given subproblem in a form that renders them valid for any other subproblems by merely substituting the revised first-stage decisions in a derived linear functional term, and also enables the derivation of suitable Benders' cuts that induce finite convergence. Some numerical examples are presented to illustrate the proposed methodology. Section 4.4 addresses finite convergence issues related to the proposed cutting plane approach for solving the subproblems, and Section 4.5 contains conclusions and suggestions for future research.

4.1 Motivation

While we derive the proposed methodology for any generic discrete optimization problem, the main motivation for this research has been to develop a more effective solution technique for stochastic programs with integer recourse. As mentioned in Chapter 2, stochastic integer programs are among the most challenging optimization problems, since they involve stochastic programs and integer programs, both of which are themselves difficult. Stochastic linear programs are typically solved with the L-shaped algorithm, a direct extension of Benders' partitioning, since the problems decompose naturally into first-stage and second-stage (or

recourse) problems. At each iteration of the L-shaped algorithm, we solve one master problem to obtain a first-stage decision, followed by one subproblem *for each* possible realization of the environment. When integer variables are involved in the subproblems, this implies that we must solve a number of integer programs at each step of the L-shaped algorithm. Given any reasonably sized problem, it is impractical to solve each of the subproblems with traditional IP techniques such as branch-and-bound. As we demonstrate in the following section, if we had an *a priori* explicit representation for the convex hull of certain suitable subsystems, we could implement the traditional Benders' method or the L-shaped algorithm with the subproblems being reduced to linear programs. The effort to obtain explicit representations for such convex hulls, however, is generally prohibitive. Due to the number of times we re-solve the subproblems for varying first-stage decisions, even partial convex hull representations that are constructed sequentially can be of great computational benefit. In the following section, we will verify that valid Benders' cuts can be obtained even if we only use certain partial convex hull representations. Rather than *a priori* generating even such partial convex hull representations, however, we propose to solve the subproblems through a cutting plane technique, where the cuts are derived using the RLT process and are designed to construct relevant parts of the convex hull representations in an as-needed fashion. Furthermore, we propose lifting mechanisms for deriving these cuts as functions of the first-stage variables, enabling them to be re-usable in subsequent visits to the subproblem solution stage, and facilitating the development of effective valid Benders' cuts for the master problem.

4.2 Derivation of the Proposed Benders' Strategy

For the sake of wider applicability, we describe our development in terms of the generic problem P that is given below in (4.1). Although this form does not specifically correspond to the notation used for stochastic IPs, it should be evident from the foregoing discussion that the structure of this problem subsumes this class of problems. (Note that in this context, it would be computationally facile, but not necessary, to have constant technology and recourse matrices, as variously assumed in the literature – for example, see Caroe and Tind (1997)).

$$\begin{aligned} \textbf{P}: \quad & \text{Minimize} && cx + dy && (4.1a)\\ & \text{subject to} && Ax + Dy \geq b && (4.1b)\\ & && x \in X, x \in \{0,1\}^n, y \in Y && (4.1c) \end{aligned}$$

where X represents a nonempty polytope in R^n that is defined in terms of the binary variables x, and Y is a compact subset of R^m and represents some linear restrictions on the y-variables, in addition to binary restrictions on a *subset* (say, $y_1, \ldots, y_p$) of the variables. By appropriately incorporating an artificial (interval-bounded) variable column within the y-variable set, we will assume that P is feasible for any fixed $x \in X$, x binary, and moreover, we will also assume that an optimum exists for P.

4.2.1 Benders' Cuts Given a Convex Hull Representation

In order to develop the proposed methodology, we first consider the case where we have an

explicit representation of the convex hull for the subproblem. In this respect, let us define (denoting e as a compatible vector of ones)

$$Z = conv\{(x, y) : Ax + Dy \geq b, 0 \leq x \leq e, y \in Y\} \tag{4.2a}$$
$$\equiv \{(x, y) : Gx + Hy + Fw \geq f\}, \text{ say,} \tag{4.2b}$$

where for convenience, we have also absorbed any simple bound restrictions within the inequalities describing (4.2b). Note that the description (4.2b) is assumed to be derived in a higher dimensional space (including a set of new w-variables), as for example by using the RLT process (see Sherali and Adams, 1990, 1994, 1999). Note also that aside from the bounding constraints $0 \leq x \leq e$ on the x-variables, the other constraining restrictions $x \in X$ on these variables are not included in the definition of Z. (This might be computationally advantageous in deriving the convex hull representation for Z; also, see Proposition 4.2 below for details on how this can be further exploited in the presence of dual-angular special structures.) Later, we will discuss a sequential scheme for partially generating this system as needed, but for now, assume that the entire description of Z is at hand.

Consider the problem

$$\mathbf{P'}: \quad \text{Minimize} \quad cx + dy \tag{4.3a}$$
$$\text{subject to} \quad Gx + Hy + Fw \geq f \tag{4.3b}$$
$$x \in X, x \in \{0, 1\}^n. \tag{4.3c}$$

Proposition 4.1. P′ has an optimal solution, and moreover, it is equivalent to P in the sense that if (x^*, y^*, w^*) solves P′, where (y^*, w^*) is an extreme point optimum to P′ for x fixed at x^*, then (x^*, y^*) solves P.

Proof. By our assumptions on P, the set Z given by (4.2) is bounded and P′ is feasible. Hence P′ has an optimum (x^*, y^*, w^*) where (y^*, w^*) satisfies the condition stated in the proposition. Moreover, since P′ is a relaxation of P, and its constraints imply $Ax + Dy \geq b$, $x \in X$, and the linear constraints describing $y \in Y$, it is sufficient to show that y^* satisfies the required binary restrictions on its subcomponents. From (4.2), any extreme point $(\bar{x}, \bar{y})$ of Z satisfies $\bar{y} \in Y$ (including the binary restrictions). Furthermore, if we define $Z(x^*) = Z \cap \{(x, y) : x = x^*\}$, then since $Z(x^*)$ is a face of Z, any extreme point $(x^*, \bar{y})$ of $Z(x^*)$ has $\bar{y} \in Y$ as well. Noting that $Z(x^*)$ defines the feasible region of P′ when x is fixed at x^*, and that (x^*, y^*) is a vertex of $Z(x^*)$, we have (x^*, y^*) is feasible, and therefore optimal, to P. This completes the proof. □

4.2.2 Specialized Modifications for Dual Angular Structures

Before proceeding further, it is instructive to comment on a modified derivation of the equivalent representation P′ when the original problem P exhibits a dual-angular structure (as in the special case of two-stage stochastic IPs). This analysis also lends further insights into the flexibility of constructing only partial convex hull representations in deriving an equivalent restatement of the problem to which Benders' decomposition method is applicable. Toward this

end, suppose that P possesses a dual-angular structure as revealed by the coefficient matrices given in the form

$$A \equiv \begin{bmatrix} A^1 \\ \vdots \\ A^S \end{bmatrix}, \; D = \begin{bmatrix} D^1 & & \\ & \ddots & \\ & & D^S \end{bmatrix}, \; b \equiv \begin{bmatrix} b^1 \\ \vdots \\ b^S \end{bmatrix}, \text{ and } d \equiv \begin{bmatrix} d^1 \\ \vdots \\ d^S \end{bmatrix}, \tag{4.4a}$$

where the vector y is also accordingly partitioned into components y^s, for $s = 1, \ldots, S$, with $y \in Y$ being replaced by

$$y^s \in Y_s \;\; \forall s = 1, \ldots, S. \tag{4.4b}$$

Here, for $s = 1, \ldots, S$, each Y_s is assumed to impose certain polyhedral restrictions on the (recourse) variables y^s (pertaining to scenario s), including binary restrictions on a subset of variables.

Now, let us define for each $s = 1, \ldots, S$,

$$Z_s = conv\{(x, y^s) : A^s x + D^s y^s \geq b^s, 0 \leq x \leq e, y^s \in Y_s\}, \tag{4.5a}$$

and let

$$Z' = \{(x, y) : (x, y^s) \in Z_s \text{ for each } s = 1, \ldots, S\}. \tag{4.5b}$$

Note that in general, $Z \subseteq Z'$, and that it is relatively easier to characterize Z' than it is to construct Z. Moreover, Z' retains the separability of the (recourse) variables y^s, $s = 1, \ldots, S$. The following result asserts that the equivalence of P′ and P as stated in Proposition 4.1 remains valid when Z is replaced by Z' under (4.4). In this context, similar to (4.2b), the construction (4.5) would yield P′ in the form given by (4.3) where the coefficient matrices in (4.3b) would possess the structure

$$G \equiv \begin{bmatrix} G^1 \\ \vdots \\ G^S \end{bmatrix}, \; H = \begin{bmatrix} H^1 & & \\ & \ddots & \\ & & H^S \end{bmatrix}, \; F = \begin{bmatrix} F^1 & & \\ & \ddots & \\ & & F^S \end{bmatrix}, \text{ and } f \equiv \begin{bmatrix} f^1 \\ \vdots \\ f^S \end{bmatrix} \tag{4.6}$$

and where the higher-dimensional vector w is also decomposed into the corresponding components w^s, $s = 1, \ldots, S$.

Proposition 4.2. Supposed that P has a dual angular structure as given by (4.4) , and let P′ be defined by replacing Z with the set Z' given by (4.5) and (4.6). Then P′ is equivalent to P in the sense asserted by Proposition 4.1.

Proof. Let (x^*, y^*, w^*) solve P′, where (y^*, w^*) is as stated in the proposition. Note from (4.6) that when we fix $x = x^*$, the problem P′ separates into *S* problems (by scenarios) given as follows:

$$\text{minimize } \{d^s y^s : (x^*, y^s) \in Z_s\}. \qquad (4.7)$$

Again, because (4.5) includes the hypercube restrictions $0 \le x \le e$, we have that $Z_s(x^*) \equiv Z_s \cap \{(x, y^s) : x = x^*\}$ is a face of Z_s, and therefore, its extreme points satisfy the required binary restrictions on y^s. Noting that $Z_s(x^*)$ is the feasible region of (4.7), this completes the proof. □

In what follows, for the sake of simplicity in notation and generality, we will assume that the set *Z* conforms with Z' whenever we have the dual angular structure exhibited by (4.4), with the system (4.2b) possessing the structure exhibited by (4.6). Hence, whenever we employ (4.2b), or develop lower-level RLT relaxations for the system $\{\cdot\}$ in (4.2a), we assume via Proposition 4.2 that in the presence of a dual-angular structure, we respectively have the structure (4.6), or that we correspondingly apply the lower-level RLT relaxation to the system $\{\cdot\}$ in (4.5a) for each $s = 1, \ldots, S$. We will periodically make some related comments in the sequel to re-emphasize this feature.

4.2.3 Derivation of a Benders' Approach for Problem P′

Assuming tentatively that we have explicitly constructed the equivalent formulation P′, we can apply Benders' partitioning to solve this problem as follows.

$$\underset{x \in X \cap \{0,1\}^n}{\text{Minimize}} \{cx + \text{ minimum } \{dy : Hy + Fw \ge f - Gx\}\} \qquad (4.8a)$$

$$\text{i.e., } \underset{x \in X \cap \{0,1\}^n}{\text{Minimize}} \{cx + \text{ maximum } \{\pi(f - Gx) : \pi H = d, \pi F = 0, \pi \ge 0\}\}. \qquad (4.8b)$$

Since we have assumed that the inner problem in (4.8) is feasible and bounded for any fixed $x \in X \cap \{0, 1\}^n$ letting

$$\{\pi^q, q = 1, ..., Q\} \equiv \text{vert}(\Lambda), \text{ where } \Lambda \equiv \{\pi : \pi H = d, \pi F = 0, \pi \ge 0\}, \qquad (4.9)$$

we obtain the following projected form of P′.

$$\text{Minimize} \quad z \qquad (4.10a)$$

$$\text{subject to} \quad z \ge cx + \pi^q(f - Gx) \text{ for } q = 1, ..., Q \qquad (4.10b)$$

$$x \in X \cap \{0,1\}^n. \qquad (4.10c)$$

Recall that (4.10) is the Benders' (overall) *master program*, and the inner minimization problem in (4.8a), or its dual in (4.8b), for any fixed *x* is referred to as the Benders' *subproblem*. This subproblem generates the *Benders' cuts* (4.10b) (along with upper bounds on the problem).

Note that in case we do not incorporate suitable artificial variable column(s) as needed to ensure that the inner problem in (4.8a) is feasible for any fixed $x \in X \cap \{0, 1\}^n$, we would also need to generate feasibility or extreme direction cuts in (4.10) of the following type, where δ^r, $r = 1, ..., R$, are extreme directions of the polyhedron Λ that is defined in (4.9).

$$\delta^r(f - Gx) \leq 0 \quad \text{for } r = 1, ..., R. \tag{4.11}$$

Remark 4.1. Note that in a practical implementation, we need not solve the relaxed Benders' master programs to optimality at each iteration. Rather, a branch-and-cut approach could be adopted, with the enumeration process set up only once, and with the current relaxed master program (RMP, say) being used to determine lower bounds, the subproblem (SP, say) providing upper bounds, and the (globally valid) Benders' cuts (4.10b) being generated as needed, i.e., whenever an incumbent solution to the current relaxed master program is found that has an objective value less than the present upper bound on the overall problem. Geoffrion and McBride (1978) and Adams and Sherali (1993) provide details for such an approach. Any actual application of Benders' method discussed here can be adapted to follow such a scheme. □

Example 4.1. As an illustration, consider the following example.

$$\begin{aligned}
\textbf{P}: \quad & \text{Minimize} \quad -x_1 - 2y_1 && (4.12a)\\
& \text{subject to} \quad -4x_1 - 3y_1 \geq -6 && (4.12b)\\
& \qquad\qquad (x_1, y_1) \text{ binary.} && (4.12c)
\end{aligned}$$

Figure 4.1 depicts the solution of this problem and identifies the set *Z*, along with the key facet that describes this set. By (4.2), this set *Z* is given by

$$Z = \text{conv}\{(x_1, y_1) : -4x_1 - 3y_1 \geq -6,\ 0 \leq x_1 \leq 1,\ y_1 \text{ binary}\}. \tag{4.13}$$

Since there is only one *y*-variable for this problem, we can develop the complete RLT representation of *Z* by multiplying each of the constraints in (4.13) by the two bound-factors associated with y_1. This yields the following equivalent Problem P′ as defined by (4.3):

$$\begin{aligned}
\text{Minimize} \quad & -x_1 - 2y_1 && (4.14a)\\
\text{subject to} \quad & 3y_1 - 4w \geq 0 && (4.14b)\\
& -4x_1 - 6y_1 + 4w \geq -6 && (4.14c)\\
& y_1 - w \geq 0 && (4.14d)\\
& x_1 - w \geq 0 && (4.14e)\\
& -x_1 - y_1 + w \geq -1 && (4.14f)\\
& w \geq 0 && (4.14g)\\
& x_1 \text{ binary.} && (4.14h)
\end{aligned}$$

Note that (4.14b) and (4.14c) are obtained by the RLT product of $-4x_1 - 3y_1 \geq -6$ with y_1 and $(1 - y_1)$, respectively, and (4.14d-g) are bound-factor RLT product constraints obtained via the

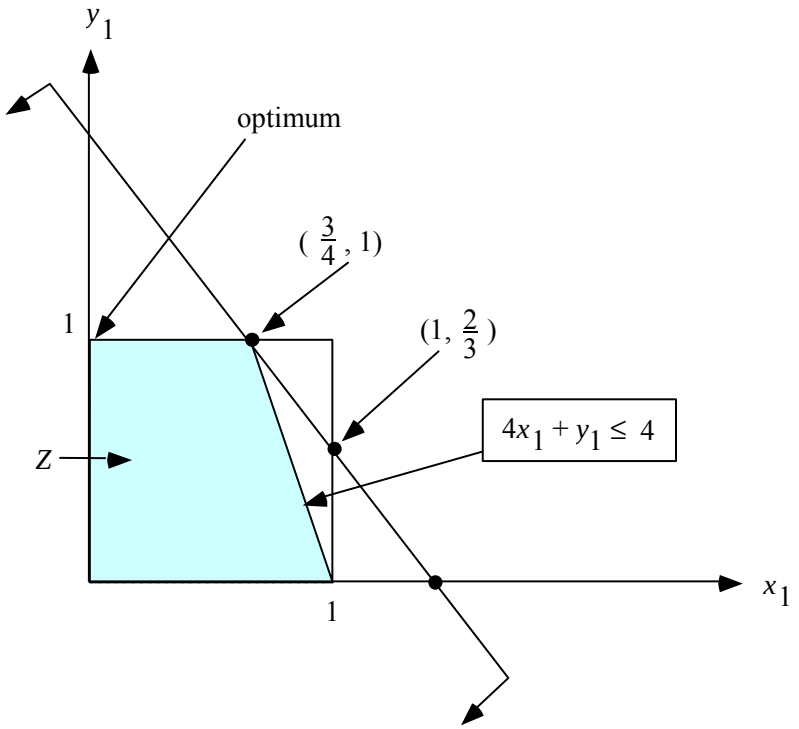

Figure 4.1. Illustration for Example 4.1.

products of the bounding inequalities $0 \leq x_1 \leq 1$ with y_1 and with $(1 - y_1)$. Observe that the surrogate of (4.14b) and (4.14f) according to

$$(3y_1 - 4w) + 4(-x_1 - y_1 + w + 1) \geq 0 \tag{4.15a}$$

produces the required key facet of Z identified in Figure 4.1 as

$$-4x_1 - y_1 \geq -4\,. \tag{4.15b}$$

In essence, by projecting the region of (4.14) onto the (x_1, y_1) space (only for illustrative purposes; this combinatorial step would not be performed in actual implementations), we get that (4.14) can equivalently be written as follows.

$$\begin{aligned} &\text{Minimize} && -x_1 - 2y_1 && (4.16a)\\ &\text{subject to} && -4x_1 - y_1 \geq -4 && (4.16b)\\ & && x_1 \text{ binary},\ 0 \leq y_1 \leq 1. && (4.16c) \end{aligned}$$

We could now apply Benders' partitioning to solve (4.14), which in essence, would be tantamount to applying this method to (4.16). For the sake of convenience, we apply it directly to (4.16) and obtain the decomposition

$$\underset{x_1 \in \{0,1\}}{\text{minimize}} \{-x_1 + \text{maximum } \{\pi_1(4x_1 - 4) - \pi_2 : -\pi_1 - \pi_2 \le -2, (\pi_1, \pi_2) \ge 0\}\}. \quad (4.17)$$

Noting that the extreme points of the inner maximization problem in (4.17) are $(\pi_1, \pi_2) = (2, 0)$ and (0, 2), and that (4.12) is feasible for any binary x_1, the complete Benders' master program is derived as follows.

$$\begin{aligned} &\text{Minimize} && z && (4.18a)\\ &\text{subject to} && z \ge 7x_1 - 8 && (4.18b)\\ &&& z \ge -x_1 - 2 && (4.18c)\\ &&& x_1 \text{ binary.} && (4.18d) \end{aligned}$$

The optimum to (4.18) (which would ultimately be generated via the usual process of applying Benders' methodology) is given by $x_1^* = 0$ and $z^* = -2$. Solving (4.16) (or (4.14)) with x_1 fixed at $x_1^* = 0$ yields $y_1^* = 1$ (and $w^* = 0$), with $v(x_1^*) = z^* = -2$. Since the relaxed master problem and subproblem have the same objective values, we have obtained an optimal solution to (4.12). □

4.3 Benders' Partitioning Using a Sequential Partial Convex Hull Constructive Process

The approach (4.8)-(4.10) is based on an *a priori* generation of the convex hull representation *Z* defined in (4.2) (or Z' defined by (4.5) and (4.6) under the structure (4.4)). If the size of the problem permits this construction (in particular, if we have few *y*-variables, or each partitioned constraint set in (4.5a) has a relatively simple structure), then this is a viable option, and leads to a usual application of Benders' decomposition as per Remark 4.1. Otherwise, we can generate a partial representation for *Z* as needed in a sequential convexification process, as discussed below. The following remark first highlights a key concept that is used in developing our proposed solution process.

Remark 4.2. Let $\bar{Y}$ denote the continuous relaxation of *Y*, and let $J^* = \{j : y_j \text{ is restricted to be binary in } Y\}$. For any $J \subseteq J^*$, define

$$Z^J = \text{conv}\{(x, y) : Ax + Dy \ge b,\ 0 \le x \le e,\ y \in \bar{Y},\ y_j \text{ binary } \forall j \in J\}. \quad (4.19)$$

Note that $Z^{\varnothing}$ along with $x \in X$ represents the continuous relaxation of (4.1), and $Z \equiv Z^{J^*}$. Since $Z \subseteq Z^J$ for each $J \subseteq J^*$, valid Benders' cuts can be derived from any such set Z^J. In fact, using the RLT process, we can construct a higher dimensional representation of Z^J for any $J \subseteq J^*$ that could be characterized as a surrogate of the representation (4.2b) for *Z* using suitable nonnegative multipliers (see Sherali and Adams 1990, 1994). Hence, Benders' cuts derived via the relaxation Z^J substituted in place of *Z* would correspond to cuts obtained via some feasible, though not necessarily extreme point, solution to Λ. Likewise, Benders' cuts derived via lower-

level RLT applications to $Z^{\varnothing}$ (levels less than $|J|$ for the case of Z^J) based on considering binariness on the variables y_j for $j \in J$, but not necessarily having constructed the entire convex hull representation Z^J, would be valid as well. Moreover, since the description of such a lower level representation can be obtained by surrogating the constraints of Z^J, and hence those of Z, the resulting cuts can also be viewed as implicitly obtained from feasible, nonextremal solutions to Λ. □

Based upon these insights, we now develop a finitely convergent method for solving Problem P, or Problem P′ via (4.8)-(4.10), by sequentially constructing a partial convex hull representation as needed. In this approach, for any fixed $\bar{x}$, the corresponding Benders' subproblem in (4.8b) that is reproduced below as

$$\textbf{SP}: \quad \text{maximize}\{\pi(f - G\bar{x}) : \pi H = d, \pi F = 0, \pi \geq 0\}, \tag{4.20}$$

is solved *implicitly* via an RLT-based or lift-and-project cutting plane approach (see Balas *et al*. (1993), and Sherali *et al*. (2000)). In the proposed method, we explicitly generate appropriate surrogated versions of *Z* as needed to derive valid RLT or lift-and-project cutting planes as needed for solving the subproblems. The key idea is that these generated cuts are characterized as functions of *x*, and can therefore be updated and re-used for subsequent subproblems based on the corresponding fixed value of *x*. Likewise, the Benders' cuts derived via the solution of the subproblems using such a cutting plane approach recognize these cuts as function of *x*, and are hence shown to be globally valid. This leads to an overall finitely convergent solution process.

Remark 4.3. To set ideas, let us first consider a preliminary rudimentary approach for solving Problem P′ via Benders' decomposition. This simple approach solves various restricted versions of the subproblems (4.20) (or relaxed versions of its dual) as follows. For the first instance of Problem SP, we let $k = 0$ and take $J_k = \varnothing$. Using $Z^{J_k} = Z^{\varnothing}$ as the current RLT representation within the inner minimization in (4.8a), we solve SP and generate the associated Benders' constraint for the relaxed master problem. At each subsequent visit to SP, if the current subproblem yields a binary *y*-solution, we use this solution to update the incumbent solution and to generate a Benders' cut. Otherwise, we increment *k* and take $J_k = J_{k-1} \cup \{j\}$ where y_j is restricted to be binary, but currently has a fractional value. We then construct Z^{J_k} as the *updated* RLT representation using the scheme described in Sherali and Adams (1990, 1994), solve SP, and generate the associated Benders' constraint for the relaxed master problem.

Note that this process creates a nested sequence of sets $J_0 \subseteq J_1 \subseteq J_2 \subseteq \cdots$ leading up to J^* in the worst case. Within a finite number of visits to SP, this procedure generates cuts based on *Z* via either a partial or full representation of this set, thereby deriving valid upper bounds from each such SP, and resulting in an overall finitely convergent algorithm based on the finiteness of the set $X \cap \{0, 1\}^n$. Alternatively, we could derive valid upper bounds from each subproblem by continuing to expand the set J_k at each iteration *k* to include fractionating *y*-variable indices until an integer feasible *y*-solution is obtained. This alternative is more in the conceptual spirit of the proposed approach as explained below. □

Clearly, the approach described in Remark 4.3 of sequentially generating approximations leading up to Z is computationally intensive because of the potentially exponential size of these (partial) convex hull representations. The procedure we propose below instead relies on generating cuts as needed to solve each subproblem SP based upon its fractionating variables, rather than generating full (partial) convex hull representations. More importantly, it characterizes these cuts in a fashion that permits them to be re-used in a suitably modified form for other subsequent subproblems. Furthermore, the cuts are generated in the original dimensional space, and previously generated cuts can be retained or deleted as desired.

As alluded above, the proposed method implicitly generates an appropriate surrogated representation of Z as needed for each individual SP via an RLT cutting plane approach as follows. Suppose that we are solving SP for a given $\bar{x}$. In essence, we wish to solve

$$v(\bar{x}) = c\bar{x} + \text{ minimum } \{dy : Dy \geq b - A\bar{x}, y \in Y\} \tag{4.21}$$

but we conceive solving this (albeit implicitly) via the problem

$$v(\bar{x}) = c\bar{x} + \text{ minimum } \{dy : Hy + Fw \geq f - G\bar{x}\} \tag{4.22}$$

from (4.8a), so that we can derive a valid Benders' cut. (Note that in the presence of a dual angular structure, (4.22) would yield a separable system as per (4.6).) Now suppose that we adopt a sequential convexification lift-and-project type of cutting plane scheme to solve (4.21), using RLT cuts based on enforcing binariness on one variable as in Balas *et al*. (1993), or on multiple variables as in Sherali *et al.* (2000). (See Section 4.4 for details on the finite convergence of such a cutting plane algorithm.) Suppose that we obtain the final cut-enhanced problem that solves (4.21) as given by (4.23) below, where (4.23c) represents the continuous relaxation $\bar{Y}$, and where (4.23d) represents the set of RLT or lift-and-project cuts generated.

$$v(\bar{x}) = c\bar{x} + \text{ minimum } dy \tag{4.23a}$$
$$\text{subject to } Dy \geq b - A\bar{x} \tag{4.23b}$$
$$\Gamma y \geq \gamma \tag{4.23c}$$
$$\alpha_t y \geq \beta_t - \phi_t \bar{x} \text{ for } t = 1, ..., T. \tag{4.23d}$$

Each of the cuts $t = 1, ..., T$ in (4.23d) is derived via the following steps.

Step 1. Based on some current fractional solution $\bar{y}$, generate an appropriate RLT enhancement of Z^{ϕ} given as follows (by enforcing binariness on one or more variables – see Section 4.4, and in particular, Remark 4.5 given later for some additional details):

$$G_t x + H_t y + F_t w \geq f_t. \tag{4.24}$$

(In the presence of dual-angularity, this system would have a structure similar to that in (4.6).)

Step 2. Fix $x = \bar{x}$, and determine dual multipliers $\pi_t \geq 0$ for (4.24) that solves the following separation problem, where e is a conformable vector of ones, and where (4.25c) is a

normalization constraint (that can be imposed separably in the context of dual-angular structures).

$$\text{Minimize} \quad \pi_t(H_t\bar{y}) - \pi_t(f_t - G_t\bar{x}) \tag{4.25a}$$
$$\text{subject to} \quad \pi_t F_t = 0 \tag{4.25b}$$
$$e \cdot \pi_t = 1 \tag{4.25c}$$
$$\pi_t \geq 0. \tag{4.25d}$$

Note that by virtue of the RLT process, an appropriate representation (4.24) can be generated that yields a negative value in (4.24). Let $\tilde{\pi}_t$ be the solution of (4.24). Then we have that

$$\tilde{\pi}_t H_t y \geq \tilde{\pi}_t (f_t - G_t \bar{x}) \tag{4.26}$$

deletes the current fractional solution $\bar{y}$. Furthermore, with the substitution

$$\alpha_t \equiv \tilde{\pi}_t H_t, \beta_t \equiv \tilde{\pi}_t f_t, \text{and } \phi_t \equiv \tilde{\pi}_t G_t, \tag{4.27}$$

we have that (4.26) is of the form (4.23d).

The final representation (4.23) can be used to derive a valid Benders' cut, as shown in Proposition 4.2. This leads to a finitely convergent algorithm, as demonstrated subsequently in Proposition 4.3. Following this, we will comment on the re-use of previously generated cuts for new subproblems (4.21)-(4.23) solved for revised values for $\bar{x}$.

Proposition 4.2. Consider Problem (4.23), and let ψ_1, ψ_2, and $(\psi_{3t}, t = 1, ..., T)$ be the optimal nonnegative dual multipliers obtained for the constraints (4.23b), (4.23c), and (4.23d), respectively. Then, noting (4.27), the inequality

$$z \geq cx + \psi_1(b - Ax) + \psi_2\gamma + \sum_{t=1}^{T} \psi_{3t}(\beta_t - \phi_t x) \tag{4.28}$$

is a valid Benders' cut.

Proof. Consider the system (4.3b) that is derived from (4.2). Since the original constraints in (4.2a) are implied by (4.2b) via a suitable surrogation process, and noting the definition of (4.23c), there exist nonnegative surrogate multiplier matrices τ_1 and τ_2 such that

$$\tau_1[G, H, F] = [A, D, 0], \text{ with } \tau_1 f \geq b, \text{ and} \tag{4.29}$$
$$\tau_2[G, H, F] = [0, \Gamma, 0], \text{ with } \tau_2 f \geq \gamma. \tag{4.30}$$

Similarly, since any lower-level or partial RLT application such as (4.24) is implied by (4.3b) via a surrogation process, there exist nonnegative surrogate multiplier matrices $\tau_{3t}, t = 1, ..., T$, such that

$$\tau_{3t}[G, H, F] = [G_t, H_t, F_t], \text{ with } \tau_{3t} f \geq f_t, \ \forall \ t = 1, ..., T. \tag{4.31}$$

Now, let us define

$$\bar{\pi} = \psi_1 \tau_1 + \psi_2 \tau_2 + \sum_{t=1}^{T} \psi_{3t} \tilde{\pi}_t \tau_{3t} \tag{4.32}$$

where $\tilde{\pi}_t$ is obtained as an optimum to (4.25) and satisfies (4.26). Note that $\bar{\pi} \geq 0$, and from (4.26), (4.29) – (4.32), we get

$$\bar{\pi} H = \psi_1 D + \psi_2 \Gamma + \sum_{t=1}^{T} \psi_{3t} \tilde{\pi}_t H_t$$

$$\text{i.e.} \quad \bar{\pi} H = \psi_1 D + \psi_2 \Gamma + \sum_{t=1}^{T} \psi_{3t} \alpha_t = d \tag{4.33}$$

via duality in (4.23). Moreover, we have from (4.25b), (4.29) – (4.32) that

$$\bar{\pi} F = \psi_1 (0) + \psi_2 (0) + \sum_{t=1}^{T} \psi_{3t} \tilde{\pi}_t F_t = 0. \tag{4.34}$$

Hence, $\bar{\pi} \in \Lambda$ as defined in (4.9), and so the constraint

$$z \geq cx + \bar{\pi}(f - Gx) \tag{4.35a}$$

is a valid Benders' inequality. But from (4.26), (4.29) – (4.32), we have,

$$\bar{\pi}(f - Gx) \geq \psi_1 (b - Ax) + \psi_2 \gamma + \sum_{t=1}^{T} \psi_{3t} \tilde{\pi}_t (f_t - G_t x)$$

$$\text{i.e.} \quad \bar{\pi}(f - Gx) \geq \psi_1 (b - Ax) + \psi_2 \gamma + \sum_{t=1}^{T} \psi_{3t} (\beta_t - \phi_t x). \tag{4.35b}$$

Noting (4.35a) and (4.35b), we have that (4.28) is a valid Benders' cut, and this completes the proof. □

Remark 4.4. Note that the key insight above is that although the right-hand sides in (4.23) are real numbers in the process of solving the underlying subproblem, the Benders' inequality generated from its optimal dual solution via (4.28) needs to recognize the right-hand sides of both (4.23b) and (4.23d) as *functions* of *x*, much as in the usual Benders approach. In particular, we need to store the constant β_t and the vector ϕ_t for each cut $t = 1, ..., T$ in (4.23d). Note that the parent matrices or RLT representations that generated these cuts need not be stored. Furthermore, because of the global validity of the inequality

$$\alpha_t y \geq \beta_t - \phi_t x \tag{4.36}$$

for *any x* by virtue of (4.24) and the surrogation of the type in (4.26), we can impose the previously generated cuts of type (4.23d) in any subsequent subproblem solution, *simply by modifying its right-hand side according to the current* $\bar{x}$. This re-use opportunity can greatly benefit the solution procedure. Section 4.4 further addresses the finite convergence issues related to such a cutting plane process applied to any given subproblem. □

Despite the fact that we might not be generating extreme points of Λ in the cuts (4.28), the following result establishes finite convergence of the overall algorithm, assuming that each subproblem is solved finitely (as discussed in Section 4.4 below).

Proposition 4.3. Suppose that we implement Benders' algorithm in the traditional fashion as follows. At each iteration, we solve the relaxed master program (4.10), where the Benders' cuts (4.10b) are replaced by the current set of cuts of type (4.28). Let $(\bar{z}, \bar{x})$ be an optimal solution to this relaxed master program, where $\bar{x} \in X \cap \{0, 1\}^n$. Next, we solve the subproblem (4.23) to determine the value $v(\bar{x})$ of Problem *P* when *x* is fixed at $\bar{x}$, and accordingly, either terminate if $\bar{z} \geq v(\bar{x})$ (equivalently, $\bar{z} = v(\bar{x})$), or else, generate a Benders' cut (4.28) if $\bar{z} < v(\bar{x})$. Then, this process will converge finitely with an optimum for Problem P.

Proof. Note that by the validity of (4.28) in Proposition 4.2, the result holds true if we show that we will finitely obtain the termination criterion $\bar{z} \geq v(\bar{x})$. Observe that by duality in (4.23), the right-hand side of (4.28) evaluated at $x = \bar{x}$ yields $v(\bar{x})$. Hence, whenever a previous $\bar{x}$ is regenerated by the master program, the termination criterion would hold true. Since there are only a finite number of solutions in $X \cap \{0, 1\}^n$, this must occur finitely, and the proof is complete. □

As mentioned previously, an actual implementation would follow Remark 4.1. Figure 4.2 provides a flow-chart for such a process.

Example 4.2. Consider the problem of Example 4.1. To illustrate the concept of the proposed approach, suppose that we have a relaxed master program RMP that currently has the Benders' inequality (4.18c), but not (4.18b). This problem yields the solution $\bar{x}_1 = 1$ and $\bar{z} = -3$. We now

solve for $v(\bar{x}_1)$ via the following problem, using a cutting plane process in the spirit of (4.23).

$$v(\bar{x}_1) = -\bar{x}_1 + \text{ minimum } \{-2y_1 : -3y_1 \geq 4\bar{x}_1 - 6,\ y_1 \text{ binary}\}. \tag{4.37}$$

The continuous optimum for (4.37) is $\bar{y}_1 = 2 / 3$. At Step 1 of the cut generation process, let the RLT constraints (4.24) be given by (4.14b) – (4.14g) as in Balas *et al.*'s (1993) lift-and-project scheme. The corresponding separation problem (4.25) at Step 2 is given as follows, where (for "*t*"= 1), $\pi_{11}, ..., \pi_{16}$ denote the surrogate multipliers with respect to the constraints (4.14b) – (4.14g), respectively.

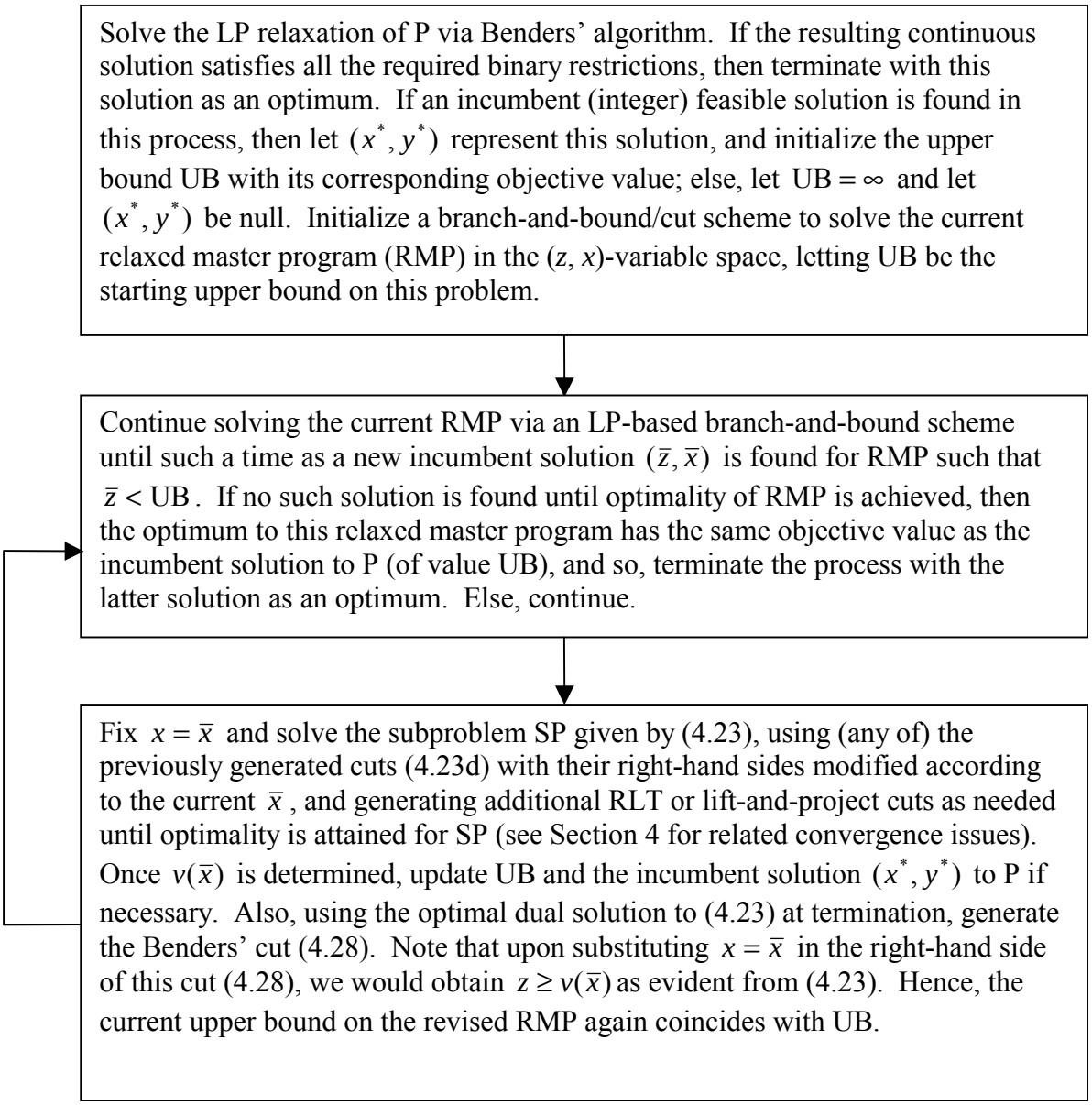

Figure 4.2. Flow-chart of an Implementation for the Proposed Benders' Algorithm.

$$\begin{aligned} \text{Minimize} \quad & 2\pi_{11} - 2\pi_{12} + \frac{2}{3}\pi_{13} + \pi_{14} - \frac{2}{3}\pi_{15} \\ \text{subject to} \quad & -4\pi_{11} + 4\pi_{12} - \pi_{13} - \pi_{14} + \pi_{15} + \pi_{16} = 0 \\ & \pi_{11} + \pi_{12} + \pi_{13} + \pi_{14} + \pi_{15} + \pi_{16} = 1 \\ & (\pi_{11}, ..., \pi_{16}) \geq 0 . \end{aligned}$$

This problem yields the solution $\tilde{\pi}_{11} = \frac{1}{5}$, $\tilde{\pi}_{15} = \frac{4}{5}$, $\tilde{\pi}_{12} = \tilde{\pi}_{13} = \tilde{\pi}_{14} = \tilde{\pi}_{16} = 0$, with an objective value of –2/15, thereby indicating that a cut is generated. From (4.27), this cut yields

$$\alpha_1 = -\frac{1}{5}, \ \beta_1 = -\frac{4}{5}, \quad \text{and } \phi_1 = -\frac{4}{5}. \tag{4.38}$$

The globally valid cut of type (4.32) is then given via (4.26) as

$$-(1/5)y_1 \geq (-4/5) + (4/5)x_1 \tag{4.39}$$

which corresponds to the facet of Z depicted in Figure 4.1. The particular cut (4.23d) that is incorporated within (4.37) is obtained by fixing $x_1 = \bar{x}_1 \equiv 1$ in (4.39). This yields the inequality $-y_1 \geq 0$, thereby producing (4.23) as

$$\begin{aligned} v(\bar{x}_1) = -\bar{x}_1 + \quad & \text{minimum} \quad -2y_1 && \text{(4.40a)} \\ & \text{subject to} \quad -3y_1 \geq 4\bar{x}_1 - 6 \equiv -2 && \text{(4.40b)} \\ & -y_1 \geq -4 + 4\bar{x}_1 = 0 && \text{(4.40c)} \\ & 0 \leq y_1 \leq 1 . && \text{(4.40d)} \end{aligned}$$

The optimal solution is given by $\bar{y}_1 = 0$, with the dual multipliers with respect to (4.40b,d) being zeroes and with respect to (4.40c) being 2, yielding $v(\bar{x}_1) = -1 > \bar{z} = -3$. Hence, we generate the Benders' cut (4.28) as

$$\begin{aligned} & z \geq -x_1 + 2(-4 + 4x_1) \\ \text{i.e.} \quad & z \geq 7x_1 - 8. \end{aligned} \tag{4.41}$$

This produces the revised relaxed Benders' master program given by (4.18) as in Example 4.1, which results in an optimal solution being detected as before.

4.4 Finite Convergence of a Cutting Plane Procedure for Solving Subproblems

In the foregoing section, we have developed a Benders partitioning approach for Problem P of the type (4.1) based on the use of a suitable cutting plane approach for solving each subproblem (4.21) via (4.23). The cuts derived via (4.24) – (4.27) were generated to be directly

valid for *Z* itself, but were then imposed on the current subproblem by fixing $x = \bar{x}$, where $\bar{x}$ corresponds to the given first-stage decision for the present subproblem. This not only permitted their re-use for other subproblems, but also enabled the derivation of the required Benders' cuts that induced an overall finitely convergent process. In this section, we now address the issue of designing a finitely convergent cutting plane procedure of this type for computing $v(\bar{x})$ defined in (4.21) via (4.23). (As alluded variously in the foregoing section, in the context of dual-angular structures, the separability of (4.21) and the partial convex hull requirement stipulated by Proposition 4.2 can be exploited below with obvious modifications.)

Note that in practice, one could use a variety of lift-and-project or RLT cuts as presented in Balas *et al*. (1993) and Sherali *et al*. (2000) to implement (4.23). However, in order to ensure that such a process finitely solves the underlying 0-1 mixed-integer program, some care needs to be exercised while sequentially constructing the (partial) convex hull representation that is necessary to solve this problem. As in Balas *et al*.'s (1993) lift-and-project cutting plane algorithm, we rely on Jeroslow's (1980) cutting plane game concept for facial disjunctive programs. (Note that (4.21), and likewise Problem P given by (4.1), is a *facial disjunctive program* in that it involves the conjunction of the disjunctions that $y_j \le 0$ or $y_j \ge 1$ (in concert with $0 \le y_j \le 1$) for each $j = 1, \ldots, p$, along with the facial property that the intersection of either of these disjunctive restrictions with the continuous feasible region of (4.21) defines a face of this region.) However, there is one important variation in the standard process that we need to account for, in that we are generating cuts that are valid for *Z* of Equation (4.2) in our context, and then imposing these cuts in (4.23) by fixing $x = \bar{x}$. As Proposition 4.4 below establishes, the key element that validates this variation is that for any binary feasible solution $\bar{x}$, if we denote the convex hull of the feasible region of the subproblem (4.21) as $Z(\bar{x})$ and view this region in the form

$$Z(\bar{x}) = \text{conv}\{(x, y) : Dy \ge b - Ax,\ y \in Y, \text{ and } x = \bar{x}\}, \tag{4.42a}$$

then we effectively have that

$$Z(\bar{x}) = Z \cap \{(x, y) : x = \bar{x}\} \tag{4.42b}$$

since the right-hand side in (4.42b) defines a face of *Z* because *Z* includes the restrictions $0 \le x \le e$ in its definition. Consequently, we can derive the required description of the facial structure of $Z(\bar{x})$ given by (4.42a) that is necessary for solving the subproblem (4.21) by generating appropriate valid inequalities for *Z*, and then restricting $x = \bar{x}$. Figure 4.3 provides a flow-chart for such a cutting plane process in the context of lift-and-project cuts of Balas *et al*. (1993), and Remark 4.5 below provides comments on using more general RLT cuts along with some implementation suggestions. The following result establishes finite convergence of the procedure presented in Figure 4.3.

Proposition 4.4. The cutting plane procedure of Figure 4.3 finitely solves the subproblem (4.21) via (4.23), yielding a family of valid inequalities (4.23d) that can be re-used for any other subproblem by revising the corresponding first-stage decision $\bar{x}$.

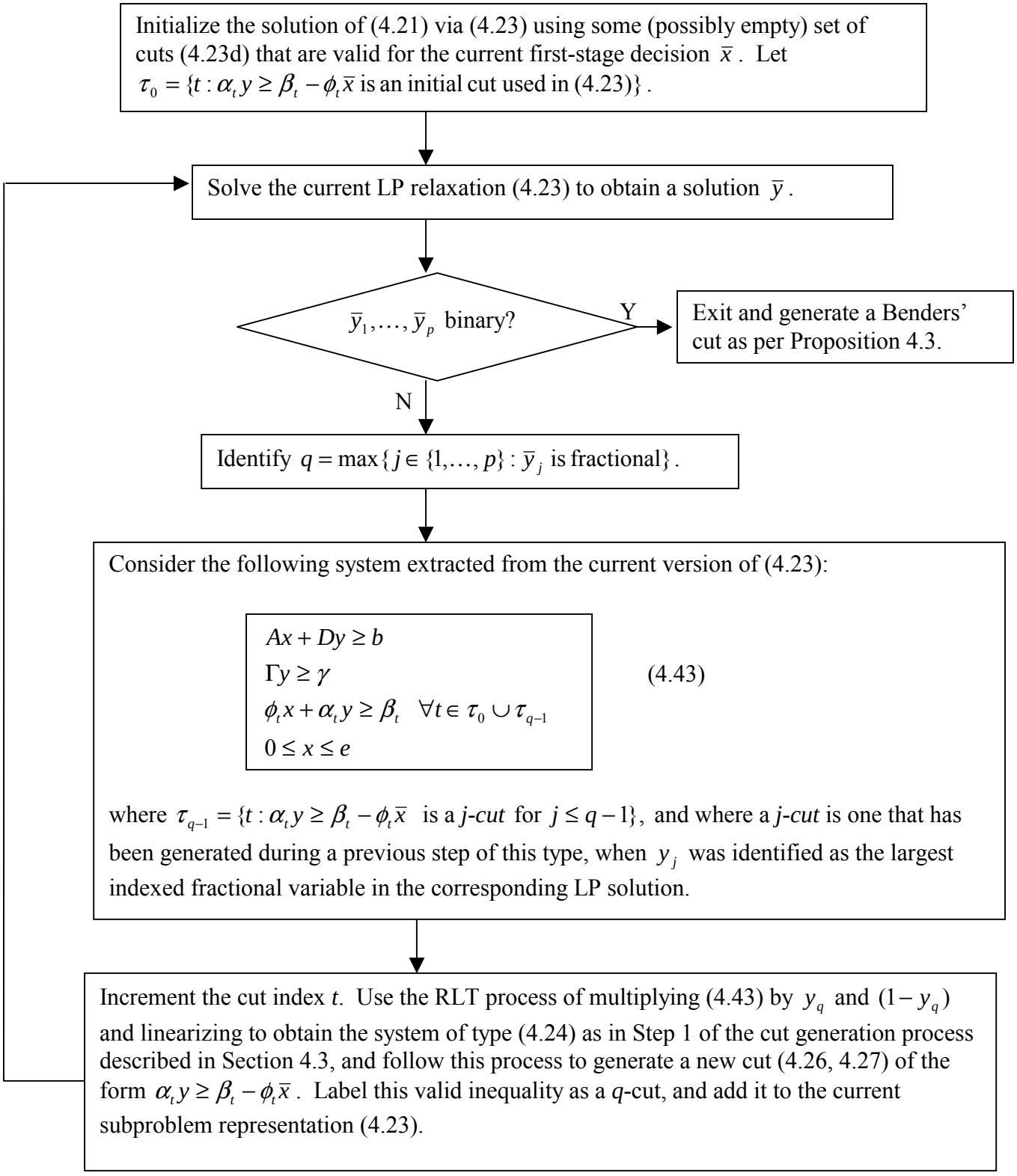

Figure 4.3. Cutting Plane Procedure for Solving any Subproblem.

Proof. First of all, note that the cut generation process of Section 4.3 is based on deriving valid inequalities for relaxations of *Z* of the type (4.24), obtained by applying RLT while enforcing binariness on a single variable y_q to some system of type (4.43) (see Figure 4.3). Hence, inductively, each inequality generated of the form $\phi_t x + \alpha_t y \geq \beta_t$ is valid for *Z*, and therefore, can be imposed for any subproblem by fixing the *x*-variables to the corresponding first-stage decision values.

Next, let us view the subproblem (4.21) that is to be solved in the following form (augmented with an initial set of valid cuts), where *x* is declared to be a variable, but the parameter *M* is assumed to be sufficiently large so that we necessarily have $x = \bar{x}$ at optimality in this problem (4.44), as well as at its LP relaxation.

$$
\begin{aligned}
v(\bar{x}) = c\bar{x} + \text{minimum } & dy + M[\sum_{j:\bar{x}_j=0} x_j + \sum_{j:\bar{x}_j=1}(1-x_j)] \\
\text{subject to} \quad & Ax + Dy \geq b \\
& \Gamma y \geq \gamma \\
& \phi_t x + \alpha_t y \geq \beta_t \quad \forall t \in \tau_0 \\
& 0 \leq x \leq e,\ y_i \in \{0,1\}\ \forall i = 1,\ldots,p.
\end{aligned}
\tag{4.44}
$$

Now, suppose that we apply the lift-and-project cutting plane procedure described in Balas *et al*. (1993) to Problem (4.44). By making *M* sufficiently large, we can assume that each LP relaxation solved in the (finite) iterative process will continue to yield $x = \bar{x}$, so that each of these LP relaxations can effectively be solved via (4.23) by fixing $x = \bar{x}$ as in the flow-chart of Figure 4.3. Note that if $\bar{y}$ is a resulting extreme point solution, then $(\bar{x}, \bar{y})$ is a vertex of the continuous relaxation to (4.44) augmented with any additional cuts, since $x = \bar{x}$ describes a face of this latter region. Consequently, the procedure of Figure 4.3 is precisely the lift-and-project cutting plane scheme that is proven in Theorem 3.1 of Balas *et al*. (1993) to converge finitely as applied to Problem (4.44), and this completes the proof. □

Remark 4.5. Note that the lift-and-project cutting plane procedure of Balas *et al*. (1993) is predicated on generating cuts based on enforcing binariness on 0-1 variables one at a time. A more general RLT process of Sherali and Adams (1990, 1994) could be used to devise a cut generation scheme that likewise enforces binariness on more than one variable at a time. In such a process, the 0-1 variables can be grouped into batches containing one or more variables per batch, perhaps based on the initial LP solution. A similar scheme as in Figure 4.3 could then be followed, in which the relaxation (4.24) of *Z* is generated by applying RLT while enforcing binariness on the highest indexed batch of variables that contains some fractionating variable(s), to a system (4.43) that contains cuts generated previously for lower-indexed batches. The convergence of such a problem would follow from Jeroslow's (1980) cutting plane game as in Proposition 4.4. Of course, the advantage of considering batches of cardinality one is that the associated separation problems are relatively easier to solve. However, Sherali *et al*. (2000) have recently demonstrated how stronger RLT cuts accruing from the simultaneous consideration of multiple variables can be efficiently generated by using suitably restricted projections of the

associated dual cone. Furthermore, in practical implementations, one could employ all the retained cuts in (4.43) of the procedure of Figure 4.3 or consider the deletion of cuts based on certain filtering criteria as well. In addition, as alluded in Remarks 4.2 and 4.3, and as evident from the foregoing discussion, we could prematurely abort the solution of any particular subproblem for a given $x = \bar{x}$ via the described cutting plane scheme, and generate a corresponding valid Benders' cut. This might entail regenerating a previous $\bar{x}$, while not yet having solved Problem P. However, so long as complete subproblem solutions are enforced after a finite number of iterations or even finitely often, we would obtain an overall finitely convergent process. Investigations of this type require extensive computational experimentations that we hope to pursue in future research. □

4.5 Summary and Conclusions

In this chapter, we have modified Benders' decomposition method using RLT and lift-and-project cuts to develop a new method for solving discrete optimization problems that yield 0-1 mixed-integer subproblems, such as those encountered in stochastic programs with integer recourse. Viewing the problem *implicitly* in the light of a suitably defined convex hull representation, with appropriate modifications when the original problem exhibits a dual-angular structure, we have demonstrated how cutting planes could be generated to derive a partial description of this convex hull representation as needed in order to devise a finitely convergent solution procedure. Importantly, the classes of cuts used in the subproblems were derived in terms of *functions* of the first-stage x-variables, enabling them to be re-used in subsequent subproblems simply by revising them according to the corresponding x-solutions. Additionally, globally valid Benders' cuts were obtained by recognizing these cuts as functions of the first-stage variables. The ability to re-use cutting planes from one subproblem to the next in this fashion is useful from the viewpoint of potentially reducing the computational effort required to solve the discrete subproblems, while providing globally valid Benders' cuts that enhance the lower-bounding mechanism via the relaxed master program. The focus of this chapter has been on developing the theory for such a modified Benders' approach. In order to gauge the effectiveness of the proposed technique, a variety of computational test, particularly in the context of stochastic programs with integer recourse, should be conducted, and we propose this task for future research.

Chapter 5: Improved MIP Models and Algorithms for the Facility Layout Problem

As discussed in Chapter 2, the facility layout problem is a challenging optimization problem that arises in the context of many practical applications. Given the dimensions of a rectangular building, the basic problem is to design a floor-plan comprised of rectangular departments in order to minimize the total amount of travel (distance times the number of trips) between the departments. The difficult nature of this optimization problem has led to a number of construction and improvement heuristics, but very little research has focused on directly using MIP formulations to solve the problem optimally. One notable exception is the paper by Meller *et al*. (1999) that examines the MIP formulation originally proposed by Montreuil (1990) and discusses several enhancements to improve and strengthen the model representation. While the results presented by Meller *et al*. are promising, we describe in this chapter a series of significant enhancements to the MIP model that lead to more accurate solutions, as well as decreased solution effort.

The remainder of this chapter is organized as follows. Section 5.1 provides a comprehensive overview of the MIP model (FLP2+) that was proposed by Meller *et al*., and Section 5.2 presents computational results obtained using this model, as well as an experimental design for evaluating our proposed enhancements. Sections 5.3 through 5.6 each outline a specific enhancement to the basic FLP model and discuss related computational results. Section 5.3 addresses a new formulation for the nonlinear area constraints, Section 5.4 develops special symmetry breaking valid inequalities, and Section 5.5 analyzes the effect of using several other classes of valid inequalities. As a final enhancement, Section 5.6 discusses two new techniques for modeling the disjunctive relationships that prohibit departments from overlapping and explores the derivation of partial convex hull representations and valid inequalities from their structure. After evaluating several combinations of proposed enhancements, we narrow our focus to two promising formulations, which are used to solve three more challenging problem instances in Section 5.7. We provide conclusions and directions for future research in Section 5.8.

5.1 Problem Overview

Given a set of departments $\{1,\ldots,n\}$, the facility layout problem seeks to determine a non-overlapping arrangement of the departments that minimizes the total travel between departments as specified by $\sum_{i}\sum_{j<i}\sum_{s} f_{ij} d_{ij}^{s}$, where the parameter f_{ij} is a given amount of flow between departments *i* and *j*, and where the variable d_{ij}^{s} represents the rectilinear distance between the respective centroids of departments *i* and *j* in the direction *s*. For notational convenience, all dimensions, distance measures, and locations that are specified in terms of their

horizontal and vertical components are denoted by the superscripts *x* and *y*, respectively. The overall building is assumed to be a rectangle of size $L^x \times L^y$, and each department *i* is required to be a rectangle with target area a_i. For each department *i*, the parameter α_i (≥ 1), known as the *aspect ratio*, delineates the maximum permissible ratio between the longest and shortest sides; i.e., $\max_{s=x,y}\{\ell_i^s\} / \min_{s=x,y}\{\ell_i^s\} \leq \alpha_i$, $\forall i$. There are four decision variables for each department *i*, namely, the half-length and half-width (ℓ_i^x, ℓ_i^y), and the centroidal location (c_i^x, c_i^y). In order to guarantee that each department *i* is contained within the building, we impose the bounds $\ell_i^s \leq c_i^s \leq L^s - \ell_i^s$ for each *s* on the placement of its centroid. In addition, we denote any valid implied upper and lower bounds on ℓ_i^s as ℓb_i^s and ub_i^s, respectively. (We note that, although not computationally effective, some previous formulations for the facility layout problem have taken $\ell b_i^s = \ell b_i \ \forall s$. In Section 5.2.1, we derive tight values for ℓb_i^s and ub_i^s by considering the bounds in direction *x* and *y* separately.) A generic version for the facility location problem can then be stated as follows.

FLP: Minimize $\sum_i \sum_{<j} f_{ij}(d_{ij}^x + d_{ij}^y)$ (5.1a)

subject to Departmental Area Constraints (5.1b)

Overlap Prevention (or Separation) Constraints (5.1c)

$d_{ij}^s = |c_i^s - c_j^s| \quad \forall\ i < j, s$ (5.1d)

$\ell_i^s \leq c_i^s \leq L^s - \ell_i^s \quad \forall\ i, s$ (5.1e)

$\ell b_i^s \leq \ell_i^s \leq ub_i^s \quad \forall\ i, s.$ (5.1f)

In addition to the constraints listed in (5.1), appropriate restrictions can be added to accommodate the case where some departments are given fixed locations or when certain areas of the building are not permitted to be occupied by any department.

5.2.1 The FLP2 Model

Throughout the remainder of this chapter, we will propose enhancements to the best existing MIP formulation of the facility layout problem that was presented as FLP2 in Meller *et al*. (1999). Before proceeding with this endeavor, we first review the notation of the FLP2 model. For ease in notation, we define *P* as the set of department pairs having positive flow interaction; that is, $P = \{(i, j), i < j : f_{ij} > 0\}$. We also denote the set *F* as the departments with fixed size and location, while its complement $\overline{F}$ contains all of the departments with variable size and location. (In the present context, we do not consider departments that are fixed with respect only size or location, although this modification could also be accommodated in a straightforward manner.) Using the aspect ratios for each department, Meller *et al*. derive implied upper and lower bounds on the half-sides of each non-fixed department, which we denote as $\overline{ub_i}$ and $\overline{\ell b_i}$, respectively. These bounds are given by

$$\overline{ub_i} = \min\{\sqrt{a_i \alpha_i}, \max_s\{L^s\}\}/2 \quad \text{and} \quad \overline{\ell b_i} = a_i/(4ub_i)\ \forall i \in \overline{F},$$

and are used to constrain the department dimensions via the restrictions

$$\overline{\ell b_i} \le \ell_i^s \le \min\{\overline{ub_i}, L^s/2\},\ \forall i, s.$$

Accordingly, the centroid of each department $i \in \overline{F}$ is bounded as $\overline{\ell b_i} \le c_i^s \le L^s - \overline{\ell b_i}\ \ \forall s$. (We note here that Meller *et al*. use the same lower and upper bounding values for the half-width and half-length of department *i* (i.e., $\ell b_i^s = \overline{\ell b_i}$ and $ub_i^s = \overline{ub_i}\ \forall s = x, y$), but we will develop tighter bounds in Section 5.3 by considering each dimensional separately.)

One of the major difficulties in modeling the MIP facility problem is to derive a suitable approximation for the nonlinear area constraints, $4\ell_i^x \ell_i^y = a_i$, for each department $i \in \overline{F}$. Toward this end, Meller *et al*. propose an approximation that uses a parameter f (empirically taken to be 0.95 in their computations), and the maximum departmental half-side denoted by $\ell_i^{\max} = \max_{s=x,y}\{\ell_i^s\}$. In order to ensure that departments do not overlap, Meller *et al*. propose several disjunctive statements that are linearized using binary variables, z_{ij}^s, to indicate relative locations, where $z_{ij}^s = 1$ if department *i* is forced to precede department *j* in the direction *s*. Given this notation, Problem FLP2 of Meller *et al*. can be stated as follows, where throughout the formulation, *i* and *j* represent the indices for the *n* departments, and *s* is an indicator representing the two directions (*x* and *y*).

FLP2: Minimize $\sum_{(i,j)\in P}\sum_{s=x,y} f_{ij} d_{ij}^s$ (5.2a)

subject to

$4(\ell_i^x + \ell_i^y) \ge 3\sqrt{a_i} + f \times 2\ell_i^{\max} \quad \forall i \in \overline{F}$ (5.2b)

$\ell_i^{\max} \ge \ell_i^s, \quad \forall i \in \overline{F}, s$ (5.2c)

$\sum_{s=x}^{y}(z_{ij}^s + z_{ji}^s) \ge 1 \ \ \forall i < j, s$ (5.2d)

$z_{ij}^s + z_{ji}^s \le 1 \ \ \forall i < j, s$ (5.2e)

$c_i^s + \ell_i^s \le c_j^s - \ell_j^s + L^s(1 - z_{ij}^s) \ \ \forall i \ne j, s$ (5.2f)

$d_{ij}^s = \left|c_i^s - c_j^s\right| \quad \forall (i,j) \in P, \forall s$ (5.2g)

$\ell_i^s \le c_i^s \le L^s - \ell_i^s \quad \forall i \in \overline{F}, s$ (5.2h)

$\overline{\ell b_i} \le \ell_i^s \le \min\{\overline{ub_i}, L^s/2\}, \forall i \in \overline{F}, s$ (5.2i)

$c_i^s \ge 0 \ \ \forall i, s$ (5.2j)

$d_{ij}^s \ge 0 \ \ \forall (i,j) \in P, s$ (5.2k)

$z_{ij}^s \in \{0,1\} \ \ \forall i \ne j, s$ (5.2l)

(c_i^s, ℓ_i^s) fixed $\forall i \in F, s$. (5.2m)

In terms of the notation of problem (5.1), constraints (5.2b) and (5.2c) capture the departmental area requirements, while (5.2d-f, l) prevent departmental overlaps. Specifically,

the constraints (5.2b) approximate the nonlinear area constraints ($4\ell_i^x \ell_i^y = a_i \; \forall i$) by forcing the actual perimeter of each department, given as the left-hand side of (5.2b), to be at least equal to an empirically determined function of a_i and $\ell_i^{\max}$ (as defined by (5.2c)) that exceeds the perimeter $4\sqrt{a_i}$ of a square department having area a_i. The motivation behind this approach is to make the area restrictions more faithful as departments become more non-square. Constraints (5.2d) and (5.2e), together with constraints (5.2f), force each pair of departments to be separated in at least one direction, and hence prevent departments from overlapping. Meller *et al*. demonstrated that the constraints (5.2e) are unnecessary, and that a tighter formulation can be found by making constraint (5.2d) an equality, which enables branching based on specially ordered set (SOS) constraints. Using (5.2d) as an equality constraint also reduces problem symmetry by curtailing alternative *z*-solutions that pertain to the same layout. Constraints (5.2m) address the set of fixed departments, forcing the respective locations and sizes equal to the corresponding given values. Although not displayed in (5.2), we note that it is also straightforward to adapt FLP2 to include constraints that require certain departments to be placed away from each other by at least some given distance.

The remainder of the model represents the constraints (5.1d-f). Note that the absolute values in (5.2g) can be linearized through either of two common techniques. The first option is to replace $d_{ij}^s = \left| c_i^s - c_j^s \right|$ with the two inequalities $d_{ij}^s \geq c_i^s - c_j^s$ and $d_{ij}^s \geq c_j^s - c_i^s$. The second option is to define two nonnegative variables, d_{ij}^{s+} and d_{ij}^{s-}, to represent the difference relationship as $d_{ij}^{s+} - d_{ij}^{s-} = c_i^s - c_j^s$ and then use the substitution $d_{ij}^s = d_{ij}^{s+} + d_{ij}^{s-}$. (In their computational experiments, Meller *et al*. implemented the first option.) The departments are required to be contained within the building through the constraints (5.2h). Finally, constraints (5.2i) impose the derived bounds on the dimensions based on area and aspect ratio considerations, and (5.2j - 5.2l) represent logical restrictions. This completes the basic FLP2 model.

5.2.2 The FLP2+ Model

After presenting this basic model, Meller *et al*. then strengthen FLP2 by developing a series of valid inequalities with the motivation of increasing the bound obtained from the linear programming relaxation of FLP2 (and any of its subsequent restrictions in a branch-and-bound framework). Typically, the LP relaxation sets the z_{ij}^s variables to fractional values, allowing the departments to overlap one another, and locates the centroid of each department at a common coordinate. This allows the d_{ij}^s variables to take on values of zero, and thus, the objective of the LP solution at the root node is equal to zero. In order to force the d_{ij}^s variables to take on non-zero values, Meller *et al*. develop transitivity constraints, lower bounding constraints for distance variables, and centroid separation constraints. The resulting enhanced model is referred to as FLP2+, and is shown below in complete form.

FLP2+: Minimize $\sum_{(i,j)\in P} \sum_{s=x,y} f_{ij} d_{ij}^s$ (5.3a)

subject to

$$4(\ell_i^x + \ell_i^y) \geq 3\sqrt{a_i} + f \times 2\ell_i^{\max} \quad \forall i \in \overline{F} \tag{5.3b}$$

$$\ell_i^{\max} \geq \ell_i^s, \quad \forall i \in \overline{F}, s \tag{5.3c}$$

$$\sum_{s=x}^{y} (z_{ij}^s + z_{ji}^s) = 1 \quad \forall i < j, s \tag{5.3d}$$

$$c_i^s + \ell_i^s \leq c_j^s - \ell_j^s + L^s(1 - z_{ij}^s) \quad \forall i \neq j, s \tag{5.3e}$$

$$d_{ij}^s \geq c_i^s - c_j^s \quad \forall (i,j) \in P, \forall s \tag{5.3f}$$

$$d_{ij}^s \geq c_j^s - c_i^s \quad \forall (i,j) \in P, \forall s \tag{5.3g}$$

$$z_{ij}^s + z_{jk}^s \leq 1 + z_{ik}^s \quad \forall i, j, k, s \tag{5.3h}$$

$$d_{ij}^x + d_{ij}^y \geq \sum_s (\ell_i^s + \ell_j^s) - \ell_i^{\max} - \ell_j^{\max} \quad \forall (i,j) \in P \tag{5.3i}$$

$$d_{ij}^s \geq (\overline{\ell b_i} + \overline{\ell b_j})(z_{ij}^s + z_{ji}^s) \quad \forall (i,j) \in P, s \tag{5.3j}$$

$$d_{ij}^s \geq (\ell_i^s + \ell_j^s) - \min\{\overline{ub_i} + \overline{ub_j}, L^s\}(1 - z_{ij}^s - z_{ji}^s) \quad \forall (i,j) \in P, s \tag{5.3k}$$

$$d_{ij}^s \geq (\overline{\ell b_i} + \overline{\ell b_j})(z_{ij}^s + z_{ji}^s) + 2\overline{\ell b_k}(z_{ik}^s + z_{kj}^s - 1) \quad \forall (i,j) \in P, k \neq i, j, \forall s \tag{5.3l}$$

$$d_{ij}^s \geq (\overline{\ell b_i} + \overline{\ell b_j})(z_{ij}^s + z_{ji}^s) + 2\overline{\ell b_k}(z_{ki}^s + z_{jk}^s - 1) \; \forall (i,j) \in P, k \neq i, j, \forall s \tag{5.3m}$$

$$\begin{aligned} d_{ij}^s \geq\; & (\ell_i^s + \ell_j^s) - \min\{\overline{ub_i} + \overline{ub_j}, L^s\}(1 - z_{ij}^s - z_{ji}^s) + 2\ell_k^s \\ & - \min\{2\overline{ub_k}, L^s\}(2 - z_{ik}^s - z_{kj}^s) \quad \forall (i,j) \in P, k \neq i, j, \forall s \end{aligned} \tag{5.3n}$$

$$\begin{aligned} d_{ij}^s \geq\; & (\ell_i^s + \ell_j^s) - \min\{\overline{ub_i} + \overline{ub_j}, L^s\}(1 - z_{ij}^s - z_{ji}^s) + 2\ell_k^s \\ & - \min\{2\overline{ub_k}, L^s\}(2 - z_{ki}^s - z_{jk}^s) \quad \forall (i,j) \in P, k \neq i, j, \forall s \end{aligned} \tag{5.3o}$$

$$\begin{aligned} d_{ij}^s \geq\; & (\overline{\ell b_i} + \overline{\ell b_j})(z_{ij}^s + z_{ji}^s) + 2\ell_k^s \\ & - \min\{2\overline{ub_k}, L^s\}(2 - z_{ik}^s - z_{kj}^s) \quad \forall (i,j) \in P, k \neq i, j, \forall s \end{aligned} \tag{5.3p}$$

$$\begin{aligned} d_{ij}^s \geq\; & (\overline{\ell b_i} + \overline{\ell b_i})(z_{ij}^s + z_{ji}^s) + 2\ell_k^s \\ & - \min\{2\overline{ub_k}, L^s\}(2 - z_{ki}^s - z_{jk}^s) \quad \forall (i,j) \in P, k \neq i, j, \forall s \end{aligned} \tag{5.3q}$$

$$c_i^s + \ell_i^s + 2\overline{\ell b_k}(z_{ik}^s + z_{kj}^s - 1) \leq c_j^s - \ell_j^s + L^s(1 - z_{ij}^s) \quad \forall i \neq j \neq k, \forall s \tag{5.3r}$$

$$c_j^s - \ell_j^s \geq 2\overline{\ell b_k} z_{ij}^s \qquad \forall i \neq j, s \tag{5.3s}$$

$$c_i^s + \ell_i^s \leq L^s - 2\ell b_j^s z_{ij}^s \qquad \forall i \neq j, s \tag{5.3t}$$

$$c_j^s - \ell_j^s \geq 2[\ell_i^s - \min\{\overline{ub_i}, L^s/2\}(1 - z_{ij}^s)] \qquad \forall i \neq j, s \tag{5.3u}$$

$$c_i^s + \ell_i^s \leq L^s - 2[\ell_j^s - \min\{\overline{ub_j}, L^s/2\}(1 - z_{ij}^s)] \qquad \forall i \neq j, s \tag{5.3v}$$

$$c_q^s \leq L^s/2 \qquad \forall s \tag{5.3w}$$

$$\overline{\ell b_i} \leq \ell_i^s \leq \min\{\overline{ub_i}, L^s/2\}, \forall i \in \overline{F}, s \tag{5.3x}$$

$$c_i^s \geq 0 \quad \forall i, s \tag{5.3y}$$

$$d_{ij}^s \geq 0 \quad \forall (i,j) \in P, s \tag{5.3z}$$

$$z_{ij}^s \in \{0,1\} \quad \forall i \neq j, s \tag{5.3aa}$$

$$(c_i^s, \ell_i^s) \text{ fixed } \forall i \in F, s. \qquad (5.3ab)$$

We now briefly comment on the derivation of the valid inequalities (5.3h) – (5.3w) that Meller *et al*. used to strengthen the basic FLP2 model. The transitivity constraints (T3) given by (5.3h) enforce logical relationships about the relative locations for any triplet of departments. The $d^{\min}$ constraint, shown in (5.3i), forces the rectilinear distance between departments *i* and *j* to be at least as large as $\min_s\{\ell_i^s\} + \min_s\{\ell_j^s\}$. In addition, Meller *et al*. enforce bounds on the distance between the centroids of departments i_1 and i_k, given that the sequence $i_1, \ldots, i_k$, $2 \le k \le n$ holds along direction *s*. Using the lower bounds, the variables themselves, and a combination of both, Meller *et al*. have developed distance bound constraints (Bka and Bkb), variable distance constraints (Vka and Vkb), and bound-variable distance constraints (BVka and BVkb), respectively. (We note that when $k = 2$, B2a ≡ B2b and V2a ≡ V2b, with BV2a and BV2b being redundant.) While these constraints can be developed for any value of *k* such that $2 \le k \le n$, Meller *et al*.'s computational analysis included constraints for only $k = 2,3$ in order to control the size of the problem. The constraints (5.3j) – (5.3q) correspond to Meller *et al*.'s B2, V2, B3a, B3b, V3a, V3b, BV3a, and BV3b. An additional series of constraints (Ska and Skb) was developed to increase the separation of the centroids of departments *i* and *j*, given any cycle $Ck: i_1, \ldots, i_k, i_1,$ for $2 \le k \le n$. In their computational analysis, Meller *et al*. included only S3a, displayed in (5.3r), noting that when $k = 2$, the S3a constraints reduce to (5.2f). The final set of valid inequalities are linearizations of the constraints

$$c_j^s - \ell_j^s \ge 2\ell_i^s z_{ij}^s \qquad \text{and} \qquad c_i^s + \ell_i^s \le L^s - 2\ell_j^s z_{ij}^s \qquad \forall i \ne j, s. \qquad (5.4)$$

The nonlinear terms $\ell_i^s z_{ij}^s$ are linearized by using $\ell b_i z_{ij}^s$ in (5.3s,t) and by using $\ell_i^s - \min\{ub_i, L^s/2\}(1 - z_{ij}^s)$ in (5.3u,v). Note that these constraints subsume (5.2h). As a final enhancement, Meller *et al*. implement a scheme to reduce problem symmetry, displayed in (5.3w), by forcing the centroid of some key department *q* to be positioned in the southwest corner of the building. The results presented in Meller *et al*. indicate that the additional inequalities of the enhanced model (FLP2+) provide significant computational advantages as compared to using the basic FLP2 model. Their computational experience revealed several test cases that were solved to optimality with FLP2+ yet could not be solved using model FLP2 . (We will discuss the effect of these constraints in more detail in Section 5.2.3.)

Although Meller *et al*. were able to strengthen the model FLP2, several significant improvements in the model formulation are yet possible. In the remaining sections of this chapter, we present various such enhancements that serve to tighten the model formulation and thereby decrease solution effort. We begin with a set of constraints that allow the nonlinear area constraints to be specified to any given accuracy, and we then address issues such as reducing problem symmetry more definitively, and deriving tighter representations of the inherent disjunctive relationships. In order to gauge the effectiveness of our proposed enhancements, we first evaluated the performance of the FLP2+ model on a series of problems that were presented in Meller *et al*. The results are described in the next section. We will subsequently use these same test problems to evaluate our proposed enhancements.

5.2 Experimental Design

In our computational analysis, we focused on the set of test problems that were presented in Meller *et al*. The problems range in size from three to nine departments, with either one or no fixed departments, and with aspect ratios ranging from three to five. Some properties of these test problems are summarized in Table 5.1. Here, as in Meller *et al*., we define flow density as the number of departmental pairs having positive flow interactions as a percentage of the maximum possible number of pairs; i.e., $(|P|/[n(n-1)/2])*100\%$. We define layout compactness as the percentage of available space occupied by the departments; that is, $\left(\sum_{i=1}^{n} a_i /(L^x \times L^y)\right)*100\%$. We also list the aspect ratio for each problem, where $\alpha_i = \alpha \ \ \forall i \in \overline{F}$. The FO problems correspond to flowshop versions of the O problems, where each flow intensity in an O problem instance is replaced by a unit value in the corresponding FO problem. We note that Meller *et al*. did not report results for problem M5 since their model FLP2+ declared this instance to be infeasible. Instead, they constructed the more relaxed problems M5-1 (with decreased area for each department) and M5-2 (with an increased aspect ratio for each department) and reported the error in area with respect to the original target values in M5. Although problems O7 and FO7 were not found to be infeasible using the FLP2+ model, Meller *et al*. employed a similar relaxation strategy for these problems to create the corresponding instances O7-1, O7-2, FO7-1, and FO7-2. However, using our more accurate modeling strategy as discussed below, we detected that M5 was indeed feasible, and so for the sake of consistency, we treat all these problems as separate test cases, and report on them individually with respect to their associated modified input parameters.

The performance of all proposed models was evaluated using an AMPL interface with CPLEX version 6.5.3 on a SUN Ultra-2 Workstation. Limits on time, number of nodes, and tree memory were set at 86,400 seconds (24 hours), 10 million nodes, and 390MB, respectively. For each problem, we report the best known integer solution (z_{MIP}) and the percentage optimality gap, $(z_{MIP} - z_{LB})/ z_{MIP}*100$, where z_{LB} is the lower bound at termination of the search. Additionally, we display the number of nodes and the solution time in CPU seconds. For each optimal solution, we also report the maximum error in department areas (due to the approximation employed in the area representations), calculated as $\max_{i \in F} \left|4\ell_i^x \ell_i^y - a_i\right| / a_i *100\%$.

The computational results of the FLP2+ analysis are presented in Table 5.2. We note that the results in Table 5.2 are similar to those presented in Meller *et al*., although there is a general reduction in computational effort which we attribute to advances in computing power and software technology. This reduced effort has led to tighter bounds for some problems that were not solved to optimality in Meller *et al*. We also note that there are some differences regarding the maximum error in department size since we report this statistic with respect to the given input parameters for each individual problem, rather than considering some problems as approximations of other instances.

Throughout the remainder of this chapter, we will evaluate the performance of alternative models for the facility layout problem. In order to assess the effect of each of our proposed

Table 5.1: Characteristics of the Test Problems.

Problem Name	Number of Departments		Aspect Ratio α	Flow Density (%)	Layout Compactness (%)
	Total	Fixed			
M3	3	0	3	66.67	88.00
M4	4	1	3	66.67	92.00
M5	5	1	3	50.00	100.00
M5-1	5	1	3	50.00	98.00
M5-2	5	1	5	50.00	100.00
M6	6	0	4	26.67	98.67
M7	7	0	4	23.81	99.00
FO7	7	0	4	28.57	99.98
FO7-1	7	0	4	28.57	97.48
FO7-2	7	0	5	28.57	99.98
FO8	8	0	4	25.00	99.98
FO9	9	0	4	22.22	100.00
O7	7	0	4	42.86	99.98
O7-1	7	0	4	42.86	97.48
O7-2	7	0	5	42.86	99.98
O8	8	0	4	53.57	99.98
O9	9	0	4	41.67	100.00

Table 5.2: Computational Results for the FLP2+ Model.

Problem	z_{MIP}	Optimality Gap (%)	Time	Nodes	Max. % Error
M3	3938.88	0	0.26	7	6.15
M4	5299.76	0	0.33	7	6.15
M5	Infeasible	n/a	2.9	163	n/a
M5-1	6370.34	0	2	85	5.50
M5-2	7621.58	0	4.4	301	10.32
M6	9412.90	0	40	518	4.17
M7	12971.30	0	670	5757	5.83
FO7	24.67	0	10000	79557	6.51
FO7-1	20.09	0	3700	21790	4.76
FO7-2	17.69	0	1500	7357	10.32
FO8	26.25	0	66000	215483	7.13
FO9	20.98	10.14	86400*	66386	4.76
O7	113.56	0	60000	393187	5.59
O7-1	96.00	0	17000	113396	4.76
O7-2	92.13	0	10000	62740	7.14
O8	182.40	26.45	86400*	107334	4.96
O9	166.66	40.00	86400*	50442	7.14

* Prematurely terminated after 24 hours of computation.

enhancements, we evaluate them in a sequential manner, starting with the basic FLP2 model and replacing the area constraints with our proposed representation. After determining the level of approximation in our area constraints that performs best, we then investigate new symmetry breaking constraints, followed by an analysis of the valid inequalities proposed by Meller *et al*. We conclude our development by examining several alternative formulations for the inherent disjunctive relationships that prevent departmental overlaps. In the process of evaluating our proposed enhancements in all these experiments, we consider only those problems that were solved to optimality by the model FLP2+. Following this, we then solve the remaining problems (FO9, O8, and O9) using some of the most promising strategies, as determined by our experimentation on the previous problems.

5.3 Improved Representation of the Nonlinear Area Constraints

One of the more challenging aspects of the facility layout problem arises in representing the nonlinear constraints that require each department to maintain a given area. Rather than relying on approximations based upon properties of rectangles, we propose an outer-linearization of the area constraints that can yield as tight an approximation as desired.

5.3.1 Development of the Area Constraints

Consider any department *i* of half-length ℓ_i^x and half-width ℓ_i^y that is to have an area of a_i with an aspect ratio of α_i, leading to the restrictions

$$\ell_i^x \le \alpha_i \ell_i^y,\ \ell_i^y \le \alpha_i \ell_i^x, \text{ and } 4\ell_i^x \ell_i^y = a_i. \tag{5.5}$$

Figure 5.1 illustrates the combinations of ℓ_i^x and ℓ_i^y that are feasible to (5.5). These combinations lie on the hyperbolic curve between the depicted points A and B. Note that the coordinates of A and B are given by

$$A \equiv (\ell b_i, ub_i),\ B \equiv (ub_i, \ell b_i),\ \text{where } \ell b_i \equiv \frac{\sqrt{a_i/\alpha_i}}{2} \text{ and } ub_i = \frac{\sqrt{a_i \alpha_i}}{2}. \tag{5.6}$$

We can additionally impose the constraints

$$2\ell_i^x \le L^x \text{ and } 2\ell_i^y \le L^y. \tag{5.7}$$

Consequently, we can tighten the upper bounds on ℓ_i^x and ℓ_i^y to ub_i^x and ub_i^y, respectively, where

$$ub_i^x = \min\{\frac{\sqrt{a_i\alpha_i}}{2}, \frac{L^x}{2}\} \text{ and } ub_i^y = \min\{\frac{\sqrt{a_i\alpha_i}}{2}, \frac{L^y}{2}\}. \tag{5.8a}$$

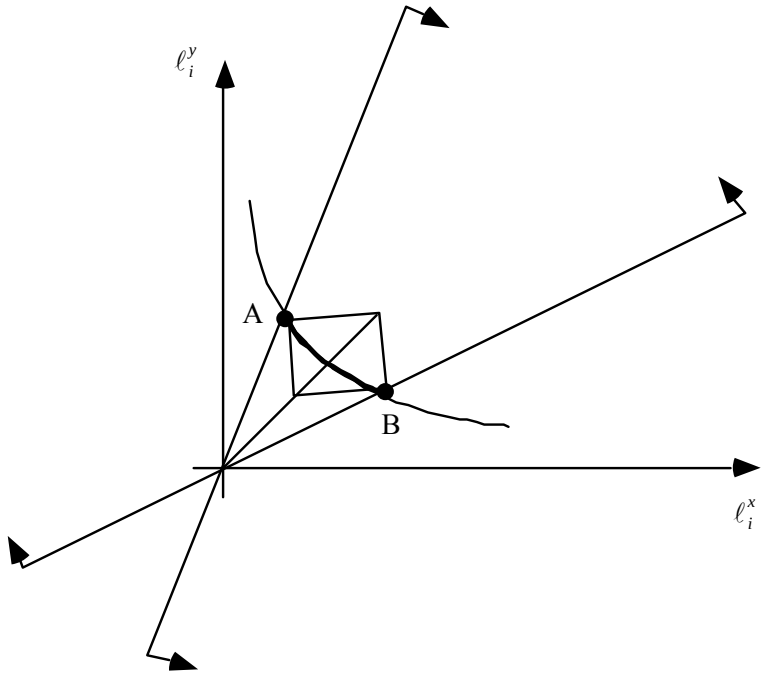

Figure 5.1. Depiction of Area Constraints.

Since $4\ell_i^x\ell_i^y = a_i$ must hold true, this correspondingly yields lower bounds ℓb_i^x and ℓb_i^y on ℓ_i^x and ℓ_i^y, respectively, as

$$\ell b_i^x = \frac{a_i}{4(ub_i^y)} \text{ and } \ell b_i^y = \frac{a_i}{4(ub_i^x)}. \tag{5.8b}$$

Hence, we can impose the bounds stated below as given by (5.8a, b):

$$\ell b_i^s \le \ell_i^s \le ub_i^s \text{ for } s = x, y, \quad \forall\ i. \tag{5.8c}$$

It is important to note that these bounds are tighter than the bounds $(\overline{\ell b_i}, \overline{ub_i})$ proposed by Meller *et al*., who take the *maximum* of L^x and L^y to bound both $2\ell_i^x$ and $2\ell_i^y$, in lieu of our bounding scheme in (5.8). More importantly, Meller *et al*. then derive an empirical approximation for the area constraints that can be significantly improved, leading to a more accurate and stronger representation.

While Meller *et al*. approximate the nonlinear area restrictions for each department with the constraints (5.2b,c), we propose instead to derive a polyhedral outer-approximation of the area constraints. Consider the hyperbolic curve between A' and B', where A' and B' refer to the

appropriate end-points (that replace *A* and *B*, respectively, in Figure 5.1) on the valid portion of this curve based on the modified values for the bounds as given by (5.8). The coordinates of these end-points A' and B' are given by

$$A' = (\ell b_i^x, ub_i^y) \text{ and } B' = (ub_i^x, \ell b_i^y).$$

In place of the nonlinear area constraints (5.5), we propose a polyhedral approximation that is comprised of the affine concave envelope that passes through A' and B', along with a suitable number of affine supports to the convex hyperbolic curve between A' and B'. The former concave envelope of this segmented function yields the valid inequality

$$\ell_i^x(ub_i^y - \ell b_i^y) + \ell_i^y(ub_i^x - \ell b_i^x) \le ub_i^x ub_i^y - \ell b_i^x \ell b_i^y. \tag{5.9}$$

Furthermore, the convex envelope (which is described by the function itself) yields the set of valid approximating linear inequalities (based on tangential supports to the curve $\ell_i^y = a_i/4\,\ell_i^x$ at various points $\bar{x}$, where $\ell b_i^x \le \bar{x} \le ub_i^x$) given by

$$\ell_i^y \ge \frac{a_i}{4\bar{x}} + (\ell_i^x - \bar{x})\left(\frac{-a_i}{4\bar{x}^2}\right)$$

$$\text{i.e.} \quad a_i \ell_i^x + 4\bar{x}^2 \ell_i^y \ge 2a_i\bar{x} \quad \forall \ell b_i^x \le \bar{x} \le ub_i^x. \tag{5.10}$$

For example, we can use values of $\bar{x}$ equal to

$$\bar{x} = \ell b_i^x + \frac{\lambda}{(\Delta - 1)}(ub_i^x - \ell b_i^x) \quad \forall \lambda = 0,1,\ldots,\Delta - 1, \text{ for any selected integer } \Delta \ge 2. \tag{5.11}$$

Note that unlike the piecewise linearization used in Lacksonen (1994), this approximation is purely linear and does not involve any binary variables. Furthermore, it can provide as tight a representation as desired unlike the approximation used in Meller *et al*., assuming that by the linearity of the problem, the ultimate values of (ℓ_i^x, ℓ_i^y) turn out to be vertices of the corresponding outer approximating polytope for each *i*. This is likely to be the case (as borne out by our results) since the problem tendency is naturally to underestimate the areas. We also note that Meller *et al*. quote maximum error values for their area approximation, but these are actually only errors stemming from an *under-representation* of the area under consideration. However, their approximation can have significant errors in *over-representing* the areas. For example, with $\alpha_i = 4$ and $(\ell b_i, ub_i) = (\sqrt{a_i}/4, \sqrt{a_i})$ from (5.6), if we take $\ell_i^x = \ell_i^y = \sqrt{a_i}$, this satisfies (5.2b), but yields an error of $100[(4a_i - a_i)/a_i] = 300\%$. The role of (5.9) above is to reduce such an over-representation (the solution $\ell_i^x = \ell_i^y = \sqrt{a_i}$ violates this constraint, for example).

5.3.2 Effect of the Proposed Area Constraints

In order to determine the effect of the proposed area constraints, we evaluated the performance of the FLP2 model with the area constraints (5.2b,c) replaced by (5.9) - (5.11). As

in Meller *et al*.'s analysis, we eliminated constraint (5.2e) and changed (5.2d) to an equality, and we modeled the absolute value constraints through a pair of inequalities. We did not, however, include any of the symmetry breaking techniques or valid inequalities that were proposed by Meller *et al*., as we will study these features of the model in subsequent sections. For each of the test problems, we varied the number of discretization points Δ for the tangential supports from five to fifty, and the results of these runs are compared to those obtained for FLP2+ in Tables 5.3 and 5.4.

In examining Tables 5.3 and 5.4, we first note that the optimal solution values of several test problems vary significantly when solved by the FLP2+ model as opposed to the FLP2 model with our proposed area constraints. In the case of problem M5, for instance, an optimal solution was found using our proposed area constraints, while the problem was declared to be infeasible using the FLP2+ model. For most problems, our proposed area constraints lead to a noticeably improved optimal solution value, while in two instances (FO7-1 and FO7-2), they lead to a slightly higher optimal value. This can be explained by recalling that the FLP2+ model approximates the nonlinear area constraints based upon relationships between the perimeter and the area of a rectangle. These approximations frequently add unnecessary restrictions to the problem and needlessly increase the optimal solution value, as evidenced by our computational results. At times, however, they admit optimal solutions to the approximating model that significantly violate the area constraints that they purport to represent, thus producing solutions that are actually infeasible to the given original problem. In contrast, our proposed area constraints model the underlying nonlinear area restrictions in a consistent manner. As the number of tangential supports increases, the solutions are forced to more closely approximate the actual nonlinear area constraints (because of the natural tendency of underapproximate the areas), thus increasing the optimal solution value. Furthermore, the results for some problems show a leveling-off effect as the number of supports increases, indicating that we are approaching solutions that exactly satisfy the nonlinear area constraints.

This increase in accuracy can also be seen by examining the maximum error for each problem. Tables 5.3 and 5.4 indicate that the proposed area constraints are *quite* effective in decreasing the error with respect to departmental area constraints. While each of the test problems exhibited a maximum error of greater than 4% (as high as 10% for some problems) when solved using the FLP2+ model, *our proposed area constraints decreased this error to less than 1% with the use of just ten tangential supports for each department*. Furthermore, as expected, increasing the number of supports led to an even greater reduction in departmental errors. Overall, the FLP2+ model produced an average maximum error of 6.45% while our proposed area constraints reduced this average maximum error to 2.28%, 0.37%, 0.19%, 0.05%, 0.04%, and 0.03% when using 5, 10, 20, 30, 40, and 50 supports, respectively.

Perhaps the most (pleasantly) surprising result, however, is the *dramatic* decrease in solution time achieved through the use of the *more accurate* proposed area constraints. In several problem instances, the solution time was decreased by over 95%. We note that the times presented in Tables 5.3 and 5.4 were obtained using simply the model FLP2 with the new area constraints. *That is, this model does not include any symmetry breaking constraints or valid inequalities, while the results from FLP2+ included both of these enhancements.* A possible explanation of this phenomenon is that a tighter control on the dimensions of each department

Table 5.3: Effect of Area Constraints on M Problems.

	Model	**FLP2+**	**FLP2 with Proposed Area Constraints**					
Problem	**Supports**	**0**	**5**	**10**	**20**	**30**	**40**	**50**
M3	z_{MIP}	3938.88	3750.86	3774.77	3778.09	3779.30	3779.51	3779.79
	Time	0.26	0.05	0.06	0.04	0.07	0.09	0.06
	Nodes	7	6	6	4	6	6	4
	Max. Error	6.15	1.08	0.62	0.09	0.08	0.02	0.02
M4	z_{MIP}	5299.76	5078.98	5103.28	5106.39	5107.94	5108.25	5108.50
	Time	0.33	0.04	0.04	0.06	0.06	0.1	0.08
	Nodes	7	7	7	7	7	7	7
	Max. Error	6.15	1.08	0.62	0.09	0.08	0.02	0.02
M5	z_{MIP}	Infeasible	6131.43	6172.16	6170.89	6174.22	6174.15	6174.88
	Time	2.9	0.14	0.16	0.24	0.22	0.28	0.35
	Nodes	163	21	21	27	20	21	27
	Max. Error	n/a	1.08	0.06	0.09	0.02	0.02	0.01
M5-1	z_{MIP}	6370.34	5068.31	5088.03	5094.98	5095.27	5095.75	5095.92
	Time	2	0.12	0.13	0.16	0.22	0.24	0.27
	Nodes	85	18	18	20	18	18	19
	Max. Error	5.50	1.07	0.53	0.07	0.04	0.03	0.01
M5-2	z_{MIP}	7621.58	5155.78	5214.41	5226.93	5225.35	5226.03	5227.23
	Time	4.4	0.19	0.22	0.22	0.29	0.31	0.4
	Nodes	301	46	33	30	46	43	45
	Max. Error	10.32	3.19	0.76	0.75	0.09	0.08	0.02
M6	z_{MIP}	9412.90	8166.68	8212.04	8222.32	8224.13	8224.30	8224.73
	Time	40	0.94	0.37	1.2	1.3	0.82	1.8
	Nodes	518	212	60	210	185	93	200
	Max. Error	4.17	3.45	0.37	0.23	0.04	0.05	0.05
M7	z_{MIP}	12971.30	10592.50	10658.40	10672.10	10673.40	10673.70	10674.40
	Time	670	0.78	1	2.3	2.2	2.1	1.9
	Nodes	5757	180	266	466	328	304	261
	Max. Error	5.83	3.45	0.37	0.23	0.04	0.05	0.05

Table 5.4: Effect of Area Constraints on FO and O Problems.

	Model	**FLP2+**	**FLP2 with Proposed Area Constraints**					
Problem	**Supports**	**0**	**5**	**10**	**20**	**30**	**40**	**50**
FO7	z_{MIP}	24.67	20.92	20.94	20.95	20.95	20.95	20.95
	Time	10000	1900	2100	3600	3500	2700	2300
	Nodes	79557	462695	444119	632642	519886	331524	245210
	Max. Error	6.51	0.44	0.12	0.05	0.01	0.00	0.01
FO7-1	z_{MIP}	20.09	20.21	20.23	20.25	20.25	20.25	20.25
	Time	3700	2200	1100	2900	2900	3600	2200
	Nodes	21790	544105	235159	528733	381646	435804	277196
	Max. Error	4.76	3.68	0.11	0.31	0.06	0.06	0.06
FO7-2	z_{MIP}	17.69	17.70	17.75	17.75	17.75	17.75	17.75
	Time	1500	410	450	340	440	980	990
	Nodes	7357	105852	101608	59747	62869	133340	122631
	Max. Error	10.32	1.01	0.02	0.01	0.04	0.05	0.01
FO8	z_{MIP}	26.25	22.22	22.27	22.31	22.37	22.38	22.38
	Time	66000	3900	4700	5100	6900	12000	7100
	Nodes	215483	734723	759959	596769	861074	1311596	728357
	Max. Error	7.13	0.91	0.57	0.40	0.04	0.02	0.03
O7	z_{MIP}	113.56	98.16	98.44	98.49	98.51	98.52	98.51
	Time	60000	5400	5500	8900	7700	8800	5500
	Nodes	393187	1252617	1063177	1615783	1187869	1077777	596892
	Max. Error	5.59	0.81	0.24	0.05	0.02	0.01	0.01
O7-1	z_{MIP}	96.00	89.79	90.68	90.76	90.84	90.85	90.84
	Time	17000	1200	720	1300	3200	2800	4000
	Nodes	113396	290059	145045	197815	482342	373823	481320
	Max. Error	4.76	2.83	0.14	0.27	0.04	0.00	0.02
O7-2	z_{MIP}	92.13	84.61	90.54	90.57	90.59	90.59	90.60
	Time	10000	1600	2900	1700	2100	5200	2700
	Nodes	62740	3525087	614357	280844	320777	691973	304978
	Max. Error	7.14	6.20	0.95	0.15	0.12	0.10	0.07

works favorably in concert with the disjunctive separation constraints, and admits a more effective scheme for fathoming inferior solutions. Table 5.5 provides a summary of the average factor of improvement (given by the corresponding FLP2+ value divided by our value) for each number of supports. We observe that on average, when at least twenty supports are used, our model yields solutions that are over 100 times more accurate than the FLP2+ solution, while curtailing effort in comparison with the FLP2+ model by a factor of over 27 times. We further note that the dramatic decrease in solution time associated with using our area constraints is seen across all problem types and sizes, and it is obtained while simultaneously providing more accurate solutions, frequently with a lower optimal objective value.

While we have demonstrated that our proposed area constraints provide increasingly accurate solutions as the number of tangential supports increases, this also entails an increase in solution effort, as evidenced by Figures 5.2 and 5.3. Our computational experience indicates that a minimum of 10 supports are clearly necessary in order to achieve a reasonable degree of accuracy in representing the area constraints. As the number of supports increases to 50 for each department, this error approaches zero. Throughout the remainder of this chapter, we will propose several additional strategies for enhancing the solvability of the facility layout model. These enhancements will not alter the objective value or the accuracy of the optimal solution, but are intended simply to further decrease solution effort. For this reason, we will evaluate each of the remaining enhancements using a fixed number of tangential supports for the area constraints of each department. For our purposes, we determined to use the minimum number of supports necessary to achieve an acceptable level of average maximum error, which we selected to be 0.25%. At such a level, for instance, the maximum amount of error corresponds to a six by six inch square for a department with a target area of 100 square feet. Accordingly, we opted to conduct all remaining experiments using 20 supports, which led to an average maximum error of 0.18%.

Table 5.5: Factor of Improvement in Solution Time and Error.

Number of Supports	Solution Time	Maximum Error
5	68.98	5.39
10	60.54	59.33
20	28.31	113.95
30	27.74	201.64
40	28.95	763.20
50	29.71	390.91

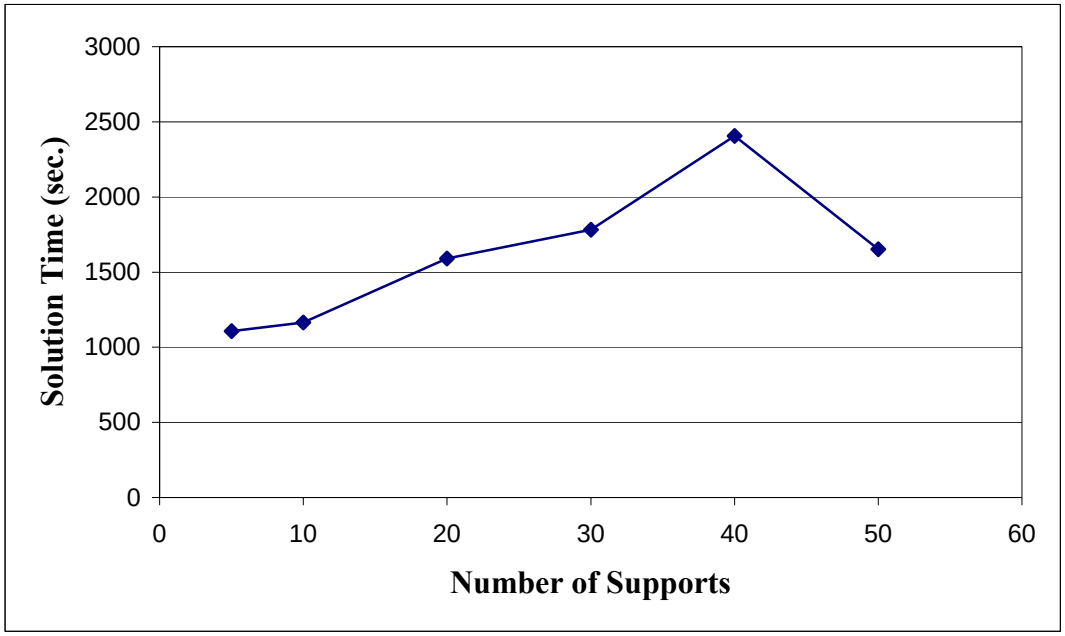

Figure 5.2. Average Solution Time versus Number of Supports.

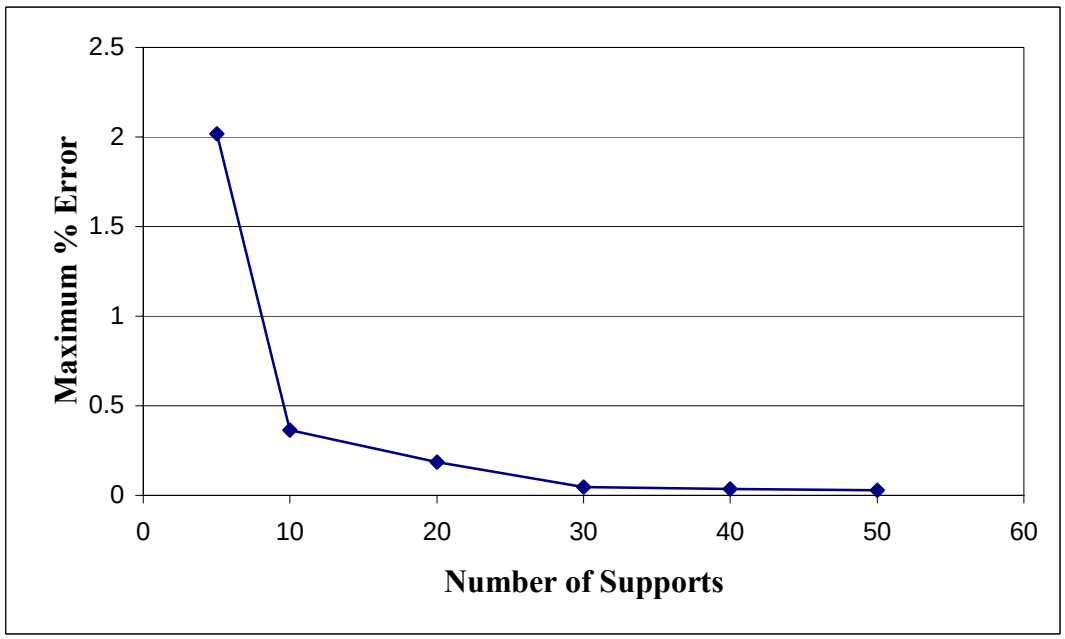

Figure 5.3. Average Maximum Error versus Number of Supports.

5.4 Reducing Problem Symmetry

As noted in Meller *et al*., the solution of FLP2 can be significantly slowed by the large degree of symmetry in the problem. While Meller *et al*. incorporated a symmetry breaking constraint as embodied by (5.3w) to reduce this effect, we propose and test two alternative symmetry breaking strategies.

5.4.1 Development of Alternative Symmetry Breaking Strategies

In order to reduce the solution effort consumed by searching for symmetrical solutions, Meller *et al*. incorporated the following symmetry-breaking constraint in their implementation:

$$c_q^s \leq L^s/2 \text{ for } s = x, y, \text{ for some key department } q. \tag{5.12}$$

This tends to eliminate the symmetry with respect to 180º flips in the x or y directions. However, as depicted for the solution in Figure 5.4, this might not always help or serve the intended purpose. Observe that constraint (5.12) continues to hold true when the layout is flipped 180º in either the x or y directions. We now propose two alternative classes of symmetry-breaking constraints that turn out to be more definitive in ameliorating symmetry effects. We first note that, in general, symmetry breaking techniques are not valid in the presence of departments with fixed locations, as frequently the problem symmetry is already eliminated by forcing certain departments to be placed at specific locations. However, if the fixed departments are themselves symmetric with respect to flips in either the x or y directions, then the corresponding symmetry breaking constraints could additionally be incorporated.

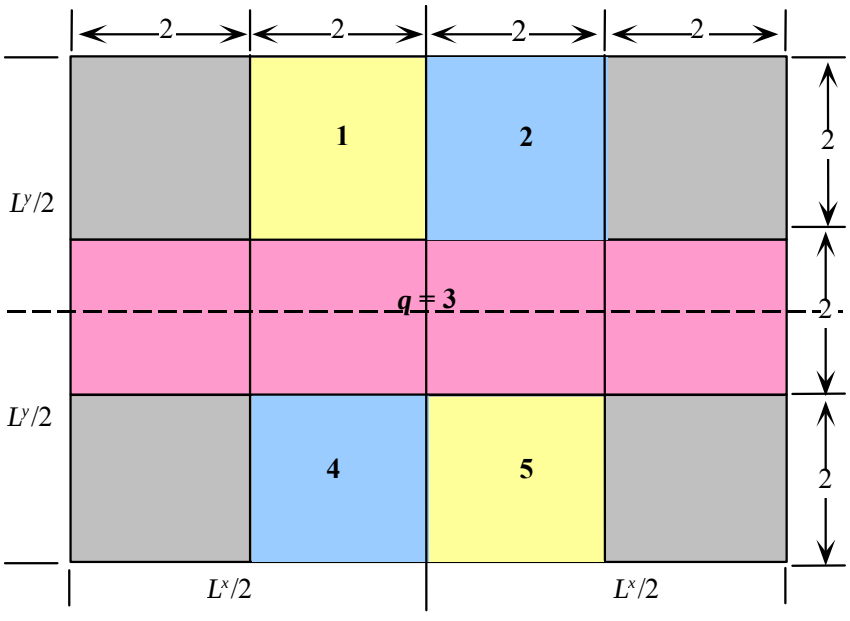

Figure 5.4. Symmetry Considerations.

As shown in Sherali and Smith (1999), the generation of suitable hierarchical constraints to curtail the symmetry inherent in many applications can greatly benefit the model representation and its consequent solvability. We now propose a set of hierarchical symmetry breaking constraints for the facility layout problem. Our first symmetry breaking method requires the orientation of the encompassing rectangle with respect to 180º flips in the x or y direction to be such that a particular hierarchy is established in a specified function value when applied along, versus in reverse to, each axis direction. For example, taking this function to be

the sum of centroids weighted by their indices, we can impose

$$\sum_{i=1}^{n} ic_i^s \le \sum_{i=1}^{n} i(L^s - c_i^s) \text{ for } s = x, y, \tag{5.13}$$

$$\text{i.e., } 4\sum_{i=1}^{n} ic_i^s \le n(n+1)L^s \text{ for } s = x, y. \tag{5.14}$$

For the example in Figure 5.4, when $s \equiv x$, the inequality (5.14) is violated since 4[1(3) + 2(5) + 3(4) + 4(3) + 5(5)] = 248 > (5) (6) (8) = 240. Hence, we would need to flip the layout 180º in the x-direction in order to satisfy (5.14). Furthermore, it can be verified that the layout satisfies (5.14) in the y-direction, but not if it is flipped in this direction.

For the second type of symmetry-breaking constraint, we consider a pair of departments p and q based on a maximum total interaction and/or area-based criterion, and we then require the centroid of p to be south and west of the centroid of q. For example, with $p = 4$, and $q = 3$, we (uniquely) obtain the configuration of Figure 5.4. However, with $p = 1$ and $q = 2$, flipping in the y-direction yields an alternative acceptable configuration. As such, we can impose

$$c_p^s \le c_q^s \text{ for } s = x, y. \tag{5.15}$$

We can further tighten the model of Meller *et al*. under (5.15) by accordingly restricting

$$z_{qp}^x = z_{qp}^y = 0, \text{ and } \sum_{s=x}^{y} (c_q^s - c_p^s) \ge \min\{\ell b_p^x + \ell b_q^x, \ell b_p^y + \ell b_q^y\}. \tag{5.16}$$

5.4.2 Effect of Symmetry Breaking Constraints

In this section, we discuss the effect of three symmetry breaking techniques for the facility layout problem. The results of our computational analysis are presented in Tables 5.6 through 5.8. The first of these three alternatives is the hierarchical constraints of (5.14), which we refer to in our analysis simply as *hierarchy*. The second alternative that we considered was using constraints (5.15) – (5.16), referred to as *position p-q*. In our experimentation, we selected departments p and q as a pair having the largest flow; that is, $f_{pq} = \max_{(i,j) \in P} f_{ij}$. In the case of ties, we selected a pair among such ties having the maximum total area. (We note here that several other methods were explored for selecting departments p and q, but the variations performed similarly, and in many cases, selected the same two departments as did the foregoing strategy.)

Table 5.6: Effect of Symmetry Breaking Techniques on M Problems.

Problem	Symmetry	Time	Nodes
M3	None	0.04	4
	Hierarchy	0.05	4
	Position p-q	0.03	1
	Position q	0.03	3
M6	None	1.20	210
	Hierarchy	1.10	149
	Position p-q	0.50	34
	Position q	0.76	93
M7	None	2.30	466
	Hierarchy	1.90	326
	Position p-q	0.73	86
	Position q	1.40	243

The final alternative that we evaluated, referred to as *position q*, is the strategy proposed by Meller *et al*. and displayed in (5.12). We applied each of the aforementioned strategies to the FLP2 model, using our proposed area constraints (5.9) - (5.11) with 20 tangential supports for each department. No additional valid inequalities were included at this stage. (As noted earlier, since these symmetry breaking techniques are only valid for problems having no fixed departments, we did not implement these strategies on problems M4, M5, M5-1, and M5-2.)

The results indicate that by using symmetry breaking techniques, we dramatically decrease the solution effort, both in terms of solution time and the number of nodes enumerated, for nearly all problem instances. It is also clear that the hierarchical symmetry breaking constraints do not perform as well as the other two alternatives, noting that in the last two problem instances, the solution time actually increased over the model with no symmetry reduction techniques. We believe that this stems from the dense nature of the hierarchical constraints, which may interfere with the special structures of the model that are exploited by CPLEX throughout the branch-and-bound process. In contrast, the other two symmetry breaking alternatives consist of sparse constraints having unit coefficients, and do not adversely affect the problem structure.

While problem effort decreases with the position p-q and position q strategies, our computational results do not indicate that either method is clearly superior to the other. For example, in terms of solution time, position p-q is best on four problems, position q is best on five, and one problem is solved equally quickly by both options. Table 5.8 displays the average decrease in solution effort (as compared to using no symmetry breaking techniques) for each of the proposed methods. This table confirms that the hierarchical constraints are outperformed by the other alternatives, which perform competitively with respect to each other. For this reason, we will investigate how each of these two latter strategies perform in conjunction with the valid inequalities explored in the following section.

Table 5.7: Effect of Symmetry Breaking Techniques on FO and O Problems.

Problem	Symmetry	Time	Nodes
FO7	None	3600	632642
	Hierarchy	1200	174605
	Position p-q	830	126741
	Position q	740	124046
FO7-1	None	2900	528733
	Hierarchy	1500	255507
	Position p-q	380	62512
	Position q	790	147831
FO7-2	None	340	59747
	Hierarchy	310	49240
	Position p-q	170	29112
	Position q	180	32413
FO8	None	5100	596769
	Hierarchy	4000	450287
	Position p-q	2000	284944
	Position q	1700	219929
O7	None	8900	1615783
	Hierarchy	3800	534090
	Position p-q	2700	452488
	Position q	1800	285649
O7-1	None	1300	197815
	Hierarchy	1700	272910
	Position p-q	1400	252366
	Position q	630	98751
O7-2	None	1700	280844
	Hierarchy	2900	446433
	Position p-q	1300	224042
	Position q	820	136410

Table 5.8: Average % Decrease in Solution Effort

Symmetry Type	Time	Nodes
Hierarchy	10.20	19.53
Position p-q	51.17	57.67
Position q	54.98	57.37

Before concluding this section, we take a moment to reflect on why the position p-q strategy did not clearly dominate the Meller *et al*. strategy, although the position p-q has been shown to eliminate symmetrical cases that are not eliminated by the position q method. The reason for this is that the position p-q strategy also introduces additional valid inequalities (5.16). While these inequalities assist by tightening the relaxation, the compromise between obtaining tighter bounds and expending more effort in this process does not turn out to be uniformly favorable. Note that the position q approach does nothing to eliminate the relaxed solution that locates all the departments at a common location, and often yields a root node relaxation value of zero. The position p-q strategy, however, eliminates this possibility by including the centroid separation constraints in (5.16). (The root node analysis of the following section contains results to support this argument.) However, as we shall see subsequently, when suitable additional valid inequalities are added to the model, the position p-q strategy begins to more strongly dominate the position q alternative.

5.5 Additional Valid Inequalities

As discussed in Chapter 2, the derivation of problem-specific valid inequalities can greatly increase the strength of a (mixed) integer program. Section 5.2.2 provided an overview of the valid inequalities incorporated by Meller *et al*. in the FLP2+ model. In the following sections, we discuss the effect of including only certain subsets of the valid inequalities, (5.3h)-(5.3v), used in FLP2+. We wish to evaluate the effect of these valid inequalities when applied to the FLP2 model using our proposed area constraints with 20 tangential supports per department, in combination with each of the competitive symmetry breaking methods: position p-q and position q. We note that when appending these valid inequalities to our model, we replace the bounds $\overline{\ell b_i}$ and $\overline{ub_i}$, respectively, by our tighter bounds ℓb_i^s and ub_i^s.

5.5.1 Root Node Analysis

In early computational experiments, it became quite evident that the addition of all the valid inequalities proposed by Meller *et al*. led to a *drastic* increase in overall solution effort. Although the proposed inequalities did reduce the number of nodes enumerated, the trade-off between better bounds and increased solution effort was not favorable. To demonstrate this effect, we display the solution effort for a sample of smaller problem instances in Table 5.9. Rather than continuing to solve the remaining problems using a solution technique that was clearly not effective, we instead conducted an analysis of how the valid inequalities performed with respect to the LP relaxation at the root node itself. Our hope was to determine a subset of the proposed inequalities that served to provide a substantial tightening of the LP relaxation, without encumbering the associated solution effort. By solving the root node LP relaxation using various subsets of the valid inequalities proposed by Meller *et al*., we were able to determine that the best results were obtained by incorporating only the constraints B2 and V2 displayed in (5.3j) and (5.3k), respectively. We display the results pertaining to the objective value at the root node in Table 5.10 and to the solution time in Table 5.11.

Table 5.9: Solution Effort for Several Smaller Problems.

Symmetry Breaking	Valid Inequalities	M6		M7		FO7-2	
		Time	Nodes	Time	Nodes	Time	Nodes
Position p-q	None	0.5	34	0.73	86	170	29112
Position p-q	All	2.6	9	4.3	8	1100	4314
Position q	None	0.76	93	1.4	243	180	32413
Position q	All	2.7	10	4.1	6	1500	5736

Table 5.10: Objective Value at the Root Node Using Various Valid Inequalities.

Valid Inequalities	None		All		B2 and V2 Only	
Symmetry	Pos. p-q	Pos. q	Pos. p-q	Pos. q	Pos. p-q	Pos. q
M3	2038.66	0.00	3778.09	3778.09	3778.09	3778.09
M4	1374.70	1374.70	5092.96	5092.96	5021.45	5021.45
M5	1406.77	1406.77	5156.55	5156.55	5053.52	5053.52
M5-1	1389.12	1389.12	5027.21	5027.21	4927.23	4927.23
M5-2	1405.95	1405.95	5138.81	5138.81	5053.96	5053.96
M6	3734.85	0.00	7954.14	7921.83	7921.28	7921.28
M7	3734.85	0.00	10473.98	10448.08	10376.04	10376.04
FO7	2.50	0.00	11.88	11.88	11.88	11.88
FO7-1	2.47	0.00	11.72	11.72	11.72	11.72
FO7-2	2.28	0.00	10.76	10.76	10.76	10.76
FO8	3.00	0.00	14.76	14.76	14.76	14.76
O7	15.00	0.00	44.73	44.66	44.64	44.64
O7-1	14.81	0.00	44.09	44.03	44.00	44.00
O7-2	13.67	0.00	41.00	40.89	40.85	40.85

Table 5.11: Solution Time at the Root Node Using Various Valid Inequalities.

Valid Inequalities	**None**		**All**		**B2 and V2 Only**	
Symmetry	**Pos. p-q**	**Pos. q**	**Pos. p-q**	**Pos. q**	**Pos. p-q**	**Pos. q**
M3	0.02	0.01	0.01	0.03	0.01	0.00
M4	0.02	0.02	0.04	0.04	0.02	0.01
M5	0.03	0.03	0.07	0.08	0.02	0.03
M5-1	0.03	0.03	0.07	0.08	0.03	0.03
M5-2	0.01	0.02	0.09	0.08	0.01	0.01
M6	0.03	0.01	0.43	0.43	0.06	0.04
M7	0.03	0.04	0.40	0.56	0.05	0.04
FO7	0.08	0.07	1.80	2.00	0.12	0.11
FO7-1	0.06	0.05	1.70	2.20	0.13	0.11
FO7-2	0.05	0.06	2.20	2.40	0.14	0.13
FO8	0.10	0.07	4.00	3.80	0.15	0.14
O7	0.04	0.05	2.90	2.60	0.17	0.17
O7-1	0.06	0.05	3.10	2.60	0.14	0.15
O7-2	0.05	0.05	2.70	2.50	0.13	0.16

First of all, note that the results support the hypothesis of the previous section that the position p-q symmetry breaking strategy increases the value of the LP relaxation, while in some cases, also increases the required solution effort. Note that when using the position q symmetry-breaking strategy with no valid inequalities, the objective value of the root node is zero for all problems having no fixed departments, indicating that the centroids of all the departments are placed at a single location. In contrast, the position p-q symmetry-breaking strategy yields strictly positive solution values for all problems by enforcing a centroidal separation for at least the two key departments. (In the case of fixed departments, the symmetry tends to be broken by the fixed departments themselves, and no additional symmetry-breaking measures are employed.) Even when using all the valid inequalities, there are several problem instances for which the position p-q strategy continues to provide strictly better bounds than the position q strategy.

The results also indicate that by including all of the valid inequalities proposed by Meller *et al*., the lower bound is substantially increased over that obtained without using any valid inequalities. We note, however, that this increase comes at quite an expense. For example, each of the O problem instances experience an increase in solution time of over 5000% as compared to when no valid inequalities are used. At the same time, however, the increase in the lower bounds averages only 198%. On the other hand, note that the use of only two of the proposed valid inequalities (B2 and V2) yields nearly the same increase in the lower bound at only a fraction of the computational cost. On average, using only B2 and V2 achieves a bound equal to 99.38% of that obtained using all the proposed inequalities, and takes only 16.64% of computational effort. If we omit the relatively simple M problems from this analysis, the average bound increases to 99.92% of that obtained using all the proposed inequalities, and the time decreases to 5.57% on average. Given such a promising performance in the root node

relaxation, we then examined the effect of including only the B2 and V2 constraints on the overall branch-and-bound search process.

5.5.2 Effect of Valid Inequalities on the Branch-and-Bound Process

Table 5.12 displays the overall solution effort when incorporating only the classes of valid inequalities B2 and V2. For convenience, we also display in Table 5.12 the cpu time and the number of nodes enumerated when no valid inequalities were included. For the problems having no fixed departments, this data corresponds to the information displayed in Tables 5.6 and 5.7. Since no symmetry breaking constraints were investigated for the cases having fixed departments, we report the corresponding solution effort obtained in both the columns pertaining to the two symmetry breaking strategies for problems M4, M5, M5-1, and M5-2.

The results of Table 5.12 show that the valid inequalities B2 and V2 are effective at decreasing both solution time and the number of nodes for nearly all problem instances. A notable exception is Problem O7. For this problem, the inclusion of inequalities B2 and V2 led to a substantial increase in effort for both types of symmetry-breaking constraints. These results are not surprising, given the performance of B2 and V2 at the root node for Problem O7. When using the position p-q symmetry-breaking technique, the root node lower bound increased by 197.6% with the inclusion of B2 and V2, but the solution time increased by 325%, indicating that the gains made by introducing B2 and V2 were not worth the computational expense. We note that, under the position p-q symmetry breaking approach, Problem O7 is the only instance in this test set for which the percentage increase in the lower bound is lower than the percentage increase in time at the root node, thereby helping to explain why using the valid inequalities B2 and V2 performed much differently on this particular problem. In practice, when solving a new instance of the facility location problem, conducting a quick analysis of how B2 and V2 affect the performance at the root node might help the user determine such instances when B2 and V2 will be detrimental to the overall search process. (We note that when the position q strategy is used and no departments are fixed, the lower bound for the root node relaxation is always zero, which precludes the foregoing type of analysis.)

We therefore will disregard Problem O7 for the remainder of the analysis in this section pertaining to determining the effect of adding B2 and V2. In addition, we will focus only on problems having seven or more departments, since the smaller problems are easily solved under any of the methods considered. Focusing on the remaining seven problems, we note that the B2 and V2 constraints are particularly effective when used in conjunction with the position p-q strategy, reducing both the number of nodes and solution time in every problem instance. When B2 and V2 are used with the position q strategy, the number of nodes enumerated decreases in all problem instances except for O7-2, while the solution time increases for problems FO8 and for all the O problems. We also note that when constraints B2 and V2 are used, five of the seven problems are solved faster under the position p-q strategy than under the position q alternative. Additionally, when using constraints B2 and V2, the seven problems are solved in 3491 seconds with the position p-q technique, but require a total of 7181 seconds with the position q method. Furthermore, the 3491 second total solution time is by far the lowest of the four studied methods thus far, with the position q method using no valid inequalities coming in second with a total of 4862 seconds. Given these results, coupled with the fact that using the position p-q strategy

Table 5.12: Effect of Valid Inequalities on the Overall Branch-and-Bound Process.

	Valid	Position p-q		Position q	
Problem	**Inequalities**	**Time**	**Nodes**	**Time**	**Nodes**
M3	None	0.03	1	0.03	3
	B2 and V2	0.04	0	0.04	0
M4	None	0.09	7	0.09	7
	B2 and V2	0.03	3	0.05	3
M5	None	0.09	27	0.09	27
	B2 and V2	0.19	18	0.2	18
M5-1	None	0.07	20	0.07	20
	B2 and V2	0.18	17	0.16	17
M5-2	None	0.75	30	0.75	30
	B2 and V2	0.16	21	0.15	21
M6	None	0.5	34	0.76	93
	B2 and V2	0.39	23	0.38	19
M7	None	0.73	86	1.4	243
	B2 and V2	0.67	51	0.8	88
FO7	None	830	126741	740	124046
	B2 and V2	790	79539	510	66292
FO7-1	None	380	62512	790	147831
	B2 and V2	270	32141	400	49987
FO7-2	None	170	29112	180	32413
	B2 and V2	120	15137	180	21045
FO8	None	2000	284944	1700	219929
	B2 and V2	320	26613	2300	204989
O7	None	2700	452488	1800	285649
	B2 and V2	10000	1322262	4600	530465
O7-1	None	1400	252366	630	98751
	B2 and V2	790	103656	690	84619
O7-2	None	1300	224042	820	136410
	B2 and V2	1200	152607	3100	413497

permits a simple root node analysis to determine the effectiveness of including valid inequalities, we will focus on only the position p-q strategy for symmetry breaking for the remainder of this chapter. Furthermore, the additional enhancements proposed in Section 5.6 will be used to augment the current best model revealed thus far, which uses our proposed area constraints (with 20 tangential supports per department), the position p-q symmetry breaking strategy, and the valid inequalities B2 and V2.

5.5.3 Effect of Valid Inequalities on the FLP2+ Model

We now take a brief aside to explore the effect of the inequalities proposed by Meller *et al*. on the basic FLP2 model. Given that superior results were attained with our model when we used only constraints B2 and V2, as a point of interest, we next performed a small experiment to see if a similar result would have occurred without using our proposed area constraints. Toward this end, we constructed the basic FLP2 model of (5.2), retaining the area constraints proposed by Meller *et al*., and replacing (5.2d,e) with the single equality (5.3d). To this model, we added only the inequalities B2 and V2, in place of the entire set proposed in FLP2+. In addition, we evaluated using both the position p-q and position q symmetry-breaking approaches. Table 5.13 displays the results obtained, where we have focused on only three problems (M7, FO7, and O7). We note that the performance of FLP2+ corresponds to the first line for each problem, in which problem symmetry was broken using the position q method, and all valid inequalities were included. The striking result is that for each of the three problems we investigated, the FLP2+ model performed the worst out of the five alternatives that we explored. The best performance seems to come from using the position p-q symmetry breaking technique with no valid inequalities, although including B2 and V2 also performed well for FO7. We recall that the B2 and V2 inequalities were already shown to be rather ineffective on Problem O7 in the previous section, yet their inclusion still significantly reduces computational effort as compared to using all the proposed valid inequalities of FLP2+. The results also demonstrate a significant reduction in the number of nodes enumerated for the position p-q over the position q method in all instances. Given that Meller *et al*. demonstrated significant computational gains when using FLP2+ over FLP2, we conclude that even more improvement would have been realized had they explored the effect of using only a subset of their proposed inequalities, as well as alternative methods for reducing problem symmetry. However, for each of these three problems, their model would still have led to solutions having a maximum error in departmental area of over 5%. Furthermore, as demonstrated previously, our proposed area constraints produce a structure that results in a dramatic decrease in solution effort over that when using the area approximation constraints of Meller *et al*.

5.6 Convex Hull Representations of the Separation Constraints

In this section, after reviewing the traditional constructs for preventing departmental overlaps, we propose two new methods for modeling the associated disjunctive relationships. We then show that these formulations exhibit partial convex hull properties. Additionally, we consider several implementation strategies for each of these modeling approaches, in order to ascertain computationally favorable options for solving the overall facility layout problem.

Table 5.13: Effect of Valid Inequalities on FLP2+.

Problem	Symmetry	Valid Inequalities	Time	Nodes
	Position q	All	670	5757
	Position p-q	None	96	32508
M7	Position q	None	230	58802
	Position p-q	B2, V2	210	364188
	Position q	B2, V2	470	647774
	Position q	All	10000	79557
	Position p-q	None	1900	732686
FO7	Position q	None	5000	1185908
	Position p-q	B2, V2	1800	507030
	Position q	B2, V2	9500	1796613
	Position q	All	60000	393187
	Position p-q	None	8300	2900017
O7	Position q	None	26000	5230940
	Position p-q	B2, V2	25000	6299996
	Position q	B2, V2	39200*	2244477*

* This problem ran out of memory with an 18% integrality gap.

5.6.1 Traditional Formulation of the Separation Constraints

Traditionally, the departmental separation constraints have been modeled through (5.2d), (5.2f), (5.2i), and (5.2l). In order to infer some of the properties associated with this set of constraints, we first introduce the following notation:

$$\theta_{ij}^s = c_j^s - c_i^s \text{ and } \phi_{ij}^s = \ell_i^s + \ell_j^s \text{ for } s = x, y, \ \forall \ i < j . \tag{5.17}$$

Now, the separation disjunction characterized by (5.2d) at equality, (5.2f), (5.2i), and (5.2l) for any $i < j$ can be equivalently modeled as:

$$\theta_{ij}^s \geq \phi_{ij}^s - L^s(1 - z_{ij}^s) \qquad \forall s \tag{5.18a}$$

$$-\theta_{ij}^s \geq \phi_{ij}^s - L^s(1 - z_{ji}^s) \qquad \forall s \tag{5.18b}$$

$$(\ell b_i^s + \ell b_j^s) \leq \phi_{ij}^s \leq (ub_i^s + ub_j^s) \qquad \forall s \tag{5.18c}$$

$$\sum_{s=x,y} (z_{ij}^s + z_{ji}^s) = 1 \tag{5.18d}$$

$$z \text{ binary.} \tag{5.18e}$$

We show in Proposition 5.1 that in the case when $ub_i^s + ub_j^s \leq L^s / 2 \ \forall s$, the continuous relaxation of (5.18) yields the convex hull of feasible solutions. However, if this condition is not satisfied, then additional tightening can be achieved.

Proposition 5.1. If $ub_i^s + ub_j^s \le L^s/2 \ \ \forall s$, then the continuous relaxation of (5.18) defines its convex hull.

Proof. It is sufficient to show that the extreme points of the set X, defined as the set of feasible solutions to (5.18) when (5.18e) is replaced by $z \ge 0$, have binary *z*-values. To prove this, we will show that under the stated condition, given any linear objective function $f_1\theta + f_2\phi + f_3 z$ that yields a unique optimum for the problem $\max\{f_1\theta + f_2\phi + f_3 z : (\theta, \phi, z) \in X\}$, we have that *z* is binary valued in this optimal solution. The foregoing problem can be re-stated as

$$\max_{(5.18d),\ z\ge 0} \{f_3 z + \max_{\theta,\phi}[f_1\theta + f_2\phi : (5.18a,b,c)]\}. \tag{5.19}$$

For any fixed z feasible to the outer optimization problem, the inner problem is given by

$$\begin{aligned}
&\text{maximize} && f_1\theta + f_2\phi && && (5.20a)\\
&\text{subject to} && \phi_{ij}^s - L^s(1 - z_{ij}^s) \le \theta_{ij}^s \le -\phi_{ij}^s + L^s(1 - z_{ji}^s) && \forall s && (5.20b)\\
& && (\ell b_i^s + \ell b_j^s) \le \phi_{ij}^s \le (ub_i^s + ub_j^s) && \forall s && (5.20c)
\end{aligned}$$

Given that $ub_i^s + ub_j^s \le L^s/2 \ \ \forall s$, for any ϕ feasible to (5.20c), the constraint (5.20b) always provides feasible bounds for θ_{ij}^s; that is,

$$\phi_{ij}^s - L^s(1 - z_{ij}^s) \le -\phi_{ij}^s + L^s(1 - z_{ji}^s) \qquad \forall s. \tag{5.21}$$

To see this, note that after regrouping terms, (5.21) corresponds to the restriction that

$$\phi_{ij}^s \le \frac{L^s}{2}[2 - (z_{ji}^s + z_{ij}^s)] \qquad \forall s. \tag{5.22}$$

Given that $ub_i^s + ub_j^s \le L^s/2 \ \ \forall s$ and noting (5.20c), we have that

$$\phi_{ij}^s \le ub_i^s + ub_j^s \le L^s/2 \le \frac{L^s}{2}[2 - (z_{ji}^s + z_{ij}^s)] \qquad \forall s,$$

where the right-most upper bound is satisfied since $z \ge 0$ and z is feasible to (5.18d). Consequently, we can rewrite (5.19) as follows:

$$\max_{(5.18d),\ z\ge 0} \{f_3 z + \max_{\phi:(5.18c)}[f_2\phi + \max_{\theta:(5.18a,b)}(f_1\theta)]\}. \tag{5.23}$$

Note that we can solve (5.23) by setting θ_{ij}^s for each *s* equal to its appropriate bound given in (5.18a,b), noting the coefficients of f_1 and then setting ϕ_{ij}^s for each *s* equal to its appropriate

bounds based upon the resulting objective coefficients. This reduces the problem (5.23) to effectively maximizing an affine function $f_4 z + f_5$, say, subject to (5.18d) and $z \geq 0$, as stated below:

$$\max\{(f_4 z + f_5 : \sum_{s=x,y} (z_{ij}^s + z_{ji}^s) = 1,\ z \geq 0\}. \tag{5.24}$$

By assumption, the solution to (5.19) (and therefore (5.24)) is unique, indicating that the solution lies at an extreme point of the feasible region of (5.24). Since (5.24) has purely binary vertices, this completes the proof. □

We emphasize that the separation embodied in (5.23) would not be possible without the assumption that $ub_i^s + ub_j^s \leq L^s / 2 \ \forall s$, since otherwise, values of ϕ feasible to (5.20c) alone could lead to inconsistent bounds for θ in the innermost optimization problem. In the next section, we develop a model for the separation constraints that retains the convex hull properties without such an assumption.

5.6.2 Alternative Formulation of the Separation Constraints

As we have shown in the previous section, the continuous relaxation of (5.18) can be tightened in certain situations. In this section, we consider an alternate set of separation constraints whose continuous relaxation captures the convex hull regardless of whether $ub_i^s + ub_j^s \leq L^s / 2$. Toward this end, let us use the notation of (5.17) and consider the separation disjunction for any $i < j$:

$$\bigvee_{s=x}^{y} (\theta_{ij}^s \geq \phi_{ij}^s) \vee (-\theta_{ij}^s \geq \phi_{ij}^s), \tag{5.25}$$

where $-(L^s - \ell b_i^s - \ell b_j^s) \leq \theta_{ij}^s \leq (L^s - \ell b_i^s - \ell b_j^s)$ and $(\ell b_i^s + \ell b_j^s) \leq \phi_{ij}^s \leq (ub_i^s + ub_j^s)$ for $s = x, y$. Defining

$$M^s = L^s + (ub_i^s - \ell b_i^s) + (ub_j^s - \ell b_j^s) \ \text{ for } s = x, y, \tag{5.26}$$

we can model this disjunction as follows:

$$\theta_{ij}^s \geq \phi_{ij}^s - M^s(1 - z_{ij}^s) \ \text{ for } s = x, y \tag{5.27a}$$
$$-\theta_{ij}^s \geq \phi_{ij}^s - M^s(1 - z_{ji}^s) \ \text{ for } s = x, y \tag{5.27b}$$
$$-(L^s - \ell b_i^s - \ell b_j^s) \leq \theta_{ij}^s \leq (L^s - \ell b_i^s - \ell b_j^s) \ \text{ for } s = x, y \tag{5.27c}$$
$$(\ell b_i^s + \ell b_j^s) \leq \phi_{ij}^s \leq (ub_i^s + ub_j^s) \ \text{ for } s = x, y \tag{5.27d}$$
$$\sum_{s=x}^{y} (z_{ij}^s + z_{ji}^s) = 1 \tag{5.27e}$$
$$z \text{ binary}. \tag{5.27f}$$

We can now use the GUB structured RLT process described in Sherali *et al*. (1998) to construct the convex hull of (5.27). Let us define the following set of continuous variables, given any $i < j$.

$$c_{ij}^s,\ c_{ji}^s,\ \ell_{ij}^s,\ \ell_{ji}^s,\ \Delta_{ij}^s,\ \text{and}\ \delta_{ij}^s,\ \text{for } s = x, y. \tag{5.28}$$

Applying conditional logic, along with an aggregation that maintains the convex hull representation, yields a reformulation of (5.27) shown below in (5.29). Propositions 5.2 and 5.3 below verify the validity and convex hull property of this representation, which introduces $6n(n-1)$ new (continuous) variables. For convenience, because of several equivalent reductions involved in deriving (5.29), we provide a self-contained proof independent of RLT constructs.

$$\ell_{ij}^s \le c_{ij}^s \le (L^s - \ell b_i^s - \ell b_j^s) z_{ij}^s \qquad \text{for } s = x, y \tag{5.29a}$$

$$\ell_{ji}^s \le c_{ji}^s \le (L^s - \ell b_i^s - \ell b_j^s) z_{ji}^s \qquad \text{for } s = x, y \tag{5.29b}$$

$$(\ell b_i^s + \ell b_j^s) z_{ij}^s \le \ell_{ij}^s \le (ub_i^s + ub_j^s) z_{ij}^s \qquad \text{for } s = x, y \tag{5.29c}$$

$$(\ell b_i^s + \ell b_j^s) z_{ji}^s \le \ell_{ji}^s \le (ub_i^s + ub_j^s) z_{ji}^s \qquad \text{for } s = x, y \tag{5.29d}$$

$$-(L^s - \ell b_i^s - \ell b_j^s)(1 - z_{ij}^s - z_{ji}^s) \le \Delta_{ij}^s \le (L^s - \ell b_i^s - \ell b_j^s)(1 - z_{ij}^s - z_{ji}^s) \qquad \text{for } s = x, y \tag{5.29e}$$

$$(\ell b_i^s + \ell b_j^s)(1 - z_{ij}^s - z_{ji}^s) \le \delta_{ij}^s \le (ub_i^s + ub_j^s)(1 - z_{ij}^s - z_{ji}^s) \qquad \text{for } s = x, y \tag{5.29f}$$

$$\theta_{ij}^s = c_{ij}^s - c_{ji}^s + \Delta_{ij}^s \qquad \text{for } s = x, y \tag{5.29g}$$

$$\phi_{ij}^s = \ell_{ij}^s + \ell_{ji}^s + \delta_{ij}^s \qquad \text{for } s = x, y \tag{5.29h}$$

$$\sum_{s=x}^{y} (z_{ij}^s + z_{ji}^s) = 1 \tag{5.29i}$$

$$z \text{ binary.} \tag{5.29j}$$

Proposition 5.2. Let ξ represent the set of variables listed in (5.28), and let θ, ϕ and z be vectors of the corresponding subscripted variables. Then, (5.27) and (5.29) are equivalent in the sense that for any (θ, ϕ, z) feasible to (5.27), there exists a ξ such that (θ, ϕ, z, ξ) is feasible to (5.29). Conversely, given any (θ, ϕ, z, ξ) feasible to (5.29), we have that (θ, ϕ, z) is feasible to (5.27).

Proof. Consider any (θ, ϕ, z) feasible to (5.27). Noting (5.27e), assume that $z_{ij}^x = 1$ and $z_{ji}^x = z_{ij}^y = z_{ji}^y = 0$. (The other three cases are similar.) Hence, from (5.27a), we have

$$\theta_{ij}^x \ge \phi_{ij}^x. \tag{5.30}$$

Now, in (5.29), let us select

$$c_{ij}^x = \theta_{ij}^x,\ \ell_{ij}^x = \phi_{ij}^x,\ c_{ji}^x = c_{ij}^y = c_{ji}^y = \ell_{ji}^x = \ell_{ij}^y = \ell_{ji}^y = \Delta_{ij}^x = \delta_{ij}^x = 0,\ \Delta_{ij}^y = \theta_{ij}^y,\ \text{and}\ \delta_{ij}^y = \phi_{ij}^y.$$

Then it is readily verified that (θ, ϕ, z, ξ) is feasible to (5.29), noting (5.30) and (5.27c,d).

Conversely, consider any feasible solution (θ, ϕ, z, ξ) to (5.29). Again, let us assume that $z_{ij}^x = 1$ and $z_{ji}^x = z_{ij}^y = z_{ji}^y = 0$, with the other three cases of 0-1 assignments to the *z*-variables via (5.29i) being similar. From (5.29c, d), we get

$$(\ell b_i^x + \ell b_j^x) \le \ell_{ij}^x \le (ub_i^x + ub_j^x), \text{ while } \ell_{ji}^x = \ell_{ij}^y = \ell_{ji}^y = 0. \tag{5.31}$$

Consequently, from (5.29a, b), we have

$$\ell_{ij}^x \le c_{ij}^x \le (L^s - \ell b_i^x - \ell b_j^x), \text{ while } c_{ji}^x = c_{ij}^y = c_{ji}^y = 0. \tag{5.32}$$

Furthermore, (5.29e, f) yield that $\Delta_{ij}^x = \delta_{ij}^x = 0$, while

$$-(L^s - \ell b_i^y - \ell b_j^y) \le \Delta_{ij}^y \le (L^s - \ell b_i^y - \ell b_j^y) \text{ and } (ub_i^y + ub_j^y) \le \delta_{ij}^y \le (ub_i^y + ub_j^y). \tag{5.33}$$

Finally, (5.29g, h) assert, using (5.30), (5.31) and (5.32), that

$$\theta_{ij}^x = c_{ij}^x \text{ and } \phi_{ij}^x = \ell_{ij}^x, \text{ while } \theta_{ij}^y = \Delta_{ij}^y \text{ and } \phi_{ij}^y = \delta_{ij}^y. \tag{5.34}$$

From (5.29i, j) and (5.31)-(5.34), we have that (5.27c-f) are satisfied. Moreover, (5.32) and (5.34) assert that (5.27a) holds true when $s = x$, while the remaining constraints in (5.27a,b) which require that $\phi_{ij}^x + \theta_{ij}^x \le M^x$, and $\phi_{ij}^y \pm \theta_{ij}^y \le M^y$ are implied by the bounds (5.27c, d). This completes the proof. □

Proposition 5.3. The continuous relaxation of (5.29) defines the convex hull of feasible solutions to (5.27).

Proof. Given the assertion of Proposition 5.2, it is sufficient to show that the extreme points of the set *X*, defined by (5.29a-i) along with $z \ge 0$, have binary values of *z*. Toward this end, we will show that the maximization of any linear objective function over *X* that yields a unique optimum $(\theta^*, \phi^*, z^*, \xi^*)$ in the notation of Proposition 5.2, necessarily has 0-1 values for z^*. Given any such linear program to maximize $g_1\theta + g_2\phi + g_3 z + g_4\xi$, say, subject to (θ, ϕ, z, ξ) in *X*, we can rewrite this problem as

$$\underset{z \ge 0, (5.21i)}{\text{maximize}} \{ g_3 z + \underset{(\theta, \phi, z)}{\text{maximize}} \{ g_1\theta + g_2\phi + g_4\xi : (5.29a) - (5.29h) \} \}. \tag{5.35}$$

The inner maximization problem can be solved as follows. First, using (5.29g,h), we can substitute θ and ϕ out of the problem. Next, note that Δ_{ij}^s and δ_{ij}^s, for $s = x, y$, can be set at their appropriate bounds in (5.29e,f), depending on the signs of the resulting objective coefficients. The remaining problem is separable in the sets of variables (c_{ij}^s, ℓ_{ij}^s) and (c_{ji}^s, ℓ_{ji}^s) for $s = x, y$, where the corresponding constraints in (5.29a-d) for each of these subproblems have all their right-hand sides scaled by z_{ij}^s or z_{ji}^s, respectively. Hence, by LP duality, $c_{ij}^s, \ell_{ij}^s, c_{ji}^s, \ell_{ji}^s$ for $s = x, y$, can each be obtained as linear functions of the *z*-variables. This means that the inner

maximization problem in (5.35) can be reduced to a linear function $g_5 z$, say, of z. Consequently, (5.35) reduces to the problem

$$\text{maximize}\{(g_3 + g_5) \cdot z : \sum_{s=x}^{y} (z_{ij}^s + z_{ji}^s) = 1,\ z \geq 0\}. \tag{5.36}$$

The optimum value z^* is therefore given by the solution to (5.36), which under the hypothesis of uniqueness and noting that (5.36) has binary vertices, asserts that z^* is binary valued. This completes the proof. □

We will investigate the computational effectiveness of this method after detailing an alternative formulation for the separation constraints in the next section. Throughout the remainder of the chapter, we refer to the model obtained by replacing (5.2f) with (5.29) as DJ1.

Remark 5.1. We note that by using the tighter bounds L^s, we could have directly constructed the convex hull of (5.18) to yield a tighter, though larger, representation than (5.29). This convex hull could have been derived by applying the GUB-specialized RLT process described in Sherali *et al.* (1998), but in this case, the simplification yields a larger representation than (5.29). Therefore, we postpone the task of evaluating this formulation for future research.

Observe also by the proof of Proposition 5.1 that if we had formulated (5.18) by using the weaker bounds M^s given by (5.26) in lieu of L^s, then the continuous relaxation of (5.18) would yield the convex hull representation whenever

$$ub_i^s + ub_j^s + \ell b_i^s + \ell b_j^s \leq L^s / 2 \quad \forall s. \tag{5.37}$$

However, whether (5.37) holds true or not, the representation (5.27) is tighter than (5.18) with L^s replaced by M^s since (5.27c) is not then implied by the latter. (Actually, (5.27) is related to (5.18) in the manner of having added (5.27c) that is implied by the continuous relaxation to (5.18), but then replacing L^s by M^s in (5.18a,b). However, (5.29) then tightens the resulting formulation by creating its convex hull representation.) Thus, we might expect (5.29) to be perhaps beneficial over (5.18), particularly when (5.37) does not hold true. □

5.6.3 A Distance-Based Formulation of the Separation Constraints

We now present an additional enhancement of the model FLP2 that uses the distance relationships themselves to develop a disjunctive formulation that would prevent departments from overlapping. Rather than beginning from the traditional FLP2 formulation presented in (5.2), we instead consider the basic FLP model presented in (5.1). For the Area Constraints in (5.1b), we continue to use the outer-linearization presented in Section 5.3. We focus here on an alternative representation of (5.1c) and (5.1d).

First, let us consider the *Separation Constraints* (5.1c), assuming that (5.1d) has been modeled as an *equality*, in contrast with the pair of *inequalities*

$$d_{ij}^s \geq c_i^s - c_j^s \text{ and } d_{ij}^s \geq c_j^s - c_i^s \quad \forall\ i < j, s. \tag{5.38}$$

In this case, we can model (5.1c) directly in terms of the d_{ij}^s-variables themselves via the disjunction

$$(d_{ij}^x \geq \ell_i^x + \ell_j^x) \vee (d_{ij}^y \geq \ell_i^y + \ell_j^y) \qquad \forall\ i < j. \tag{5.39}$$

For each $i < j$, let us define the binary variable

$$w_{ij} = \begin{cases} 1, \text{ if the separation between } i \text{ and } j \text{ is enforced along the } x\text{-direction} \\ 0, \text{ if the separation is enforced along the } y\text{-direction} \end{cases} \tag{5.40}$$

and let

$$Q_{ij}^s = \text{minimum}\{L^s, ub_i^s + ub_j^s\}\ \forall\ i < j, s. \tag{5.41}$$

Consider the following modeling of (5.39), $\forall\ i < j$, which is readily verified to be valid.

$$d_{ij}^x \geq \ell_i^x + \ell_j^x - (1 - w_{ij})Q_{ij}^x \tag{5.42a}$$
$$d_{ij}^y \geq \ell_i^y + \ell_j^y - w_{ij}Q_{ij}^y. \tag{5.42b}$$

Suppose that given any $i < j$, we define $\phi_{ij}^s = \ell_i^s + \ell_j^s\ \ \forall s$ as in (5.17), and construct the set

$$X_{ij} = \{(d_{ij}^x, d_{ij}^y, \phi_{ij}^x, \phi_{ij}^y, w_{ij}) : d_{ij}^x \geq \phi_{ij}^x - (1 - w_{ij})Q_{ij}^x \tag{5.43a}$$
$$d_{ij}^y \geq \phi_{ij}^y - w_{ij}Q_{ij}^y \tag{5.43b}$$
$$0 \leq \phi_{ij}^s \leq Q_{ij}^s \forall s,\ d_{ij}^s \geq 0 \forall s,\ w_{ij} \text{ binary}\}. \tag{5.43c}$$

Let us denote $\overline{X}_{ij}$ to be the continuous relaxation of X_{ij} in which the binary restriction on w_{ij} is replaced by $0 \leq w_{ij} \leq 1$. Proposition 5.4 asserts that $conv(X_{ij}) = \overline{X}_{ij}$, and so, any further tightening of (5.42) would need to involve more relationships than inherent within the representation (5.43). In essence, this would expand to the development of the foregoing section. Consequently, we model the separation constraints (5.1c) via (5.43) here, along with binary restrictions on the *w*-variables.

Proposition 5.4. $Conv(X_{ij}) = \overline{X}_{ij}$.

Proof. It is sufficient to show that w_{ij} is binary at each vertex of $\overline{X}_{ij}$. Toward this end, let us divide (5.43a) and (5.43b) by Q_{ij}^x and Q_{ij}^y, respectively, and accordingly, define $d_{ij}^{s'} = d_{ij}^s / Q_{ij}^s$ and $\phi_{ij}^{s'} = \phi_{ij}^s / Q_{ij}^s \forall s$. This yields an equivalent representation of $\overline{X}_{ij}$ via the constraints

$$d_{ij}^{x'} \geq \phi_{ij}^{x'} - (1 - w_{ij}),\ d_{ij}^{y'} \geq \phi_{ij}^{y'} - w_{ij} \tag{5.44a}$$

$$0 \leq \phi_{ij}^{s'} \leq 1\ \forall s,\ d_{ij}^{s'} \geq 0,\ \ 0 \leq w_{ij} \leq 1. \tag{5.44b}$$

Noting the total unimodularity (see Bazaraa *et al*., 1990) of the constraint set (5.44), we have that w_{ij} is binary at each extreme point, and this completes the proof. □

Remark 5.2. Recall that one of the two particularly effective valid inequalities from the FLP2+ model was the constraint referred to as V2, reproduced here for convenience.

$$d_{ij}^{s} \geq \ell_i^s + \ell_j^s - \min\{ub_i^s + ub_j^s, L^s\}(1 - z_{ij}^s - z_{ji}^s)\,. \tag{5.45}$$

Noting the definitions in (5.40) and (5.41), we can see that (5.45) directly corresponds to our representation (5.42). Since we have shown that (5.42) captures the convex hull of feasible solutions for the disjunction in (5.39) in the sense of Proposition 5.4, we have gained insight into why the constraint set V2 was particularly helpful in tightening the FLP2 formulation. Furthermore, the constraint set B2, which was also effective in tightening the FLP2 formulation, can also be stated in terms of w_{ij} as follows:

$$d_{ij}^{x} \geq (\ell b_i^x + \ell b_j^x) w_{ij} \tag{5.46a}$$

$$d_{ij}^{y} \geq (\ell b_i^y + \ell b_j^y)(1 - w_{ij})\,. \tag{5.46b}$$

Therefore, we can also include this constraint set in the proposed model. □

Next, let us proceed to model (5.1d). Toward this end, for each $i < j$, let us define the binary variables $y_{ij}^s\ \forall s$ as

$$y_{ij}^s = \begin{cases} 1, \text{ if } c_i^s \leq c_j^s \\ 0, \text{ if } c_i^s \geq c_j^s \end{cases} \tag{5.47}$$

with the choice of 0 or 1 being inconsequential when $c_i^s = c_j^s$. Furthermore, let us define an upper bound on d_{ij}^s as

$$U_{ij}^s = L^s - \ell b_i^s - \ell b_j^s\ \forall\ s\,. \tag{5.48}$$

Then, for each $i < j$, and each $s = x, y$, consider the following representation of (5.1d), where we have introduced a set of new continuous variables D_{ij}^s.

$$d_{ij}^s = c_i^s - c_j^s + 2D_{ij}^s \tag{5.49a}$$

$$0 \leq D_{ij}^s + (c_i^s - c_j^s) \leq U_{ij}^s (1 - y_{ij}^s) \tag{5.49b}$$

$$0 \le D_{ij}^s \le U_{ij}^s y_{ij}^s \tag{5.49c}$$

$$y_{ij}^s \text{ binary.} \tag{5.49d}$$

Proposition 5.5. For each $i < j$, and each $s = x, y$, the constraints (5.49) yield a valid representation of the relationship (5.1d).

Proof. When $y_{ij}^s = 1$, we have from (5.49b) that $D_{ij}^s = (c_j^s - c_i^s)$, which gives from (5.49a, c) that $0 \le d_{ij}^s = (c_j^s - c_i^s) \le U_{ij}^s$. Similarly, when $y_{ij}^s = 0$, we obtain $D_{ij}^s = 0$ from (5.49c), and (5.49a, b) yield $0 \le d_{ij}^s = c_i^s - c_j^s \le U_{ij}^s$. Hence, (5.49) is a valid representation of (5.1d). This completes the proof. □

Next, let us define $\theta_{ij}^s = c_j^s - c_i^s$ as in (5.17), and consider the following set based on (5.49).

$$\chi_{ij}^s = \{(d_{ij}^s, \theta_{ij}^s, D_{ij}^s, y_{ij}^s):$$

$$d_{ij}^s = 2D_{ij}^s - \theta_{ij}^s \tag{5.50a}$$

$$0 \le D_{ij}^s - \theta_{ij}^s \le U_{ij}^s (1 - y_{ij}^s) \tag{5.50b}$$

$$0 \le D_{ij}^s \le U_{ij}^s y_{ij}^s \tag{5.50c}$$

$$y_{ij}^s \text{ binary}\}. \tag{5.50d}$$

Then, the following result motivates the (unconventional) representation (5.49) of (5.1d), where $\overline{\chi}_{ij}^s$ is given by (5.50) with (5.50d) being replaced with $0 \le y_{ij}^s \le 1$.

Proposition 5.6. $Conv(\chi_{ij}^s) = \overline{\chi}_{ij}^s$.

Proof. Consider the nonsingular linear transformation

$$\alpha = D_{ij}^s / U_{ij}^s,\ \beta = (D_{ij}^s - \theta_{ij}^s) / U_{ij}^s, \text{ and } \gamma = d_{ij}^s / U_{ij}^s. \tag{5.51a}$$

This has an inverse given by

$$D_{ij}^s = \alpha U_{ij}^s,\ \theta_{ij}^s = (\alpha - \beta) U_{ij}^s, \text{ and } d_{ij}^s = \gamma U_{ij}^s. \tag{5.51b}$$

Consequently, under (5.51), $\overline{\chi}_{ij}^s$ is equivalently transformed into the following set, where there is a one-to-one preservation of extreme points because of the nonsingularity of (5.51).

$$\overline{\chi}_{ij}^s = \{(\alpha, \beta, \gamma, y_{ij}^s):$$

$$-\alpha - \beta + \gamma = 0 \tag{5.52a}$$

$$\beta + y_{ij}^s \le 1 \tag{5.52b}$$

$$\alpha \quad - y_{ij}^s \le 0 \tag{5.52c}$$

$$(\alpha, \beta) \ge 0,\ 0 \le y_{ij}^s \le 1\}. \tag{5.52d}$$

Noting the total unimodularity (see Bazaraa *et al.*, 1990) of (5.52), we have that y_{ij}^s is binary at all extreme points of $\bar{\chi}_{ij}^s$, and this completes the proof. □

Upon eliminating the variables D_{ij}^s using (5.49a) and noting the definitions in (5.40) and (5.46), the complete representation of the proposed model can be obtained as (5.53) below, where the subscripts *p* and *q* refer to the two departments whose orientation is fixed in order to reduce problem symmetry. Naturally these related constraints are omitted whenever we have any fixed departments, and also in this case, the corresponding centroidal variables are fixed in value. We refer to this model as **DJ2**.

DJ2: Minimize $\sum_{(i,j)\in P} f_{ij}(d_{ij}^x + d_{ij}^y)$

$$d_{ij}^x \ge \ell_i^x + \ell_j^x - (1 - w_{ij})Q_{ij}^x \quad \forall i < j \tag{5.53a}$$

$$d_{ij}^y \ge \ell_i^y + \ell_j^y - w_{ij}Q_{ij}^y \quad \forall i < j \tag{5.53b}$$

$$0 \le d_{ij}^s + c_i^s - c_j^s \le 2U_{ij}^s(1 - y_{ij}^s) \quad \forall i < j, s \tag{5.53c}$$

$$0 \le d_{ij}^s - c_i^s + c_j^s \le 2U_{ij}^s y_{ij}^s \quad \forall i < j, s \tag{5.53d}$$

$$d_{ij}^x \ge (\ell b_i^x + \ell b_j^x)w_{ij} \quad \forall i < j \tag{5.53e}$$

$$d_{ij}^y \ge (\ell b_i^y + \ell b_j^y)(1 - w_{ij}) \quad \forall i < j \tag{5.53f}$$

$$c_p^s \le c_q^s \quad \forall s \tag{5.53g}$$

$$\sum_{s=x}^{y}(c_q^s - c_p^s) \ge \min\{\ell b_p^x + \ell b_q^x, \ell b_p^y + \ell b_q^y\} \tag{5.53h}$$

$$\ell_i^s \le c_i^s \le L^s - \ell_i^s \quad \forall\ i, s \tag{5.53i}$$

$$\ell b_i^s \le \ell_i^s \le u b_i^s \quad \forall\ i, s \tag{5.53j}$$

$$d_{ij}^s \ge 0 \quad \forall i < j, s \tag{5.53k}$$

$$y_{ij}^s \text{ binary } \forall i < j, s \tag{5.53l}$$

$$w_{ij} \text{ binary } \forall i < j. \tag{5.53m}$$

Note that the formulation in (5.53) yields $3n(n-1)/2$ binary variables, which is similar to the formulation of Meller *et al*. when (5.2d) is written as an equality and one of the four binary variables is eliminated for each $i < j$. Furthermore, it contains the same number of continuous variables. However, our model representation captures certain partial convex hull characterizations and imparts a different structure that is worth evaluating computationally.

5.6.4 Computational Analysis of the Alternative DJ1 and DJ2 Formulations

We initially evaluated the performance of using the DJ1 and DJ2 formulations in their entirety as presented in the foregoing sections. Very early on, however, it became quite clear that these formulations would lead to a dramatic increase in solution time as compared with the model that uses only the valid inequalities B2 and V2, in combination with our proposed area and symmetry breaking constraints. For example, in considering the total solution time for all of the FO problems, the solution time increased over eight times when using DJ1 and over thirty times when using DJ2. We attribute this dramatic increase to the large increase in problem size for DJ1. For DJ2, however, we suspect that the elimination of the SOS constraints (5.3d) from the model formulation, which most solvers exploit to make more efficient specialized branching decisions, is responsible for the increase in solution effort.

We therefore considered several alternative strategies in order to impart some of the tightness accruing from these new formulations, while limiting the increase in problem size and retaining the SOS constraints of the previous models. As noted earlier, (5.29) presents a tighter formulation of the disjunctive constraints presented in (5.2f), and therefore the constraints (5.2f) are replaced by the constraints (5.29) in the model DJ1. Rather than using the complete representation DJ1, we considered the option of using (5.29) only for one pair of departments, taken as the positively interacting (non-fixed) pair having the largest total area, and retaining (5.2f) for all other pairs. Similarly, we implemented the representation DJ2 for only one pair of departments. That is, using the traditional FLP2 model, we replaced the distance relationships in (5.2g) with those in (5.53) (including the valid inequalities (5.53e,f)) for only one pair of departments. In so doing, we defined the variables w_{ij} and y_{ij}^s only for the key (i, j) pair for which (5.53) is constructed, while the variables z_{ij}^s were defined and used to represent the separation relationships for all the other (i, j) pairs as before. We note that this model defines the variables d_{ij}^s for only those pairs $(i, j) \in P$, and also retains the SOS structure of the model for all but the single pair of departments identified above for implementing (5.53).

As an additional alternative to DJ2, we considered translating the implied upper bounds on the d_{ij}^s variables in (5.53c,d) to conform with the definitions of the z_{ij}^s variables in order to derive a new class of valid inequalities. Recalling the definitions (5.47) and (5.48), constraints (5.53c,d) induce the derivation of the following relationships that can be readily verified to be valid.

$$d_{ij}^s \le c_i^s - c_j^s + 2U_{ij}^s(1 - z_{ji}^s) \quad \forall i < j, s \tag{5.54a}$$

$$d_{ij}^s \le c_j^s - c_i^s + 2U_{ij}^s(1 - z_{ij}^s) \quad \forall i < j, s. \tag{5.54b}$$

We refer to this class of upper bounding valid inequalities as *UB inequalities*, and we discuss below its effect when incorporated within the previously derived models.

Tables 5.14 and 5.15 present the results of our computational analysis. We note that all of the M problems continued to be solved in under one second for each of the studied alternatives. For this reason, we focus only on the FO and O problems in this analysis. Table

Table 5.14: Effect of the New Disjunctive Formulations and the UB Inequalities on the Solution Effort.

		UB-Inequalities			
		None		All	
Problem	Model	Time	Nodes	Time	Nodes
FO7	No DJ	790	79539	430**	40175
	DJ1 for 1 pair	320*	32604	760	66182
	DJ2 for 1 pair	480	54947	460	52371
FO7-1	No DJ	270	32141	200**	22715
	DJ1 for 1 pair	240	29144	280	32629
	DJ2 for 1 pair	190*	24993	330	42798
FO7-2	No DJ	120	15137	65*	8099
	DJ1 for 1 pair	91	12006	87	10670
	DJ2 for 1 pair	80**	11270	83	11364
FO8	No DJ	320*	26613	450**	33837
	DJ1 for 1 pair	510	47535	800	68014
	DJ2 for 1 pair	810	88097	750	72872
O7	No DJ	10000	1322262	3000**	313585
	DJ1 for 1 pair	3800	466459	3600	378341
	DJ2 for 1 pair	6100	820029	2500*	320297
O7-1	No DJ	790*	103656	900	103903
	DJ1 for 1 pair	830**	91152	1700	160084
	DJ2 for 1 pair	1200	161309	1500	188576
O7-2	No DJ	1200**	152607	3900	488526
	DJ1 for 1 pair	1400	166682	1500	168688
	DJ2 for 1 pair	890*	126146	2200	266564

* Minimum solution time for this problem instance.
** Second smallest solution time for this problem instance.

Table 5.15: Total Time and Total Ranking for Disjunctive Models.

	UB-Inequalities			
	None		All	
Model	Solution Time	Time Ranking	Solution Time	Time Ranking
No DJ	13490	26	8945	19+
DJ1 for 1 pair	7191*	21++	8727	32
DJ2 for 1 pair	9750	22	7823**	27

* Minimum total solution time.
** Second smallest total solution time.
\+ Minimum total rank-sum.
++ Second smallest total rank-sum.

5.14 compares the results for each problem using each of the six techniques composed by using neither DJ1 nor DJ2 (referred to as *No DJ*), DJ1 for the single identified pair, and DJ2 for the single identified pair, each with or without the class (5.54) of UB inequalities. Here we have used the position p-q symmetry breaking technique for each of the disjunctive models, and the results for the No DJ - No UB inequalities case correspond to the results displayed previously in Table 5.12. The results indicate that, in general, the disjunctive enhancements are effective in decreasing the solution effort, although some of the disjunctive techniques are not as effective as others. Note that each of the disjunctive enhancements succeeded in significantly reducing the solution time for problem O7, which had previously exhibited a substantial increase in effect upon *including* the valid inequalities B2 and V2. Of the proposed disjunctive methods, we see that using DJ1 for one pair of departments has the lowest total solution time, while using the UB inequalities along with the No DJ option provides the lowest total rank-sum when the methods are ranked in increasing order of solution times. For this reason, we will focus on only these two methods in the evaluation of the three challenging problems (O8, FO9, O9) that we analyze in the next section.

However, before proceeding, it might be instructive to reflect on why the class of UB inequalities proves to be effective, although the objective function is attempting to *minimize* the weighted sum of distances. The reason for this is that in concert with the other problem constraints and valid inequalities that impose lower bounds on these distance variables, the UBinequalities induce additional relationships that must be satisfied (so that the lower bounding expressions are less than or equal to the corresponding upper bounding expressions). Evidently, these additional implied relationships help further tighter the model representation.

5.7 Computational Results for the Most Challenging Test Problems

Throughout the previous sections, we have outlined and evaluated a series of proposed enhancements for the MIP formulation of the facility layout problem. At this point, we turn our attention to the three larger problems that remained previously unsolved in the literature using the FLP2+ model. Having narrowed our focus to only two potential models, we now solve these three problems with each of these models. In both of the proposed models, we use the proposed area constraints along with 20 tangential supports, include the valid inequalities B2 and V2, and reduce problem symmetry using the position p-q approach. In the first model, we employ the disjunction DJ1 for one pair of departments as identified in Section 5.6.4, while for the other model we include the class of UB-inequalities with no other disjunctive enhancements. The results of this analysis are presented in Table 5.16, where for the sake of comparison, we also display the results obtained using the previously best FLP2+ model.

We first note that both of our proposed models were able to solve Problem FO9, using a 0.01% optimality tolerance, within the allowable limits on time (24 hours) and tree memory (390 MB), even though this problem had been previously unsolved in the literature. Furthermore, both of our models obtained optimal solutions that had a maximum error in departmental area of 0%. That is, each of the departments *exactly* met the proposed area requirements. On the other hand, the FLP2+ model was terminated (upon reaching the 24-hour time limit) with a 10.14% optimality gap, and its best-known integer solution had a maximum error of 4.62% with respect

Table 5.16: Accuracy and Solution Effort for the More Challenging Test Problems.

Problem	Model	Best Integer Solution	Optimality Gap (%)	Maximum Error (%)	Solution Time	Number of Nodes
FO9	FLP2+	23.35	10.14	4.76	86400*	66386
	DJ1 for 1 pair	23.46	0.00	0.00	5900	407779
	UB-Inequalities	23.46	0.00	0.00	11000	625275
O8	FLP2+	248.00	26.45	4.96	86400*	107334
	DJ1 for 1 pair	251.65	15.32	0.04	37060+	2423000
	UB-Inequalities	257.52	22.34	0.03	36000+	2039034
O9	FLP2+	277.76	40.00	7.14	86400*	50442
	DJ1 for 1 pair	269.49	32.92	0.03	33000+	1985589
	UB-Inequalities	270.71	35.06	0.10	32000+	1890026

* Terminated due to 24-hour time limit.
\+ Terminated when memory requirements for the search tree reached 390 MB.

to departmental areas. In addition to having a lower quality of solution as compared to our proposed models, we observe that the FLP2+ had a solution time of over fourteen times that obtained when using the model with DJ1 for one pair of departments, and this time would have been even larger if not for the time limit that we imposed.

While neither of our proposed models could solve Problems O8 and O9 to exact optimality, they did substantially reduce the optimality gap at termination. We note that both of our models were terminated due to the amount of memory required for the search tree, while the FLP2+ model was terminated due to the 24-hour time limit. As such, the FLP2+ model was run for an average of 2.5 times as long as the proposed models. Nonetheless, the model using the UB Inequalities reduced the optimality gap of the FLP2+ solution by an average of 13.93%, while the model using DJ1 for one pair of departments reduced the gap by 29.65%. (The optimality gaps for FLP2+ for the problems O8 and O9 were 26.45% and 40%, respectively, while the optimality gaps for these test instances using the better of methods (DJ1 for once pair of departments) were 15.44% and 32.92%, respectively, at termination.) Furthermore, both of our proposed models led to a dramatic reduction in the maximum error for departmental areas. Therefore, even for the problems that could not be solved to optimality, both of our proposed models still demonstrated significant advantages over the FLP2+ model.

Upon examining the results for the three more challenging problems, it is clear that the model using DJ1 for one pair of departments outperforms the model using the UB Inequalities. While each of these models solved Problem FO9 to optimality, the UB-Inequalities solution time was nearly twice that of the DJ1 model. For each of the two problems (O8 and O9) that could not be solved to optimality, the DJ1 model provided tighter bounds at termination than the UB-Inequalities model. Considering the performance on these more challenging problems, coupled with the results exhibited on the previous test problems, we recommend the model using DJ1 for one pair of departments as the most effective of our proposed models. We conclude by noting that the initial evaluation of the complete DJ1 model led to very discouraging results. Only upon experimenting with its implementation by applying it to only one pair of departments was the strength of this formulation realized.

5.8 Conclusions

In this chapter, we have developed a variety of enhancements for an MIP formulation of the facility layout problem. Through a series of computational tests performed on a set of problems from the literature, we have shown that our proposed enhancements are very effective in increasing the accuracy of solutions while simultaneously decreasing solution effort. Our computational analysis has demonstrated that the proposed area constraints systematically drive the maximum error in departmental area to zero as the number of tangential supports increase. Additionally, we have proposed a new symmetry breaking approach that reduces computational effort, particularly when employed in conjunction with effective valid inequalities and disjunctive models. We have also conducted a thorough analysis of previously proposed valid inequalities for this problem, which has revealed that retaining only a limited subset of these inequalities can significantly decrease the solution time. Finally, we have examined several alternative methods for modeling the disjunctive relationships that prevent departments from overlapping, and we have shown that these characterizations capture certain partial convex hull properties and induce additional useful classes of valid inequalities.

Our computational analysis indicates that the best performance was obtained using the model DJ1 for one pair of departments. This model used the tightened bounds (5.8) on the half-length and half-width of each department, as well as the area constraints (5.9) – (5.11) with 20 tangential supports per department. The symmetry of the problem was reduced using the Position p-q strategy as presented in (5.15) - (5.16), and the valid inequalities B2 and V2 (5.3j,k) were included for all positively interacting pairs of departments. In addition, the disjunctive relationship of (5.29) was included for one key pair of departments, taken as the positively interacting pair having the largest total area. The next most effective model, denoted as UB Inequalities, used the same area constraints, symmetry breaking approach, and valid inequalities as the foregoing model. However, rather than including the DJ1 representation for one pair of departments, this model included the UB inequalities (5.54) for all positively interacting departments. Recalling that these UB inequalities were derived from the disjunctive representation DJ2, we see that the two best performing models were obtained by experimenting with alternative representations for the DJ1 and DJ2 formulations, which each led to increased solution time when applied in their original form.

As a final comparison between the performance of our proposed model (DJ1 for one pair of departments) and FLP2+, Table 5.17 summarizes the factor of improvement (corresponding FLP2+ value divided by our model value) in the optimal objective value, the maximum error, and the solution time for all the test problems. Note that in some cases, our proposed model leads to an increase in the optimal objective value, since the optimal solution produced by the FLP2+ turns out to be infeasible to our more accurate area constraints. However, the maximum departmental error and solution time of our proposed model are *dramatically* smaller than those achieved using the FLP2+ model. Considering all of the test problems evaluated throughout this chapter, the solutions obtained using the FLP2+ model have maximum errors of 13 to 904 timesas large as those obtained using our proposed model. Additionally, for all problems that were solved by FLP2+ within the 24-hour time limit, the solution time for the FLP2+ model was cut by a factor of 6 to 1456 times using our proposed model.

Table 5.17: Factor of Improvement over FLP2+.

Problem	Objective	Error	Solution Time
M3	1.04	71.62	8.67
M4	1.04	71.62	6.60
M5	n/a[1]	n/a[1]	13.81[1]
M5-1	1.25	75.58	14.29
M5-2	1.46	13.71	27.50
M6	1.14	18.38	102.56
M7	1.22	25.69	1456.52
FO7	1.18	137.61	31.25
FO7-1	0.99	15.52	15.42
FO7-2	1.00	904.46	16.48
FO8	1.18	18.00	129.41
FO9	1.00	n/a[2]	2.33
O7	1.15	112.98	15.79
O7-1	1.06	17.50	20.48
O7-2	1.02	46.66	7.14
O8	0.99	124.09	2.33
O9	1.03	278.91	2.62

[1] This problem was found infeasible by FLP2+ but not with our model.
[2] Using our proposed method, the solution had a maximum error of 0%.

Our computational analysis has amply demonstrated that the proposed enhancements for FLP2 provide a dramatic decrease in solution effort while providing a more accurate representation of the underlying problem. We note that although substantial computational advances have been made, several moderately large sized problems cannot yet be solved to optimality within a reasonable amount of time. For this reason, we believe that further research breakthroughs are needed in this area. Although our improved model representation can enable the solution of larger instances to reasonable tolerances of optimality, further improvements can reduce the latter total for relatively larger sized problems. Throughout this chapter, we have considered a series of proposed enhancements in a sequential manner. We recommend that any future research into this or other difficult MIP problems should take a similar approach, since our computational analysis revealed that many previously proposed valid inequalities led to an increase in computational effort, rather than the decrease that was surmised, and other approaches that could have been dismissed when applied as developed within our framework, turned out to be beneficial when implemented in a modified or reduced fashion.

Chapter 6: Conclusions and Future Research

While efficient solution techniques for linear and convex programming are well-known, the most pressing challenge to the optimization community is to develop efficient solution techniques for the class of nonconvex optimization problems. These problems remain difficult to solve to optimality, despite advances in computer processing speed and memory. Typically, both continuous and discrete types of nonconvex problems are solved through the same types of enumerative techniques. Without tight bounds, the search process would hopelessly continue to explore vast extents of non-improving areas of the solution space and thereby dramatically increase computational effort. Therefore, it is imperative to employ both general-purpose and problem-specific techniques, in conjunction with existing methods, to develop tight model formulations for all classes of nonconvex optimization problems.

This dissertation has focused on a set of general and specific problems in nonconvex optimization, providing theoretical developments that, in turn, have led to more efficient solution techniques. This dissertation can generally be separated into three major areas, each dealing with a different type of nonconvex optimization problem. Each of these endeavors has sought to combine traditional and modern optimization techniques in novel ways in order to create even more efficient solution strategies. The first portion on this research uses concepts from the recently emerging field of semidefinite programming to develop a new class of cutting planes that can be used to enhance general RLT formulations. The second area combines traditional Benders' decomposition techniques with modern RLT and lift-and-project cutting planes in order to develop a solution technique for stochastic integer programs and other suitable discrete optimization problems. The final part of this dissertation uses the concepts of outer-linearizations, symmetry breaking techniques, and disjunctive programming to tighten an MIP formulation for the facility layout problem. Throughout each of these endeavors, we have relied on problem-specific and general purpose approaches, in combination with a variety of optimization techniques, in order to develop solution methodologies that significantly advance the state-of-the-art.

The first part of the dissertation develops a mechanism to tighten RLT-based relaxations by importing concepts from semidefinite programming (SDP), leading to a new class of *semidefinite cutting planes*. Given an RLT relaxation, the usual nonnegativity restriction on the matrix of RLT product variables is replaced by a constraint that the matrix of variables remain positive semidefinite. Instead of relying on specific SDP solvers, the definition of positive semidefiniteness is used to re-write the semidefinite restriction as an infinite set of linear restrictions. This enables the problem to be written as a (semi-infinite) linear programming representation, which can be solved using traditional optimization software. *This research represents the first time that the semidefinite restriction has been used to derive valid linear inequalities, thereby providing the tightness of an SDP formulation in a framework that is more amenable to optimization techniques. In addition, this research provides a theoretical extension of the semidefinite concept to matrices of dimension greater than two for the first time in the published literature.*

In order to implement the semidefinite cutting plane solution strategy, the infinite set of constraints is initially relaxed, and members of this set are generated as needed via a polynomial time separation routine. In essence, this process yields an RLT relaxation that is augmented with valid inequalities, which we call *semidefinite cuts*. The general concept of the proposed solution strategy is specialized for the problem of minimizing a nonconvex quadratic objective function over a simplex. The algorithm has been implemented in C++, using CPLEX callable routines to solve the linear programming problems. In addition, two types of semidefinite restrictions have been explored, along with several implementation strategies, to further improve the solution technique. In an experiment to evaluate the effectiveness of the cuts in tightening the lower bound, the semidefinite cuts were shown to provide up to a 65% increase in the bound provided by using RLT alone. When implemented within a branch-and-bound framework to find a global optimum, the semidefinite cuts led to *dramatic* improvements over the performance of using RLT alone. In problems containing 10, 20, and 30 variables, the semidefinite cuts reduced the size of the enumeration tree by 95%, 94%, and 94%, while the overall solution time was reduced by 67%, 64%, and 55%, respectively. Furthermore, while both the RLT and semidefinite cut techniques were able to solve all of the 10-variable problems to optimality, there were several larger problem instances that were solved to optimality when using the semidefinite cuts, but could not be solved when using RLT alone. *Overall, the computational results indicated that the cutting plane algorithm provides a significant tightening of the lower bound obtained by using RLT alone. Moreover, when used within a branch-and-bound framework, the proposed methodology significantly reduce the effort required to obtain globally optimal solutions. As such, the results of this research suggest a new technique that can be used to enhance solvability for many classes of nonconvex optimization problems.*

The second part of the dissertation develops a modification of Benders' decomposition method, using concepts from RLT and lift-and-project cuts, in order to design a solution strategy for discrete optimization problems, such as those that arise in the case of two-stage stochastic programs with integer recourse. Stochastic programs are linear programs where the set of first-stage decisions is made before a realization of the environment is revealed, where the latter occurs according to some probabilistic distribution. The second-stage variables determine the best action to compensate for the ensuing effect of the environment. Stochastic programs that contain purely continuous variables are typically solved using Benders' decomposition, an iterative strategy in which information is passed back and forth between a master problem (which involves only first-stage variables) and a set of subproblems (which couple the first- and second- stage variables for each possible outcome of the environment). In the presence of (mixed-) integer second-stage variables, however, Benders' decomposition cannot be applied in the traditional sense. This research suitably modifies Benders' decomposition to be applicable for the case of problems that decompose into discrete subproblems. The proposed procedure is based on sequentially generating cutting planes to approximate the solution of the subproblems in the process of deriving valid Benders' cuts for the master problem. In addition, the procedure is modified to perform even more efficiently in the case of stochastic programs, by exploiting the dual angular structure that they possess. The key idea is to solve the subproblems using an RLT or lift-and-project cutting plane scheme, and to generate and store the cuts as *functions of the first-stage variables*. Hence, these cutting planes can be re-used from one subproblem solution to the next simply by updating the values of the first-stage decisions. The proposed Benders' cuts also recognize these RLT or lift-and-project cuts as functions of the first-stage variables, and

are hence shown to be globally valid, thereby leading to an overall finitely convergent solution procedure. *This research represents the first proposed methodology for solving stochastic integer programs with mixed-integer recourse. Furthermore, it is the first time that Benders' decomposition has been effectively modified for handling discontinuous, nonconvex value functions. This method is expected to provide a seminal and viable solution technique for a class of problems that has not previously been adequately solved in the literature.*

The final part of the dissertation focuses on an improved mixed-integer programming (MIP) representation for the facility layout problem. Given a rectangular building and area (as well as certain aesthetic) requirements for each department, the problem is to determine the dimensions and location of each (rectangular) department within the building in order to minimize a total travel measure (number of trips times the distance) between all the departments. The distance between pairs of departments is measured as the rectilinear distance between the departmental centroids. Although the facility layout problem can be stated rather simply, it is extremely difficult to solve to optimality, for even small problem instances. The difficulty in this problem arises from the nonlinear area constraints for each department and the disjunctive constraints that no two departments can overlap. This research develops several model enhancements to produce *more accurate solutions* while *decreasing* the solution effort required. In order to approximate the area constraints, tangential supports are used to derive a polyhedral outer approximation of the nonlinear constraints, and this representation is shown to provide as tight an approximation as desired. In addition, valid inequalities are used to reduce problem symmetry and to impose implied upper bounds on the centroidal separations. Finally, several different formulations are developed for the disjunctive constraints that prohibit the departments from overlapping. These proposed enhancements have been evaluated, using an AMPL interface with CPLEX, and compared with published results to gauge their effectiveness. *The improved area constraints have yielded solutions that are within 0.5% of the target areas, while previously published models led to errors as high as 10%. Furthermore, as compared with previously published models, the new area constraints admitted solutions to some test problems that could not previously be solved, while reducing solution time by a factor of 1.28 to 291.3 for other problem instances. The additional improvements in the model have provided even greater reductions in computational effort, thereby yielding tremendous improvements in the solvability of this class of problems. The overall solution effort was reduced by a factor of 2.33 to 2.62 for the three most challenging problems and by a factor of 6.6 to 1456.5 for the remaining problems.*

Throughout this dissertation, we have developed a variety of novel solution techniques for several classes of nonconvex optimization problems. There are several ways in which the various concepts exposed in this research can be extended in the future. In the case of semidefinite cuts, it would be most interesting to develop and evaluate a technique for generating SDP cuts corresponding to higher level RLT relaxations. Additionally, we look forward to investigating how SDP cuts might impact the solution of other types of nonconvex optimization problems, such as polynomial programs and integer programming problems. It would also be beneficial to investigate the possibility of generating certain classes of SDP constraints *a priori*, rather than through cutting plane generation separation routine. The proposed Benders' methodology for discrete optimization problems also provides future research opportunities. The most important step following the proposed theoretical development is to conduct computational experiments to gauge the effectiveness of the proposed technique, particularly as applied to

stochastic integer programs with recourse. It would also be instructive to explore special structures that might be particularly amenable to our proposed approach. In the case of the facility layout problem, while we have demonstrated significant computational gains, the solution to even moderately large sized problems remains rather challenging. A further analysis into the polyhedral structure of these problems might provide additional computational benefits. Moreover, it might be worthwhile to learn from the successes in developing solution techniques for the facility layout problem, particularly with respect to the disjunctive formulations, and extend these ideas to other similar classes of mixed-integer programming formulations.

In conclusion, although we have developed some very promising solution techniques for several classes of nonconvex optimization problems, many more problems cannot yet be solved efficiently. The development of tight model representations and effective solution techniques for such classes of challenging problems along the lines exposed in this dissertation, offers a rich and exciting arena for future research.

References

W.P. Adams and H.D. Sherali, "Mixed-Integer Bilinear Programming Problems," *Mathematical Programming*, 59(3): 279-305, 1993.

S. Ahmed, M. Tawarmalani and N.V. Sahindis, "A Finite Branch and Bound Algorithm for Two-Stage Stochastic Integer Programs," Working Paper, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 2000.

F. Alizadeh, "Interior Point Methods in Semidefinite Programming with Applications to Combinatorial Optimization," *SIAM Journal of Optimization*, 5(1): 13-51, 1995.

C. Audet, P. Hansen, B. Jaumard and G. Savard, "Branch and Cut Algorithm for Nonconvex Quadratically Constrained Quadratic Programming," *Mathematical Programming*, 87(1): 131-152, 2000.

E. Balas, "Disjunctive Programming: Properties of the Convex Hull of Feasible Points," Technical Report MSRR-348, Management Sciences Research Group, Carnegie-Mellon University, Pittsburgh, PA, 1974.

E. Balas, "Disjunctive Programming: Cutting Planes from Logical Conditions," in *Nonlinear Programming*, Academic Press, New York, NY, 1975.

E. Balas, "Disjunctive Programming: Properties of the Convex Hull of Feasible Points," *Discrete Applied Mathematics*, 89: 3-44, 1998.

E. Balas, S. Ceria and G. Cornuejols, "A Lift-and-Project Cutting Plane Algorithm for Mixed 0-1 Programs," *Mathematical Programming*, 58: 295-324, 1993.

E. Balas and R.G. Jeroslow, "Strengthening Cuts for Mixed Integer Programs," Technical Report MSRR-359, Management Sciences Research Group, Carnegie-Mellon University, Pittsburgh, PA, 1975.

P. Banerjee, B. Montreuil, C.L. Moodie and R.L. Kashyap, "A Modeling of Interactive Facilities Layout Designer Reasoning Using Qualitative Patterns," *International Journal of Production Research*, 30: 433-453, 1992.

M.S. Bazaraa, H.D. Sherali and C.M. Shetty, *Nonlinear Programming Theory and Applications, 2nd Edition*, John Wiley & Sons. Inc., New York, NY, 1993.

J. Benders, "Partitioning Procedures for Solving Mixed-Variables Programming Problems," *Numerische Mathematik*, 4: 238-252, 1962.

S. Benson, Y. Ye and X. Zhang, "Mixed Linear and Semidefinite Programming for Combinatorial and Quadratic Optimization," Working Paper, Applied Mathematics and Computer Science, University of Iowa, Iowa City, IA 52242, 1998.

D. Bertsimas and Y. Ye, "Semidefinite Relaxations, Multivariate Normal Distributions, and Order Statistics," *Handbook of Combinatorial Optimization,* Du and Pardalos (eds.), 3: 1-19, 1998.

J.R. Birge and M.A.H. Dempster, "Stochastic Programming Approaches to Stochastic Scheduling," *Journal of Global Optimization*, 9(3-4): 417-451, 1996.

J.R. Birge and F.V. Louveaux, "A Multicut Algorithm for Two-Stage Stochastic Linear Programs," *European Journal of Operational Research*, 34(3): 384-392, 1988.

J.R. Birge and F.V. Louveaux, *Introduction to Stochastic Programming*, Springer, New York, NY, 1997.

S. Burer and R. Monteiro, "A Nonlinear Programming Algorithm for Solving Semidefinite Programs via Low-rank Factorization," Presented at the *ISMP Conference*, Atlanta, GA, 2000.

D.R. Carino, T. Kent, D.H. Meyers, C. Stacy, M. Sylvanus, A.L. Turner, K. Watanabe and W.T. Ziemba, "The Russell-Yasuda Kasai Model: An Asset/Liability Model for a Japanese Insurance Company Using Multistage Stochastic Programming," *Interfaces*, 24(1): 29-49, 1994.

C.C. Caroe and R. Schultz, "Dual Decomposition in Stochastic Integer Programming," *Operations Research Letters*, 24(1): 37-45, 1999.

C.C. Caroe and J. Tind, "A Cutting-Plane Approach to Mixed 0-1 Stochastic Integer Programs," *European Journal of Operational Research*, 101(2): 306-316, 1997.

C.C. Caroe and J. Tind, "L-Shaped Decomposition of Two-Stage Stochastic Programs with Integer Recourse," *Mathematical Programming*, 83(3): 451-464, 1998.

S. Chittratanawat and J.S. Noble, "An Integrated Approach for Facility Layout, P/D/ Location and Material Handling System Design," *International Journal of Production Research*, 37(3): 683-706, 1999.

R. Dakin, "A Tree Search Algorithm for Mixed Integer Programming Problems," *Computer Journal*, 8: 250-255, 1965.

H. Delmaire, A. Langevin and D. Riopel, "Skeleton-Based Facility Layout Design Using Genetic Algorithms," *Annals of Operations Research*, 69: 85-104, 1997.

Y. Ermoliev and R. J.-B. Wets, eds., *Numerical Techniques for Stochastic Optimization*, Springer-Verlag, New York, NY, 1988.

A. Geoffrion and R. McBride, "Lagrangean Relaxation Applied to Capacitated Facility Location Problems," *AIIE Transactions*, 10(1): 40-47, 1978.

M.C. Georgiadis, G. Schilling, G.E. Rotstein and S. Macchietto, "A General Mathematical Programming Approach for Process Plant Layout," *Computers and Chemical Engineering*, 23: 823-840, 1999.

F. Glover, "Polyhedral Annexation in Mixed Integer Programming," *Bulletin of the Operations Research Society of America*, 22 supp. 1: B123, 1974.

M. Goemans and D. Williamson, "Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming," *Journal of the Association for Computational Machinery*, 42(6): 1115-1145, 1995.

R. Gomory, "An Algorithm for the Mixed Integer Problem," RM-2597 Rand Corporation, July 1960.

C. Helmberg and F. Rendl, "Solving Quadratic (0-1)-Problems by Semidefinite Programs and Cutting Planes," *Mathematical Programming*, 82(3): 291-315, 1998.

S. Heragu and A. Kusiak, "Efficient Models for the Facility Layout Problem," *European Journal of Operational Research*, 53: 1-13, 1991.

J.L. Higle and S. Sen, "Stochastic Decomposition: An Algorithm for Two-Stage Linear Programs with Recourse," *Mathematics of Operations Research*, 16(3): 650-669, 1991.

J.L. Higle and S. Sen, "Conditional Stochastic Decomposition: An Algorithmic Interface for Optimization and Simulation," *Operations Research*, 42(2): 311-322, 1994.

J. Higle and S. Sen, *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programs*, Kluwer Academic Publishers, Boston, MA, 1996.

J.L. Higle and S. Sen, "The C^3 Theorem and a D^2 Algorithm for Large Scale Stochastic Integer Programming: Set Convexification," Working Paper, Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ 85721. (Also presented at the *17th International Symposium on Mathematical Programming*, Atlanta, GA, August 7-11, 2000.)

K. Hoffman and M. Padberg, "Improving LP-representations of zero-one linear programs for branch-and-cut," *Operations Research*, 3: 121-134, 1991.

R.G. Jeroslow, "Cutting-Plane Theory: Disjunctive Models," *Annals of Discrete Mathematics*, 1:293-330, 1977.

R.G. Jeroslow, "A Cutting Plane Game for Facial Disjunctive Programs," *SIAM Journal on Control and Optimization*, 18(3): 264-280, 1980.

P. Kall and S.W. Wallace, *Stochastic Programming*, John Wiley & Sons, Chichester, England, 1994.

W.K. Klein Haneveld, L. Stougie and M.H. van der Vlerk, "An Algorithm for the Construction of Convex Hulls in Simple Integer Recourse Programming," *Annals of Operations Research*, 64: 67-81, 1996.

W.K. Klein Haneveld and M.H. van der Vlerk, "Stochastic Integer Programming: General Models and Algorithms," *Annals of Operations Research*, 85: 39-57, 1999.

K.L. Hoffman and M. Padberg, "LP-Based Combinatorial Problem Solving," *Annals of Operations Research*, 4:145-194, 1985.

M. Kojima and A. Takeda, "Complexity Analysis of Successive Convex Relaxation Methods for Nonconvex Sets," Research Report B-350, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Meguro, Tokyo 152-8552, Japan, 1999.

M. Kojima and L. Tuncel, " Discretization and Localization in Successive Convex Relaxation Methods for Nonconvex Quadratic Optimization," Research Report B-341, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Meguro, Tokyo 152-8552, Japan, 1999.

T.A. Lacksonen, "Static and Dynamic Layout Problems with Varying Areas," *Journal of the Operations Research Society*, 45: 59-69, 1994.

A. Land and A. Doig, "An Automatic Method of Solving Discrete Programming Problems," *Econometrica*, 28(3): 497-520, 1960.

A. Langevin, B. Montreuil and D. Riopel, "Spine Layout Design," *International Journal of Production Research*, 32" 429-442, 1994.

G. Laporte and F.V. Louveaux, "The Integer L-Shaped Method for Stochastic Integer Programs with Complete Recourse," *Operations Research Letters*, 13(3): 133-142, 1993.

G. Laporte, F.V. Louveaux and H. Mercure, "The Vehicle Routing Problem with Stochastic Travel Times," *Transportation Science*, 26(3): 161-170, 1992.

G. Laporte, F.V. Louveaux and L. van Hamme, "Exact Solution of a Stochastic Location Problem by an Integer L-Shaped Algorithm," *Transportation Science*, 28(2): 95-103, 1994.

R.D. Meller and K.Y. Gau, "The Facility Layout Problem: Recent and Emerging Trends and Perspectives," *Journal of Manufacturing Systems*, 15: 351-366, 1996.

R.D. Meller, V. Narayanan and P.H. Vance, "Optimal Facility Layout Design," *Operations Research Letters*, 23: 117-127. 1999.

B. Montreuil, "A Modeling Framework for Integrating Layout Design and Flow Network Design," *Proceedings of the Materials Handling Research Colloquium* (Hebron, KY), 43-58, 1990.

B. Montreuil, U. Venkatadri and H.D. Ratliff, "Generating a Layout from a Design Skeleton," *IIE Transactions*, 25: 3-15, 1993.

F.H. Murphy, S. Sen and A.L. Soyster, "Electric Utility Capacity Expansion Planning with Uncertain Load Forecasts," *AIIE Transactions*, 14: 52-59, 1982.

G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization*, John Wiley & Sons, New York, NY. *Mathematical Programming*, 6: 48-61, 1988.

I. Nowak, "Some Heuristics and Test Problems for Nonconvex Quadratic Programming Over a Simplex," Preprint 98-17, Humboldt University Berlin, 1998a. (Also available online at http://www-iam.mathematik.hu-berlin.de/~ivo/ivopages/work.html.)

I. Nowak, "A Global Optimality Criterion for Nonconvex Quadratic Programming Over a Simplex," Preprint 98-18, Humboldt University Berlin, 1998b. (Also available online at http://www-iam.mathematik.hu-berlin.de/~ivo/ivopages/work.html.)

I. Nowak, "A New Semidefinite Programming Bound for Indefinite Quadratic Forms Over a Simplex," *Journal of Global Optimization*, 14: 357-364, 1999.

M. Padberg and G. Rinaldi, "Optimization of a 532-City Travelling Salesman Problem by Branch-and-Cut," *OR Letters*, 6:1-8, 1987.

C.C. Paige and M.A. Saunders, "Solution of Sparse Indefinite Systems of Linear Equations," *SIAM Journal on Numerical Analysis*, 12(4): 617-629, 1975.

R.G. Parker and R.L. Rardin, *Discrete Optimization*, Academic Press, Inc., Boston, MA, 1988.

M. Ramana and A.J. Goldman. "Some Geometric Results in Semidefinite Programming," *Journal of Global Optimization*, 7: 33-50, 1995.

M. Ramana and P. Pardalos, "Semidefinite Programming," *Interior Point Methods of Mathematical Programming*, Terlaky (ed.), 369-398, 1996.

A. Ruszczynski, "Some Advances in Decomposition Methods for Stochastic Linear Programming," *Annals of Operations Research*, 85: 153-172, 1999.

R. Schultz, "On Structure and Stability in Stochastic Programs with Random Technology Matrix and Complete Integer Recourse," *Mathematical Programming*, 70(1): 73-90, 1995.

R. Schultz, L. Stougie and M.H. van der Vlerk, "Two-Stage Stochastic Integer Programming: A Survey," *Statistica Neerlandica*, 50(3): 404-416, 1996.

R. Schultz, L. Stougie and M.H. van der Vlerk, "Solving Stochastic Programs with Integer Recourse by Enumeration: A Framework using Grobner Basis Reductions," *Mathematical Programming*, 83(2): 229-252, 1998.

H.D. Sherali, "Disjunctive Programming," *Encyclopedia of Optimization*, C.A. Floudas and P.M. Pardalos (eds.), Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999.

H.D. Sherali and W.P. Adams, "A Hierarchy of Relaxations Between the Continuous and Convex Hull Representations for Zero-One Programming Problems," *SIAM Journal on Discrete Mathematics,* 3(3): 411-430, 1990.

H.D. Sherali and W.P. Adams, "A Hierarchy of Relaxations and Convex Hull Characterizations for Mixed-Integer Zero-One Programming Problems," *Discrete Applied Mathematics,* 52(1): 83-106, 1994.

H.D. Sherali and W.P. Adams, "Computational Advances Using the Reformulation-Linearization Technique (RLT) to Solve Discrete and Continuous Nonconvex Problems," *Optima*, 49: 1-6, 1996.

H.D. Sherali and W.P. Adams, *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*, Kluwer Academic Publishing, Boston, MA, 1999.

H.D. Sherali, W.P. Adams and P.J. Driscoll, "Exploiting Special Structures in Constructing a Hierarchy of Relaxations for 0-1 Mixed Integer Problems," *Operations Research*, 46(3): 396-405, 1998

H.D. Sherali and P.J. Driscoll, "Evolution and State-of-the-Art in Integer Programming," *Journal of Computational and Applied Mathematics*, special issue on *"The State of the Art in Numerical Analysis,"* ed. Layne T. Watson, 124: 319-340, 2000.

H.D. Sherali, Y. Lee and Y. Kim, "Partial Convexification Cuts," Manuscript, Grado Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061. (Also presented at the *17th International Symposium on Mathematical Programming*, Atlanta, GA, August 7-11, 2000.)

H.D. Sherali and C.M. Shetty, "On the Generation of Deep Disjunctive Cutting Planes," *Naval Research Logistics Quarterly*, 27(3): 453-475, 1980.

H.D. Sherali and C.M. Shetty, *Optimization with Disjunctive Constraints*, from the series *Lecture Notes in Economics and Mathematical Systems*, Volume 181, Springer-Verlag, Berlin, 1980.

H.D. Sherali and J.C. Smith, "Improving Discrete Model Representations Via Symmetry Considerations," Working Paper, Grado Department of Industrial and Systems Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, 1999.

H.D. Sherali and C.H. Tuncbilek, "A Global Optimization Algorithm for Polynomial Programming Problems Using a Reformulation-Linearization Technique," *Journal of Global Optimization,* 2: 101-112, 1992.

H.D. Sherali and C.H. Tuncbilek, "A Reformulation-Convexification Approach for Solving Nonconvex Quadratic Programming Problems," *Journal of Global Optimization*, 7:1-31, 1995.

H.D. Sherali and C.H. Tuncbilek, "New Reformulation-Linearization/Convexification Relaxations for Univariate and Multivariate Polynomial Programming Problems," *Operations Research Letters,* 21(1): 1-10, 1997.

H.D. Sherali and H. Wang, "Global Optimization of Nonconvex Factorable Programming Problems," *Mathematical Programming*, 89(3): 459-478, 2001.

N.Z. Shor, *Nondifferentiable Optimization and Polynomial Problems*, Kluwer Academic Publishing, Boston, MA, 1998.

A. Takeda, Y. Dai, M. Fukuda and M. Kojima, "Towards the Implementation of Successive Convex Relaxation Method for Nonconvex Quadratic Optimization Problems," Research Report B-347, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Meguro, Tokyo 152-8552, Japan, 1999.

M.J. Todd, *Semidefinite Programming: Applications, Duality, and Interior-Point Methods*, presented at the Fall INFORMS meeting, Seattle, WA, 1998. Also available online at http://www.orie.cornell.edu/~miketodd/todd.html.

R.M. Van Slyke and R. Wets, "L-Shaped Linear Programs with Applications to Optimal Control and Stochastic Programming," *SIAM Journal of Applied Mathematics*, 17(4), 638-663, 1969.

L. Vandenberghe and S. Boyd, "Semidefinite Programming," *SIAM Review*, 38(1): 49-95, 1996.

R.J. Vanderbei and H.Y. Benson, "On Formulating Semidefinite Programming Problems as Smooth Convex Nonlinear Optimization Problems," Working Paper, Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ, 2000.

H. Wolkowicz, R. Saigal and L. Vandenberghe, *Handbook of Semidefinite Programming: Theory, Algorithms, and Applications*, Kluwer Academic Publishers, Boston, MA, 2000.

Vita

In May 2001, Barbara M. P. Fraticelli completed her Ph.D. studies in Industrial and Systems Engineering at Virginia Tech with a 3.95 GPA. Studying under the direction of Dr. Hanif D. Sherali, Barbara's research focused on extensions of the Reformulation-Linearization Technique (RLT) of Sherali and Adams for nonconvex optimization problems. Barbara was named as a finalist for the Paul E. Torgersen Research Excellence Awards (given by the Virginia Tech College of Engineering) for this research effort, which has resulted in three archival publications. At Virginia Tech, Barbara was funded through the Charles E. Minor and Pratt fellowships, and as a research assistant on a *National Science Foundation (NSF)* project. In addition, for her first year at Virginia Tech and for her Master's work at Penn State, she was funded by an NSF Graduate Research Fellowship. Barbara worked under the direction of Dr. Tom Cavalier and Dr. El-Amine Lehtihet for her undergraduate honors' and Masters' theses, both of which used optimization techniques to improve discrete parts manufacturing. This research resulted in two papers published in the *International Journal of Production Research*. As an undergraduate, Barbara was selected as University Marshall for the Penn State College of Engineering, recognizing her as the highest-ranking student in the College with a perfect 4.0 GPA.

Barbara has remained active at Virginia Tech and in the community. She held a two-year term as Chief Justice of the Graduate Honor System and on the Commission on Graduate Studies and Policies. For three years, she also served on the College of Engineering Graduate Student Committee, which organizes an annual seminar for professional development and administers awards for excellence in graduate research. At Penn State, she served as President of the Newman Club and treasurer of the INFORMS chapter. Since May 1996, Barbara has been married to Thomas Darin Fraticelli, who currently serves as the Local Sales Manager at WDBJ-7 TV in Roanoke. The couple is very active at St. Mary's Catholic Church in Blacksburg.