ROW GENERATION TECHNIQUES FOR APPROXIMATE SOLUTION OF LINEAR PROGRAMMING PROBLEMS

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

A. Burak Paç

September, 2010

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Emre Alper Yıldırım (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Osman Oğuz

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Orhan Arıkan

Approved for the Institute of Engineering and Science:

Prof. Levent Onural
Director of the Institute

ABSTRACT

ROW GENERATION TECHNIQUES FOR APPROXIMATE SOLUTION OF LINEAR PROGRAMMING PROBLEMS

A. Burak Paç
M.S. in Industrial Engineering
Supervisor: Assoc. Prof. Emre Alper Yıldırım
September, 2010

In this study, row generation techniques are applied on general linear programming problems with a very large number of constraints with respect to the problem dimension. A lower bound is obtained for the change in the objective value caused by the generation of a specific row. To achieve row selection that results in a large shift in the feasible region and the objective value at each row generation iteration, the lower bound is used in the comparison of row generation candidates. For a warm-start to the solution procedure, an effective selection of the subset of constraints that constitutes the initial LP is considered. Several strategies are discussed to form such a small subset of constraints so as to obtain an initial solution close to the feasible region of the original LP. Approximation schemes are designed and compared to make possible the termination of row generation at a solution in the proximity of an optimal solution of the input LP. The row generation algorithm presented in this study, which is enhanced with a warm-start strategy and an approximation scheme is implemented and tested for computation time and the number of rows generated. Two efficient primal simplex method variants are used for benchmarking computation times, and the row generation algorithm appears to perform better than at least one of them especially when number of constraints is large.

Keywords: Row generation, simplex method, clustering.

ÖZET

DOĞRUSAL PROGRAMLAMA PROBLEMLERİNİN YAKLAŞIK ÇÖZÜMÜ İÇİN KISIT TÜRETME TEKNİKLERİ

A. Burak Paç
Endüstri Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Doç. Dr. Emre Alper Yıldırım
Eylül, 2010

Bu tez çalışmasında, kısıt sayısı problem boyutuna göre çok fazla olan doğrusal programlama problemleri üzerinde kısıt türetme teknikleri uygulandı. Eklenen bir kısıtın amaç fonksiyon değerinde ortaya çıkardığı değişim için bir alt sınır değeri hesaplanabileceği ortaya kondu. Her kısıt türetme adımında, problemin olurlu bölgesinde büyük bir daralma ve amaç fonksiyon değerinde büyük bir değişim sağlayabilmek için bu alt sınır hesabı kısıt türetme adaylarının karşılaştırılması için kullanıldı. Problem çözümüne hızlı bir başlangıç yapabilmek için, ilk doğrusal programlama alt probleminin kısıtlarının etkin bir biçimde seçimini sağlayacak yöntemler araştırıldı. İlk alt problem çözümünün asıl doğrusal programlama probleminin olurlu bölgesine yakın olmasını sağlayacak, mümkün olduğunca küçük bir başlangıç kısıt alt kümesinin elde edilmesinde kullanılabilecek yöntemler değerlendirildi. Asıl problemin en iyi çözümüne yeterince yakın bir çözüm noktasında kısıt türetimine son verebilmek için yaklaşım tasarımları ele alındı ve karşılaştırıldı. Bu çalışmada sunulan kısıt türetme algoritması bir hızlı başlangıç tekniği ve yaklaşım tasarısıyla geliştirilerek bilgisayar üzerinde uygulandı. Bu uygulama algoritmanın hesaplama süresinin ve türettiği kısıt sayısının sınanmasında kullanıldı. Hesaplama zamanları iki verimli temel simpleks yöntemi ile karşılaştırıldı. Karşılaştırmalara göre, kısıt türetme algoritmasının bu iki yöntemden en az birinden, özellikle kısıt sayısı büyük olduğunda, daha hızlı çalıştığı ortaya çıktı.

Anahtar sözcükler: kısıt türetme, simpleks yöntemi, kümeleme .

Acknowledgement

Greatest thanks to the friend whose support is ever with me, my family, my parents the foremost.

Kind, tolerant and supportive attitude of the members of the academy was very important for me during my studies. For every new topic learned, I owe greatly to this attitude. I especially thank my advisor Proffesor Emre Alper Yıldırım for his great tolerance and patience with me during our research.

I would like to thank Proffesors Orhan Arıkan and Osman Oğuz for answering my urgent call to the thesis presentation, for their time and constructive feedback. Indeed, their contribution to my academic experience is beyond this.

I am grateful to Dear Esra Koca, if it were not for her, research would not reach beyond scribbles.

Finally, I want to thank every person who has a role in the warm and nice culture of Bilkent University; our founder Hocabey first of all, the academia, colleagues, and the administrative staff.

Contents

List of Tables

List of Algorithms

Chapter 1

Introduction

A linear program is the minimization or maximization of a linear objective function of a finite number of continuous variables over their domain which is constrained by finitely many linear equalities and inequalities. A linear programming (LP) model is applicable, for instance, when profit maximization is aimed in a production system with limited resources to be allocated to the production of different types of products. In the model of such a system, the variables are the production quantities of, thus the proportional allocation amounts of resources to, each type of product. The objective function is the sum of contributions from the production of each type to the profit. Resources used in the production system can be utilized up to their certain specific level, which poses restrictions on the allocations. These restrictions constitute the constraints, formulated as an inequality: the sum of allocations of a resource type to each product type should be less than or equal to the maximum available amount of that resource type. Again, the linear programming model is applicable to a manufacturing environment in an attempt to minimize manufacturing costs, where the output of several products is required in certain fixed levels. Different products demand different manufacturing processes and activities that are completed by the manufacturing machines and facilities capable of performing these tasks. The cost of processes and activities on the manufacturing units sum up to form the objective function.

The LP solution aims the most cost effective setting of variables, i.e., the allocation of production capabilities of manufacturing units; to meet the constraints: the activity and process allocation for each product type should be sufficient to meet the output requirement for that product.

During late 1940's, linear programming was initially devised as a mathematical model motivated by the urge for planning of complex military training, logistics and operations programs. Soon after the development of mathematical foundations and systematization of solution by the simplex method, LP became a widely applied modeling technique beyond its use as a military planning tool. During the same years LP emerged to become focus of academic interest, digital electronic computers were designed and made available for problem solving [19]. With great impact on scientific decision making, LP was broadly utilized for assisting decisions on commercial and industrial systems posing increasingly more complex problems in the course of the period of economic and technological progress after World War II. Since then, LP has been the most frequently used, or seldom the second most frequently used, operations research tool. Moreover, LP surpassed most computer science and applied mathematics problems in terms broadness of fields of real world practice. Economics, finance, government planning, military operations, manufacturing, agriculture, transportation, medical imaging, statistics, engineering disciplines, physical and social sciences can be named among many fields of application of LP.

Although nature is highly nonlinear, many modern systems have linear structure in design. If nonlinearity is encountered in a human designed system, it is likely that there is a systematic way of linearizing it, namely representing the nonlinearity in a linear form. Moreover, simplification is an important aspect of successful mathematical modeling applications, and it is often justifiable to simplify models to linear programs under reasonable assumptions on the system modeled. Model clarity and readability, together with efficient solution methods, further add to the preferability of LP modeling.

When simplex method appeared as the first systematic method for solving linear programs in 1947, it was acknowledged as a successful algorithm in terms of

solution efficiency. Simplex remained as the main method of LP solver software implementations until mid-1990's. Serious attention and effort from academia and researchers was concentrated on improving simplex algorithm and its implementations. One of the initial theoretical issues was degeneracy and stalling of the solution process due to cycling. It was possible for the simplex method to repeat a sequence of non-improving degenerate simplex pivots in an infinite loop, which is known as cycling. Handling of degeneracy and development of anti-cycling pivoting techniques for the simplex method was one of the most important topics of early research [16], [48], [51], [64],[77], [18], [26]. An anti-cycling pivoting technique is also called a finite pivoting rule, emphasizing a key property of simplex method: unless cycling occurs, the simplex method finds an optimal solution or terminates by certification of infeasibility or unboundedness in a finite number of iterations. This is of theoretical significance, since this property of simplex method can be used for any problem that can be modeled as an LP problem, to prove that it can be solved or discovered to be infeasible or unbounded in finitely many steps. Numerical stability appeared as another important issue, related to software implementations. Straightforward computer implementations of theoretically efficient algorithms do not always fulfill the performance expectations arising from the theoretical results. Accumulation of round-off error through iterations due to the limited numerical precision of digital electronic computers often results in failure of straightforward simplex implementations to return reasonable solutions. Due to the instability caused by the storage and updating of the inverse of the basis in its original matrix form, alternative forms of storage and update of the basis inverse were considered. Of various suggestions on triangular and orthogonal factorizations of the basis matrix [22], [28], [29], [30], [36], [69], LU factorization was the most preferable for its accuracy, stability and efficiency. LU factorizations of the basis matrix in (Gauss-Jordan) product form of inverse (PFI) and (Gaussian) elimination form of the inverse (EFI) were incorporated into the simplex algorithm by studies of Forrest and Tomlin [23], and Bartels and Golub [9], [8], [7], respectively. The two methods differed in the pivot selection criteria and storage of the basis update matrices. Bartels-Golub factorization proved to be more stable numerically, in addition to the superiority of EFI in maintaining the sparsity of the basis factorization [17], [46]. Both of the factorizations were

used successfully in implementations, and further advanced to take advantage of sparse LP problems [75], [76], [70].

The pivot column choice of the early simplex method by George B. Dantzig was that with the highest reduced cost among columns with positive reduced costs, for a minimizing LP model. The reduced cost is the reduction caused in the objective value by one unit of increase in the pivot column variable on entering the basis. Yet the step size, that is, the increase in the pivot column variable is a non-negative number determined by the ratio test; which can be arbitrarily close to zero, equaling zero in the case of degeneracy. Therefore, the highest reduced cost choice does not necessarily bring forth the greatest reduction in the objective value, actually it bears no sign of greater improvement than any of the other candidates. In late 1960's, a pivot selection technique with significant geometric interpretation was brought forward. The column selection of this technique named as the steepest edge pivoting corresponds to the simplex iteration traversing the edge having the most acute angle with the objective direction. This assures the greatest improvement in objective per unit move in the solution space. Although the step size is determined by the ratio test, and again it can be arbitrarily small; the delusion by reduced cost due to the varying norms of edge directions corresponding to non-basic columns, is eliminated in this method. This method is shown to reduce the total number of simplex iterations for solution of LP problems considerably, and rival the computation times of the devex code [31]. Devex is one of the most prominent commercial LP implementations still used, and its computation principle is similar to the steepest edge simplex method [35]. Whereas the steepest edge computes and updates edge norms, devex estimates them by representative data.

Despite the recognized efficiency of the simplex method, there were alternative solution approaches in as early as 1940's. Instead of traversing around the LP feasible region on an edge path on the polyhedron boundary, the possibility of convergence to an optimal solution through points in the interior of the feasible region was investigated even in those years. Yet, serious consideration of the interior point approach as a competitor of the simplex method had to wait for several decades. In 1979, the ellipsoid method came forth as an interior algorithm.

It was not an efficient algorithm for solving LP problems, but was important for proving the classification of LP as a polynomially solvable problem [43]. The discovery of this potential, while the worst-case exponential behavior of simplex was known [32], was followed by the interior point method by Karmarkar, realizing the potential [42]. The method was efficient and even superior to simplex on large-scale problems. The focus of academic interest on interior point methods continued through late 1980s and 1990s, until the competition between simplex and interior points culminated in the development of the infeasible primal-dual interior point method that efficiently solved large-scale LP problems[2], [3], [24], [44], [57], [85], [87], [45].

The simplex method continued to be an important research topic even after the victory of interior point methods and solvability of very large-scale LP problems. The larger steps during initial iterations of interior point methods change into a slow convergence as optimality is approached. Shifting to a simplex method with rather solid iterations when convergence slows down is considered as a fruitful option [15]. Furthermore, in spite of the overall dominance of interior point methods, problem specific structure may often dictate the simplex method to be the key algorithm for solving that type of problem. Recognition and handling of problem specific structure embedded in the LP model is crucial for LP software competitivity, therefore a good implementation of interior point and simplex methods needs be incorporated into LP codes. Solvability of large-scale LPs in reasonable time is not the ultimate target for solution efficiency. LP solvers form a basis for many integer, mixed integer and other solvers. The solution of an integer programming problem might demand thousands of LP solutions, where utilizing the simplex method to inherit and use the basis from iteration to iteration is crucially advantageous.

Dual simplex brought a new view point to the simplex method, with a more simple geometry: feasible region determined by constraint half spaces and basic solutions uniquely defined by a system of independent linear equations formed by the normal vectors of basic constraints. It is simpler than the primal simplex geometry in standard form: the affine space due to an underdetermined linear system constrained by the non-negative orthant. Intersections with hyperplanes

of the non-negativity constraints define basic feasible solutions and shift to a neighbor in the equation system is by replacing one non-negativity hyperplane by another. Shift to a neighboring solution in dual simplex by replacing one basic constraint by a non-basic constraint. Dual simplex is a field developing with rather late efforts. The application of the steepest edge pivoting rules to dual simplex resulted in implementations that were the initial versions of the dual simplex method, which were considered efficient. Techniques improving primal simplex to its current state were also applicable to dual simplex, generally in simpler form parallel to the dual geometry. In the recent years, the dual simplex method has been implemented in several commercially competitive general LP solver softwares. Today's efficient LP solver softwares are state-of-the-art implementations of the primal simplex, dual simplex and interior point methods, or a combination of these.

Exterior point simplex algorithms (EPSA) emerged as simplex variants with the idea of traversing basic solutions instead of basic feasible solutions. Whereas the primal simplex algorithm iterates over primal feasible solutions, and dual simplex over dual feasible solutions, EPSA does not require feasibility on a basis change. While the shortest feasible edge-path connecting two basic feasible solutions on a polyhedron may contain arbitrarily high number of edges for any fixed dimension greater than one, the number is linearly bounded in row and column number when connecting two basic solutions by a path passing through basic solutions[66]. Allowing infeasible bases brings the challenge of designing a penalty function for infeasibility, similar to those used in phase one simplex algorithms. This penalty function combined into the objective function is critical in determining the performance of EPSA, and some efficient implementations exist [66], [67], [65].

Column and row generation algorithms have a potential for better handling of large-scale LPs [39]. The term "'generation"' conveys the meaning that rows and columns are incorporated into the LP as needed. Column generation starts with a minimal feasible system, if it exists, admitting an initial subset of variables that are promising candidates for the optimal basis. After the initial system is solved to optimality, most promising of the variables are set aside, and among those

one which indicates an improving direction is incorporated into the system. The solution process ends by optimality if there is no such variable, or continues with the solution of the revised LP. Row generation algorithms start with a subset of constraints forming a bounded LP, if possible. One of the excluded constraints is incorporated into the optimized system until feasibility is attained. At this point optimality for the relaxed system indicates optimality for the original system. Row generation aims constructing the part of the polyhedral boundary on which an optimal vertex lies. To achieve this, of the violated constraints, the one more likely to bind an optimal vertex is selected. Both column and row generation procedures are finite, since generation is bounded by problem dimensions and each step of generation is finite due to finiteness of LP solution. Starting and finishing with as few number of columns or rows as possible is crucial for faster LP solutions, thus the efficiency of both algorithms. This sets forth the main challenge for these algorithms: detecting and generating columns or rows constructing an optimal basis in fewest generation steps. Column and row generation algorithms favor simplex as the LP solver, due to the small expected number of changes on the basis inherited from the previous generation step.

Chapter 2

Problem Definition

After the introduction of the thesis subject together with a brief review of the related literature, this chapter is devoted to the formal definition of the dual row generation problem.

We consider general linear programming problems, combining the row generation method with warm-start initial constraint set selection techniques. The solution procedure starts with the selection of an initial subset of the LP constraints that constitutes a polyhedron containing at least one basic feasible solution. After the selection of the initial constraints, our algorithm proceeds with the one-by-one inclusion of the remaining constraints in the LP, until either LP infeasibility is encountered or a feasible solution with the desired approximation level to optimal value is computed. In the worst case, the procedure could end up with the inclusion of all constraints, in the case of unboundedness, for instance.

Several strategies are suggested and compared for both warm-start and constraint selection at each step of row generation, with the aim of faster convergence to the optimal value by generation of a minimal number of constraints in total.

A general LP can be formulated as:

$$(P) \qquad \begin{array}{lrcl} \text{maximize} & c^T x & & \\ \text{subject to} & Ax & = & b \\ & x & \geq & 0 \end{array}$$

where $A \in \mathbb{R}^{m \times n}$ is the constraint coefficient matrix, $x \in \mathbb{R}^{n \times 1}$ is the column vector of n variables and $b \in \mathbb{R}_{+}^{m \times 1}$ is the right-hand-side coefficient vector, and $c \in \mathbb{R}^{n \times 1}$ is the objective function coefficient vector. Non-negativity of b can be assured for a general LP, since A may have both negative and positive entries, and a row of A can be negated together with the corresponding entry of b if this entry is negative. In addition, we assume that $m < n$ as the rows of A are linearly independent and the equation system is underdetermined.

The dual LP of (P) is as follows:

$$(D) \qquad \begin{array}{lrcl} \text{minimize} & b^T w & & \\ \text{subject to} & A^T w & \geq & c \end{array}$$

In this model, $w \in \mathbb{R}^{m \times 1}$ is the column vector of m variables. The existence of a basic solution of (D) is implied by the fact that A^T has full column rank.

Again, (D) is a general LP form. Any LP problem can be formulated in this form, since *equality* constraints can be replaced by two inequality constraints and any *less than or equal* type inequality constraint, including non-positivity constraints, can be multiplied by -1 and written as a *greater than or equal* type constraint. Let the original formulation of an LP problem be given by:

$$\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{n} c_j x_j & & & & \\
\text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j & \geq & b_i & i \in I_1 & (2.1)\\
& \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i & i \in I_2 & (2.2)\\
& \sum_{i=1}^{n} a_{ij} x_j & = & b_i & i \in I_3 & (2.3)\\
& x_j & \geq & 0 & j \in J_1 & (2.4)\\
& x_j & \leq & 0 & j \in J_2 & (2.5)\\
& x_j \quad \text{free} & & & j \in J_3 & (2.6)
\end{aligned}$$

In this formulation, I_1, I_2 and I_3 are disjoint sets whose union $\{1, \ldots, m\}$, similarly J_1, J_2 and J_3 are a partition of $\{1, \ldots, n\}$. Introducing a new variable x'_j such that $x'_j = -x_j$ for $j \in J_2$, and regarding non-negativity constraints as regular LP constraints, an equivalent formulation in the form of (D) for the problem is obtained as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{j \in J_1 \cup J_3} c_j x_j + \sum_{j \in J_2} (-c_j) x'_j & & & & \\
\text{subject to} \quad & \sum_{j \in J_1 \cup J_3} a_{ij} x_j + \sum_{j \in J_2} (-a_{ij}) x'_j & \geq & b_i & i \in I_1 \cup I_3 & (2.7)\\
& \sum_{j \in J_1 \cup J_3} (-a_{ij})\, x_j + \sum_{j \in J_2} a_{ij} x'_j & \geq & -b_i & i \in I_2 \cup I_3 & (2.8)\\
& x_j & \geq & 0 & j \in J_1 & (2.9)\\
& x'_j & \geq & 0 & j \in J_2 & (2.10)
\end{aligned}$$

Therefore, the LP originally in the former generic format can be modified into the format we will be using to represent general linear programs. With the

relabeling of coefficients and variables, the above LP can be written as:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} \hat{c}_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} \hat{a}_{ij} x_j \geq \hat{b}_i \qquad i \in \{1, .., m\} \end{aligned}$$

or in matrix form:

$$\begin{aligned} \text{minimize} \quad & \hat{c}^T x \\ \text{subject to} \quad & \hat{A}x \geq \hat{b} \end{aligned}$$

which is exactly the format of (D). Therefore, regardless of its original format, every LP problem instance can be formulated in the format of (D), which is the format used for modeling, analysis and solution of LPs in this study.

From this point on (D) is stated as follows:

$$(D) \qquad \begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax \geq b \end{aligned}$$

In this formulation there is a change of notation. $A \in \mathbb{R}^{m \times n}$ is a full column rank matrix with $m > n$, $b \in \mathbb{R}^{m \times 1}$, and $c \in \mathbb{R}_{+}^{n \times 1}$. It can be assured that c is non-negative for a general LP, since A may have both negative and positive entries, and a column of A can be negated together with the corresponding entry of c if this entry is negative. Therefore, any LP can be represented by this format.

Before defining the problem in detail, the notation used in the study can be briefly summarized as follows:

- The index sets $\{1, \ldots, m\}$ and $\{1, \ldots, n\}$ are named as I and J, respectively.
- If S represents a permuted set, $S(i)$ is the element of S whose order is i in S in the permuted form, $i \in \{1, \ldots, |S|\}$.
- For a permuted index set $S \subset I$, $A_{S\bullet}$ is an $|S| \times n$ matrix whose row i is the row $S(i)$ of A. b_S is a vector of $|S|$ elements with i^{th} element equal to $b_{S(i)}$, the $S(i)^{th}$ entry of $b \in \mathbb{R}^m$ (independent of whether b is a column or row vector), for $i \in \{1, \ldots, |S|\}$.

- For a permuted index set $S \subset J$, $A_{\bullet S}$ is an $m \times |S|$ matrix whose column j is the column $S(j)$ of A, $j \in \{1, \ldots, |S|\}$.
- For $i \in I$, $A_{i\bullet} = A_{\{i\}\bullet}$ and $b_i = b_{\{i\}}$.
- For $j \in J$, $A_{\bullet j} = A_{\bullet\{j\}}$.
- $\vec{1}$ is a column vector with all entries equal to 1, e_i is a column vector with entries equal to 0 except entry i, which is equal to 1. Both of these represent vectors with various sizes appropriate to the context they are used in.
- In the context of matrix operations, I represents the identity matrix of appropriate size.
- $\|x\|$ denotes the Euclidean norm of vector x defined on the space of x.

For the discussion in this section, (D) is assumed to be a feasible and bounded problem. Furthermore, the existence of an initial partition of I into three sets B, N, and V is assumed. This partition is such that $|B| = n$, $rank\,(A_{B\bullet}) = n$, and

$$(D_0) \qquad \begin{array}{lll} \text{minimize} & c^T x & \\ \text{subject to} & A_{B\bullet}\, x & \geq \quad b_B \\ & A_{N\bullet}\, x & \geq \quad b_N \end{array}$$

is a bounded LP with an optimal solution $\hat{x}_0 = (A_{B\bullet})^{-1} b_B$. The initial construction of such a partition is discussed in Section 4.7. In this section, this initial partition is assumed to be known.

If $\hat{x}_0$ is feasible to (D), i.e.,

$$\hat{x}_0 \in \mathcal{F} = \{x \in \mathbb{R}^n : Ax \geq b\}, \tag{2.11}$$

it is optimal to (D). This follows from the fact that (D_0) is a relaxation of (D):

$$\mathcal{F} \subset \mathcal{F}_0 = \{x \in \mathbb{R}^n : A_{B\bullet}\, x \geq b_B, A_{N\bullet}\, x \geq b_N\} \tag{2.12}$$

and

$$c^T \hat{x}_0 = \inf_{x \in \mathcal{F}_0} c^T x \leq \inf_{x \in \mathcal{F}} c^T x. \tag{2.13}$$

In this case, an optimal solution of (D) is found by the solution of an LP with $|B| + |N|$ rows.

If $\hat{x}$ is not feasible to (D), then for nonempty $\bar{V} \subset V$, we have $A_{\bar{V}\bullet}\,\hat{x} < b_{\bar{V}}$. In this case, a selection of an index $i_1 \in \bar{V}$ is made for incorporation of a new constraint into (D_0). The appending of a violated constraint into LP is called row generation. Then, the LP after the addition of row i_1 is given by:

$$
(D_1) \qquad \begin{array}{lrcl} \text{minimize} & c^T x & & \\ \text{subject to} & A_{B\bullet}\, x & \geq & b_B \\ & A_{N\bullet}\, x & \geq & b_N \\ & A_{i_1\bullet}\, x & \geq & b_{i_1} \end{array}
$$

The property desired in the selection of the row to be added is to shift the LP optimal towards a feasible solution of (D), with a minimum number of row generations in total to reach feasibility. Reaching this point with a small number of row generations means finding an optimal solution to (D) by solving a sequence of smaller sized LP problems.

Ideally, a row generation technique has the aim of selecting rows which appear in an optimal basis. However, finding such constraints that form the boundary of the polyhedron where an optimal vertex lies is not a trivial task. Thus, for quick convergence to feasibility by a small number of row generations, a rather myopic goal for row generation is established. Generally, a relaxation of (D) has a better, that is to say smaller, optimal objective value than (D). At each row generation step, the greatest reduction in this gap between the optimal value of the relaxed problem and the optimal value of (D) is aimed. Another such goal for row generation is to select a constraint such that the polyhedron in the following step is as distant as possible to the optimal solution of the LP before row generation. Namely, cutting out a region as large as possible is aimed. These two goals are interrelated, in that a strategy performing well in terms of one of the goals generally performs well for the other.

2.1 Constructing a Lower Bound for Objective Value Change

Row generation has a computational advantage over feasible simplex methods. When feasible simplex methods pick an edge to move from one basic feasible solution to another, satisfaction of all nonbasic constraints is required. Especially when the number of constraints is large, computing the step size of the move on the edge selected assures feasibility. This computation, called the ratio test, demands significant computational effort when the number of constraints is large. The costly effort required to consider all nonbasic constraints is prohibitive for the comparison of multiple edge directions in terms of their true effect on the objective value. Although an edge direction with more potential for objective value improvement can be selected, the true change in objective is determined by the ratio test, and might turn out to be very small.

On the other hand, row generation candidates are compared to each other in terms of their relation with the basic constraints. Since the number of basic constraints is limited by the problem dimension, a criterion for comparison that has less computational cost can be designed. Therefore, consideration of multiple candidates for row generation is possible. Moreover, the (absolute) change in objective value brought in by a row generation has a certain lower bound. Let (D_k) be the LP problem obtained after k row generations, with i_k as the last row generated:

$$
(D_k) \qquad \begin{array}{lrcl} \text{minimize} & c^T x & & \\ \text{subject to} & A_{B_{k-1}\bullet}\, x & \geq & b_{B_{k-1}} \\ & A_{N_{k-1}\bullet}\, x & \geq & b_{N_{k-1}} \\ & A_{i_k\bullet}\, x & \geq & b_{i_k} \end{array}
$$

Here, B_{k-1} represents the optimal basic indices of (D_{k-1}). In dual simplex, the basis consists of n linearly independent rows, i.e., n constraints with linearly independent normal vectors. The basis defines n edges each emanating from one of the basic constraint hyperplanes, while lying in the intersection of the

remaining constraint hyperplanes. For $j \in J$, the edge emanating from the hyperplane of constraint $B_{k-1}(j)$, denoted by ρ_j is uniquely defined by the linear equation system:

$$\rho_j = A^{-1}_{B_{k-1}\bullet} e_j, \qquad j \in J \tag{2.14}$$

These edges emanating from $\hat{x}_{k-1} = A^{-1}_{B_{k-1}\bullet} b_{B_{k-1}}$ intersect the hyperplane of a violated constraint if their inner product with the normal vector of this hyperplane is positive. Therefore, the set $J_+ = \{j \in J : A_{i_k\bullet}\, \rho_j > 0\}$ corresponds to the edges that intersect the hyperplane $\{x \in \mathbb{R}^n : A_{i_k\bullet} x = b_{i_k}\}$. J_+ is nonempty under the assumption that (D) is feasible. The new constraint intersects the half line $\{x \in \mathbb{R}^n : x = \hat{x}_{k-1} + \mu\rho_j, \mu \geq 0\}$, $j \in J_+$, at point $p_j = \hat{x}_{k-1} + \mu_j \rho_j$. The step size is

$$\mu_j = \frac{b_{i_k} - A_{i_k\bullet}\, \hat{x}_{k-1}}{A_{i_k\bullet}\, \rho_j}, \tag{2.15}$$

where the numerator is also positive, since row i_k is generated only if it is violated by $\hat{x}_{k-1}$.

It should be noted that μ_j is not the step size determined by the dual ratio test for edge ρ_j, and p_j is generally not the point that the solution would move to when a feasible dual simplex method is used. Any non-basic constraints that edge j intersects before hitting $\{x \in \mathbb{R}^n : A_{i_k\bullet} x = b_{i_k}\}$ are ignored in this analysis. Still, the points p_j, $j \in J_+$ give a lower bound for Δ_k, the change in the objective value by generation of row i_k, i.e. the difference between the objective values of optimal solutions of problems (D_{k-1}) and (D_k):

$$\Delta_k = z_k - z_{k-1} = c^T \hat{x}_k - c^T \hat{x}_{k-1} \tag{2.16}$$

$$\Delta_k \geq \min_{j \in J_+} c^T p_j - c^T \hat{x}_{k-1} = \min_{j \in J_+} \mu_j c^T \rho_j \tag{2.17}$$

This lower bound follows from the fact that an optimal solution of (D_k), $\hat{x}_k$, is the sum of a convex combination of p_j, $j \in J_+$ and a feasible direction of the polyhedron

$$\mathcal{H}_k = \left\{x \in \mathbb{R}^n : A_{B_{k-1}\bullet}\, x \geq b_{B_{k-1}}, A_{i_k\bullet}\, x = b_{i_k}\right\}. \tag{2.18}$$

This result and its proof is presented in the following lemma.

Lemma 1 *Let $\hat{x}_{k-1}$ be an optimal solution of (D_{k-1}) with corresponding basis B_{k-1}. Let J_+ be the set of indices corresponding to the edges of basis B_{k-1} that intersect the hyperplane $\{x \in \mathbb{R}^n : A_{i_k\bullet}x = b_{i_k}\}$. Let p_j, $j \in J_+$ be the points that the edges of basis B_{k-1} intersect $\{x \in \mathbb{R}^n : A_{i_k\bullet}x = b_{i_k}\}$.*

Given that (D_k) is a feasible and bounded problem, there exists an optimal solution $\hat{x}_k$ of (D_k) that can be represented by:

$$\hat{x}_k = \sum_{j \in J_+} \lambda_j p_j + d_{\hat{x}_k}, \tag{2.19}$$

where $\lambda_j \geq 0$, $j \in J_+$, $\sum_{j \in J_+} \lambda_j = 1$, and $d_{\hat{x}_k}$ is a direction of:

$$\mathcal{H}_k = \left\{x \in \mathbb{R}^n : A_{B_{k-1}\bullet}\, x \geq b_{B_{k-1}}, A_{i_k\bullet}\, x = b_{i_k}\right\}. \tag{2.20}$$

Proof. The feasible regions of (D_{k-1}) and (D_k) are given by:

$$\mathcal{F}_{k-1} = \left\{x \in \mathbb{R}^n : A_{B_{k-1}\bullet}\, x \geq b_{B_{k-1}}, A_{N_{k-1}\bullet}\, x \geq b_{N_{k-1}}\right\} \tag{2.21}$$

and

$$\mathcal{F}_k = \left\{x \in \mathbb{R}^n : A_{B_{k-1}\bullet}\, x \geq b_{B_{k-1}}, A_{N_{k-1}\bullet}\, x \geq b_{N_{k-1}}, A_{i_k\bullet}\, x \geq b_{i_k}\right\}, \tag{2.22}$$

respectively.

The hyperplane $\{x \in \mathbb{R}^n : A_{i_k\bullet}x = b_{i_k}\}$ separates $\{\hat{x}_{k-1}\}$ and $\mathcal{F}_k$, as $A_{i_k\bullet}\hat{x}_{k-1} < b_{i_k}$ and $A_{i_k\bullet}y \geq b_{i_k}$ for all $y \in \mathcal{F}_k$.

Since $\mathcal{F}_{k-1}$ and $\mathcal{F}_k$ are convex sets, and $\mathcal{F}_k \subset \mathcal{F}_{k-1}$, for any $y \in \mathcal{F}_k$, the line segment connecting y and $\hat{x}_{k-1}$ is inside $\mathcal{F}_{k-1}$ and leaves the set $\mathcal{F}_k$ at point $\hat{y} \in \mathcal{F}_k$ such that:

$$\hat{y} \in \bar{\mathcal{F}}_k = \left\{x \in \mathbb{R}^n : A_{B_{k-1}\bullet}\, x \geq b_{B_{k-1}}, A_{N_{k-1}\bullet}\, x \geq b_{N_{k-1}}, A_{i_k\bullet}\, x = b_{i_k}\right\}. \tag{2.23}$$

As the linear objective function $c^T x$ is also convex, $\hat{y}$ has an objective value as good as that of y. Therefore, an optimal solution of (D_k) exists in $\bar{\mathcal{F}}_k$.

Note that, the only extreme points of the polyhedron $\mathcal{H}_k$ are p_j, $j \in J_+$, since constraints in B_{k-1} only constitute n edges, and $A_{i_k\bullet}\, x = b_{i_k}$, a single hyperplane,

intersects only those edges with indices in J_+. Then, by polyhedral representation theorem, for an optimal solution

$$\hat{x}_k \in \bar{\mathcal{F}}_k \subset \mathcal{H}_k \tag{2.24}$$

of (D_k), a representation exists as

$$\hat{x}_k = \sum_{j \in J_+} \lambda_j p_j + d_{\hat{x}_k}, \tag{2.25}$$

where $\lambda_j \geq 0$, $\sum_{j \in J_+} \lambda_j = 1$ and $d_{\hat{x}_k}$ is a feasible direction of $\mathcal{H}_k$. □

Let us assume that the bound given for Δ_k is not valid, and:

$$c^T \hat{x}_k - c^T \hat{x}_{k-1} < \min_{j \in J_+} \left(c^T p_j - c^T \hat{x}_{k-1} \right). \tag{2.26}$$

This can be stated more simply as:

$$c^T \hat{x}_k < \min_{j \in J_+} c^T p_j. \tag{2.27}$$

By Lemma 1, there exists a representation $\hat{x}_k = \sum_{j \in J_+} \lambda_j p_j + d_{\hat{x}_k}$, with λ_j and $d_{\hat{x}_k}$ as defined in the lemma. Then:

$$\begin{aligned} c^T \hat{x}_k &< \min_{j \in J_+} c^T p_j \\ \sum_{j \in J_+} \lambda_j c^T p_j + c^T d_{\hat{x}_k} &< \sum_{j \in J_+} \lambda_j \min_{\bar{j} \in J_+} c^T p_{\bar{j}} \\ \sum_{j \in J_+} \lambda_j c^T p_j + c^T d_{\hat{x}_k} &< \sum_{j \in J_+} \lambda_j c^T p_j \\ c^T d_{\hat{x}_k} &< 0. \end{aligned} \tag{2.28}$$

Since $d_{\hat{x}_k} \neq 0$ is a direction of $\mathcal{H}_k$, it is also a direction of

$$\mathcal{F}_{B_{k-1}} = \left\{ x \in \mathbb{R}^n : A_{B_{k-1}\bullet}\, x \geq b_{B_{k-1}} \right\}, \tag{2.29}$$

as $\mathcal{F}_{B_{k-1}} \supset \mathcal{H}_k$. There is a contradiction at this point: an optimal basis B_{k-1} satisfies $c^T \left(A_{B_{k-1}\bullet} \right)^{-1} \geq 0$ component wise, i.e. $c^T \rho_j \geq 0$, $j \in J$, so that for any

direction d of $\mathcal{F}_{B_{k-1}}$, $c^T d \geq 0$, since d is a convex combination of ρ_j, $j \in J$. Thus the lower bound on Δ_k is proved:

$$\begin{aligned}
\Delta_k &\geq \min_{j \in J_+} c^T p_j - c^T \hat{x}_{k-1} \\
&= \min_{j \in J_+} \mu_j c^T \rho_j \\
&= \min_{j \in J_+} \frac{b_{i_k} - A_{i_k \bullet} \hat{x}_{k-1}}{A_{i_k \bullet} \rho_j} c^T \rho_j \\
&= \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k \bullet} \rho_j} \left(b_{i_k} - A_{i_k \bullet} \hat{x}_{k-1} \right). \qquad (2.30)
\end{aligned}$$

2.2 Bound on Objective Change from the Primal Perspective

As mentioned above, the step sizes μ_j in these calculations do not correspond to step sizes given by the dual ratio test of a feasible dual simplex algorithm. Yet, there is an analogy between the step sizes μ_j and the primal ratio test. When the row generation problem is approached from the primal perspective, a row generation corresponds to a column generation. Consider LP formulations of (P_{k-1}) and (P_k), the LP problems before and after the kth column generation.

$$(P_{k-1}) \qquad \begin{array}{lrcl} \text{maximize} & b^T w & & \\ \text{subject to} & A^T_{B_{k-1}\bullet} w_{B_{k-1}} + A^T_{N_{k-1}\bullet} w_{N_{k-1}} & = & c \\ & w & \geq & 0 \end{array}$$

$$(P_k) \qquad \begin{array}{lrcl} \text{maximize} & b^T w & & \\ \text{subject to} & A^T_{B_{k-1}\bullet} w_{B_{k-1}} + A^T_{N_{k-1}\bullet} w_{N_{k-1}} + A^T_{i_k \bullet} w_{i_k} & = & c \\ & w & \geq & 0 \\ & w_{i_k} & \geq & 0 \end{array}$$

Since its dual (D_{k-1}) is a feasible and optimal LP with optimal basis B_{k-1}, the same basis set determines the optimal basic columns of (P_{k-1}). The objective change due to the first simplex pivot after the generation of column i_k is equal to the lower bound calculated above. Since other simplex pivots might follow

to maximize the objective value, the objective change caused by generation of column i_k is greater than or equal to the change by this first simplex pivot. This analogy between the primal and dual perspectives follows with the next lemma.

Primal non-degeneracy is assumed in the analyses based on the primal perspective, including the following lemma. (P), thus (P_{k-1}) and (P_k) are assumed to be non-degenerate LP problems.

Lemma 2 *Let (P_{k-1}) and (P_k) be the LP problems before and after the k^{th} column generation, and let B_{k-1} be an optimal basis of (P_{k-1}).*

The change caused in the objective value by the first simplex pivot after the generation of column i_k is equal to

$$\min_{j \in J_+} \mu_j c^T \rho_j, \tag{2.31}$$

which is the lower bound for the change in objective value due to the k^{th} row generation, where

$$\mu_j = \frac{b_{i_k} - A_{i_k \bullet}\, \hat{x}_{k-1}}{A_{i_k \bullet}\, \rho_j}. \tag{2.32}$$

Proof. Let $w^*_{B_{k-1}}$ be the basic part of the optimal solution of (P_{k-1}) corresponding to the optimal basis B_{k-1}.

$$\begin{aligned} A^T_{B_{k-1}\bullet} w^*_{B_{k-1}} &= c \\ w^*_{B_{k-1}} &= A^{-T}_{B_{k-1}\bullet} c \end{aligned} \tag{2.33}$$

The non-basic part of the solution w^* is equal to zero, and remains zero after the row generation when column i_k enters the basis for the first time. Column i_k is known to be the entering variable since all other columns have non-negative reduced cost due to optimality of B_{k-1} for (P_{k-1}). The reduced cost of column i_k is negative:

$$\begin{aligned} b^T_{B_{k-1}} A^{-T}_{B_{k-1}\bullet} A^T_{i_k \bullet} - b_{i_k} &= A_{i_k \bullet} A^{-1}_{B_{k-1}\bullet} b_{B_{k-1}} - b_{i_k} \\ &= A_{i_k \bullet} \hat{x}_{k-1} - b_{i_k} \\ &< 0, \end{aligned} \tag{2.34}$$

since row i_k corresponds to column i_k in the dual, and it is generated only if it violates the optimal solution of (D_{k-1}). When i_k enters the basis, the shift d on $w^*_{B_{k-1}}$ is as follows:

$$\begin{aligned} A^T_{B_{k-1}\bullet}\left(w^*_{B_{k-1}} + d\right) + A^T_{i_k\bullet} w_{i_k} &= c \\ A^T_{B_{k-1}\bullet} d + A^T_{i_k\bullet} w_{i_k} &= 0 \\ d &= -A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet} w_{i_k}. \end{aligned} \tag{2.35}$$

d is the direction of change for the part of the solution corresponding to indices B_{k-1}. The direction $-A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet}$, along with one unit increase in w_{i_k}, where remaining variables are fixed at zero, give the edge direction of the first primal simplex iteration after the column generation. The corresponding ratio test gives the step size μ_{i_k}, the change in w_{i_k} due to this simplex iteration. Note that this step size is positive due to the assumption of non-degeneracy.

$$\begin{aligned} \mu_{i_k} &= \min_{\left(j: e_j^T A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet} > 0\right)} \frac{e_j^T w^*_{B_{k-1}}}{e_j^T A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet}} \\ &= \min_{\left(j: \rho_j^T A^T_{i_k\bullet} > 0\right)} \frac{e_j^T A^{-T}_{B_{k-1}\bullet} c}{e_j^T A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet}} \\ &= \min_{j \in J_+} \frac{c^T A^{-1}_{B_{k-1}\bullet} e_j}{A_{i_k\bullet} A^{-1}_{B_{k-1}\bullet} e_j} \\ &= \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k\bullet} \rho_j}. \end{aligned} \tag{2.36}$$

After the step size μ_{i_k} is determined by the ratio test, the change d in the former basic variables can be written as:

$$\begin{aligned} d &= -A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet} \mu_{i_k} \\ &= -A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet} \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k\bullet} \rho_j}. \end{aligned} \tag{2.37}$$

Then, the change in the objective value due to the kth row generation, Δ_k, is

greater than or equal to the objective change by this primal simplex iteration:

$$\begin{aligned}
\Delta_k &\geq b^T_{B_{k-1}} d + b_{i_k} w_{i_k} \\
&= -b^T_{B_{k-1}} A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet} \mu_{i_k} + b_{i_k} \mu_{i_k} \\
&= -b^T_{B_{k-1}} A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet} \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k\bullet} \rho_j} + b_{i_k} \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k\bullet} \rho_j} \\
&= \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k\bullet} \rho_j} \left(b_{i_k} - b^T_{B_{k-1}} A^{-T}_{B_{k-1}\bullet} A^T_{i_k\bullet} \right) \\
&= \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k\bullet} \rho_j} \left(b_{i_k} - A_{i_k\bullet} A^{-1}_{B_{k-1}\bullet} b_{B_{k-1}} \right) \\
&= \min_{j \in J_+} \frac{c^T \rho_j}{A_{i_k\bullet} \rho_j} \left(b_{i_k} - A_{i_k\bullet} \hat{x}_{k-1} \right). \qquad (2.38)
\end{aligned}$$

Note that the lower bound on the objective change obtained by the primal perspective, given by the inequality in (2.38) is identical to what is obtained by the dual perspective presented in (2.30). □

In the above analyses, (D), thus (P) is assumed to be feasible and optimal. Therefore the set J_+ is always non-empty. Note that, in the absence of this assumption, $J_+ = \emptyset$ is possible. From the dual perspective, this proves that $\mathcal{F}_k = \emptyset$, i.e. (D) is infeasible, since this implies that the edges of $\mathcal{F}_{B_{k-1}}$ do not intersect $\{x \in \mathbb{R}^n : A_{i_k\bullet} x = b_{i_k}\}$. From the primal perspective, this implies unboundedness, since the value of w_{i_k} can take arbitrary positive value without causing negativity of any variables in B_{k-1}.

Although calculations using the whole edge set provide a lower bound for objective change, the computational cost for these calculations cannot be incurred for all of the row generation candidates. Since this study is particularly motivated by LP problems with very large number of constraints compared to the problem dimension, calculations for lower bound have to be limited to a small subset of row generation candidates.

A point to note is that, the lower bound is merely an estimate of the change in the objective function. When constraint i_k is added to (D_{k-1}), constraints in N_{k-1} have to be accounted for along with those in B_{k-1}. Thus, the result of row generation is generally several simplex iterations for transition to the optimal

of (D_k), and a greater change in objective value than what is indicated by the lower bound Δ_k. Therefore, the calculations presented in this chapter have to be combined with a strategy for forming a small subset of strong candidates among the constraints that are not considered in (D_{k-1}). The problem focused on in this thesis study is the design of such a strategy for selection of a small subset of row generation candidates before lower bounds for change are computed and candidates are compared, along with the selection of an initial set of constraints to constitute (D_0).

Stating more formally, the purpose of this study is the design of selection methods for the initial constraint sets B, N and the permuted row generation set $R = \{i_1, \ldots, i_r\}$ such that $|B| + |N| + |R|$ is minimal and the optimal solution of the LP:

$$
(D_r) \qquad \begin{array}{lrcl} \text{minimize} & c^T x & & \\ \text{subject to} & A_{B\bullet}\, x & \geq & b_B \\ & A_{N\bullet}\, x & \geq & b_N \\ & A_{R\bullet}\, x & \geq & b_R \end{array}
$$

is feasible to (D).

The strategies for the construction of the sets B and N of the constraints of the initial LP are discussed in Section 4.7. The selection method for row generation, namely one-by-one determination of elements of R is discussed in the next chapter.

Chapter 3

The Method

In this chapter, the methods used for the selection from among row generation candidates are discussed. For the analyses of this chapter, A is assumed to be an $m \times n$ matrix with normalized rows, and c is assumed to be a unit vector.

For a thorough comprehension of the polyhedral structure of an LP problem, the relationships of constraints with each other due to their orientation and positioning in the space need to be analyzed. The former is determined by the constraint hyperplane normal vector, i.e. $A_{i\bullet}$, $i \in I$, and the latter is determined by the constraint right-hand side b_i. After such an analysis, the boundary where an optimal solution lies, and the constraints forming this boundary, which are determined by the orientation of the objective gradient with respect to the polyhedron, can be obtained by observing the relationship of the objective vector c with the constraints. Unfortunately, this kind of an analysis would be very costly even for small LP problems.

If there existed a clustering of all of the constraints according to their normal vectors, so that each cluster consisted of constraints with similar normal vector directions, this would be valuable information. For two constraints $A_{i\bullet}\, x \geq b_i$ and $A_{j\bullet}\, x \geq b_j$ whose normal vectors are normalized and equal, the one with smaller right hand side is redundant, namely, the other constraint is sufficient for the definition of the LP feasible region and this one can be omitted from the

inequality system. For constraints with normal vectors that are normalized and similar, while the ones with smaller right-hand-sides are not necessarily redundant, the constraint with largest right-hand side is the most important for the definition of the LP feasible region. Therefore, if constraints are clustered according to their normal vectors as mentioned above, picking the constraint with the largest right-hand-side from each cluster provides a small subset of constraints that give an approximate definition of the LP feasible region. Starting with this initial definition of the LP feasible region, it is reasonable to expect that few row generations would be needed until a feasible optimal solution of the original LP is found. However, this approach would require clustering a very large number of vectors, since this study especially focuses on LP problems with large number of constraints. Clustering such a large number of vectors would be a difficult problem in itself, and would require great computational effort; therefore alternative approaches are considered.

A rather simple analysis is possible instead of clustering all constraint normal vectors. The relationship of a constraint normal vector $A_{i\bullet}$ with the objective vector c provides information about whether the constraint is likely to be in an optimal basis, even if this is partial information. After analyzing and classifying the constraints by their relationship with c, constraint right-hand sides serve as a significant indicator of which constraints are more important in defining the polyhedron boundary among constraints of similar orientation in the space.

Constraints with normal vectors in highly acute and highly obtuse angles with c can be considered as two groups of similar orientation in space. Since two unit normal vectors that have highly acute angle with c, namely small distance with c, would have a very small angle between each other, their constraint hyperplanes would be similarly oriented, faces pointing at similar directions. Same is true for two constraints with normal vectors at highly obtuse angle with $-c$. It can thus be said that the constraints with larger right-hand-sides in either of these two groups are more important in defining the LP feasible region than the remaining ones.

Another constraint group of interest is composed of constraints whose normal

vectors are at approximately right angle with c. For the constraints in both of the former two groups, it can be said that their normal vectors are clustered around one ray emanating from the origin. Noting that all constraint normal vectors and the objective vector are normalized, this means that these normal vectors are clustered around vectors c, and $-c$. However, we cannot say that constraints whose normal vectors are approximately orthogonal to c can be clustered around a single vector.

It can be said that normal vectors of constraints that are approximately orthogonal to c lie close to the hyperplane $\left\{x \in \mathbb{R}^n : c^T x = 0\right\}$. This follows, since the cosine of right angle is zero, and the inner product of two normalized vectors give the cosine of the angle between them, therefore $A_{i\bullet}c \approx 0$. From another point of view, the distance of $A_{i\bullet}$ to $\left\{x \in \mathbb{R}^n : c^T x = 0\right\}$ is equal to

$$\left\| c\, c^T A_{i\bullet}^T \right\| = \|c\| \left| c^T A_{i\bullet}^T \right| = \left| c^T A_{i\bullet}^T \right|,$$

which is the norm of a vector obtained by the multiplication of the unit vector c by a very small number, the inner product $c^T A_{i\bullet}^T$. To sum up, unit normal vectors of constraints that are located on or near $\left\{x \in \mathbb{R}^n : c^T x = 0\right\}$ form the group of constraints that are approximately orthogonal to c. Since $\left\{x \in \mathbb{R}^n : c^T x = 0\right\}$ is an $n-1$ dimensional linear space, the analysis of constraints in this group requires further classification of normal vectors into clusters. Clustering this smaller subset of $I = \{1, \ldots, m\}$ has a significantly lower cost than clustering all constraints, and incurring this cost is reasonable since this group is essential for the solution procedure.

The greater portion of constraints are not in the three groups mentioned. It is difficult to analyze and compare these prior to the solution process unless clustering is applied on all of the problem constraints. Application of clustering on all constraints would bring in increased computational cost with problem size. Since this study focuses on LP problems with a very large number of constraints, the option of clustering all constraints is not reasonable due to complexity of the k-center clustering problem. The methods used in this study for the design of initial constraint selection strategies and row generation selection criteria are

therefore based on three classes of constraints mentioned above: those approximately parallel to objective vector c, approximately parallel to objective gradient $-c$ and approximately orthogonal to c.

Let $\alpha_{i,c}$ denote the angle between the normal vector of constraint i and objective vector c, and $\alpha_{i,l}$ denote the angle between the normal vectors of constraints i and l, $i, l \in I$. Let α_Π and α_Ω denote the largest angles that should be between a normal vector and c, for that vector to be a member of approximately parallel and orthogonal groups, respectively. Then the parallel group Π is formed as follows:

$$\Pi = \{i \in I : \alpha_{i,c} \leq \alpha_\Pi\} \tag{3.1}$$

and the orthogonal group Ω is formed as follows:

$$\Omega = \left\{i \in I : \frac{\pi}{2} - \alpha_\Omega \leq \alpha_{i,c} \leq \frac{\pi}{2} + \alpha_\Omega\right\} \tag{3.2}$$

where $\alpha_{i,c} \in [0, \pi]$, $i \in I$.

From the geometric perspective, assuring precise approximation of parallelity and orthogonality is important, but it is also important to adjust the level of α_Π and α_Ω, so that the sets Π and Ω are non-empty, but do not have too many elements either. This is required to assure efficiency in analysis of the polyhedron structure and row generation iterations. Otherwise, not only similarity in orientation of constraints inside groups would be reduced, but also the excessive number of orthogonal group constraints would result in an exhaustive clustering process.

The process of forming parallel and orthogonal groups, and orthogonal clusters is given in Algorithm 1. Note that there are three parameters α_Π, α_Ω and κ, in addition to LP parameters A, b and c. First two discussed above, α_Π and α_Ω, are the angles that determine proximity of parallelity and orthogonality, respectively. The parameter κ takes values in the interval $(0, 1]$ to determine how many clusters the approximately orthogonal vectors are grouped into. The vectors approximately orthogonal to c are grouped into $\omega = \lceil \kappa |\Omega| \rceil$ clusters. Thus, κ is the parameter that determines the clustering problem to be a ω-center clustering problem. In this study, selection of constraints exploiting constraint

groups and clusters, together with other information, is the main focus. α_Π, α_Ω and κ are determined by comparing different settings of these parameters during computational testing. Further analyses on the ideal settings of these parameters for efficiency remain for future research.

Algorithm 1 Forming Sets Π, Ω, and Clustering Ω

Input: $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $\alpha_\Pi \in [0,\pi]$, $\alpha_\Omega \in [0,\pi]$, $\kappa \in [0,1]$
Normalize c and rows of A
$c \leftarrow \dfrac{c}{\|c\|}$
for $i := 1$ to m **do**
  $A_{i\bullet} \leftarrow \dfrac{A_{i\bullet}}{\|A_{i\bullet}\|}$
  $b_i \leftarrow \dfrac{b_i}{\|A_{i\bullet}\|}$
Calculate cosines of angles between constraint normal vectors and c
for $i \in \Omega$ **do**
  $\cos\alpha_{i,c} = A_{i\bullet}c$
Calculate cosines of the angles between constraint normal vectors
for $i \in \Omega$ **do**
  $\cos\alpha_{i,i} = 1$
  for $l \in \Omega$, $l > i$ **do**
    $\cos\alpha_{i,l} = A_{i\bullet}A_{l\bullet}^T$
  for $l \in \Omega$, $l < i$ **do**
    $\cos\alpha_{i,l} = \cos\alpha_{l,i}$
Form sets Π and Ω
$\Pi = \{i \in I : \cos\alpha_{i,c} \geq \cos\alpha_\Pi\}$
$\Omega = \left\{i \in I : \cos\left(\dfrac{\pi}{2} - \alpha_\Omega\right) \geq \cos\alpha_{i,c} \geq \cos\left(\dfrac{\pi}{2} + \alpha_\Omega\right)\right\}$
ω-center clustering of Ω
$\omega = \lceil \kappa\,|\Omega| \rceil$
$o_1 = \arg\min_{o\in\Omega}\{\cos\alpha_{o,c}\}$
$\Omega^1 = \{o_1\}$
for $i := 2$ to ω **do**
  $o_i = \arg\min_{o\in\Omega}\left\{\max_{l\in\{1,\ldots,i-1\}}\{cos\alpha_{o,l}\}\right\}$
  $\Omega^i = \{o_i\}$
for $o \in \Omega \setminus \{o_1,\ldots,o_\omega\}$ **do**
  $i \leftarrow \arg\max_{l\in\{1,\ldots,\omega\}}\{\cos\alpha_{o,l}\}$
  $\Omega^i \leftarrow \Omega^i \cup \{o\}$

For two vectors c and d in $\mathbb{R}^n$, $c^Td = \|c\|\,\|d\|\cos\alpha_{c,d}$, where $\alpha_{c,d}$ is the angle between c and d. When c and d are unit vectors, $c^Td = \cos\alpha_{c,d}$. After the normalization of c and the constraint normal vectors, this identity is used for

the calculation of cosine values. Note that cosine values replace angles for the comparisons in the definition of sets Π and Ω in Algorithm 1. This works, since the cosine function has monotonic behavior in the interval $[0, \pi]$.

The monotonic property of the cosine function in $[0, \pi]$ is exploited in the clustering of Ω. The distance between two vectors $A_{i\bullet}$ and $A_{l\bullet}$ is $2 \sin \frac{\alpha_{i,l}}{2} \|A_{i\bullet}\|$, given $\|A_{i\bullet}\| = \|A_{l\bullet}\|$. Then, $\|A_{i\bullet} - A_{l\bullet}\| = 2 \sin \frac{\alpha_{i,l}}{2}$. This holds since the vectors are normalized to have unit norms. A larger cosine implies smaller distance due to the following trigonometric identity:

$$\sin \frac{\alpha}{2} = \sqrt{\frac{1 - \cos \alpha}{2}} \qquad \alpha \in [0, \pi] \,, \tag{3.3}$$

it can therefore be said that the ω-center clustering is done based on Euclidean distance. The algorithm used is the furthest neighbor approximation algorithm for k-center clustering problem, and it has an approximation factor of 2 [33]. While a factor of 2 might seem large, it is not an indication of inefficiency of the algorithm; and it is proved that 2 is a tight bound for approximation of k-center clustering problem: an approximation algorithm for k-center clustering problem with a factor $2 - \epsilon$, $\epsilon > 0$ implies $P = NP$ [37].

Once sets Π, Ω and Ω^i, $i \in \{1, \ldots, \omega\}$ are formed using Algorithm 1, this information can be used to order constraints so that the constraints that are more likely to define the boundary that an optimal solution lies, and the constraints that are more important for the definition of the LP feasible region come prior to others. The set Π comes foremost in this ordering, since for a constraint normal vector $\tilde{c} \approx c$, the constraint $\tilde{c}^T x \geq b_{\tilde{c}}$ approximates a bound on the objective function: $c^T x \geq b_{\tilde{c}}$. One such constraint might not assure that the objective is bounded, since $\tilde{c}$ is not exactly equal to c, but it is expected that a few members of Π whose normal vectors slightly differ from c together form a bound on the objective value $c^T x$. Therefore, the constraints with indices in Π, especially those with larger right-hand-side values, are expected to be in the optimal basis, whose constraints form the boundary that the optimal solution is on.

Since members of Π are oriented similarly in space, when considered separately from the remaining constraints, these constraints generally form a broad region

that the feasible and optimal solution of (D) might exist on. The critical role of the set Ω and clusters Ω^i, $i \in \{1, \ldots, \omega\}$ is narrowing down this region. Due to their orientation, the constraints in Ω are not expected to form boundaries that impose upper or lower bounds to the objective function $c^T x$, as for each $l \in \Omega$, $A_{l\bullet} c \approx 0$, on each constraint hyperplane $A_{l\bullet} x = 0$ there are directions that objective increases and reduces at a high rate. Therefore, when considered separately from other constraints, the members of Ω are expected to form a region in $\mathbb{R}^n$ where the objective value can increase and reduce freely. This region can be imagined as a narrow tunnel, since directions with dominant components orthogonal to c are obstructed by the constraints in Ω. Similar to the case for Π, the constraints with larger right-hand-side values in each cluster Ω^i are prior to other constraints in the same cluster Ω^i, $i \in \{1, \ldots, \omega\}$.

The constraints with normal vectors approximately parallel to $-c$ come after the members of Π and Ω. Opposite to the members of Π, a constraint $\tilde{c}x \geq b_{\tilde{c}}$ with normal vector $\tilde{c} \approx -c$ approximates an upper bound of objective value: $c^T x \leq -b_{\tilde{c}}$. Again, constraints with normal vectors approximately parallel to $-c$ are expected to constitute a boundary that imposes an upper bound to the objective value $c^T x$. These constraints might be useful in narrowing down the range for the objective value that the LP solutions attain, when joined with members of Π. Yet, they have lower priority than members of Π and Ω, since as opposed to the members of Π, it is expected that they to not appear in an optimal basis and on the boundary where the optimal solution lies.

A constraint whose normal vector is approximately parallel to $-c$ and whose right-hand side is large follows members of $/Pi$ and $/Omega$ in the order. This is achieved by separating constraints into two sets:

$$S_+ = \{o \in S : \cos \alpha_{o,c} \geq 0\},$$

and

$$S_- = \{o \in S : \cos \alpha_{0,c} < 0\},$$

where V is composed only of members of Π and Ω, at this point.

Selection from S_+ and S_- is made alternatingly to order the remaining constraints. The element v selected from S_+ has the largest $\cos \alpha_{v,c} b_v$. This measure aims to combine the properties of being at an acute angle with c and having a larger right-hand side. Similarly, selection of an element $v \in S_-$ is due to a more acute angle with $-c$ and a large right-hand side. This time, the element with smallest $\cos \alpha_{v,c} b_v$ is selected, since $\cos \alpha_{v,c}$ is negative.

Algorithm 2 starts the ordering by alternating members of Π and Ω. The element of Π with the largest right-hand-side is picked, until Π is exhausted, followed by the element of Ω^i with the largest right-hand-side. One element of Ω follows one element of Π continuing this way; selection from Ω is done by alternating clusters Ω^i, $i \in \{1, \ldots, \omega\}$ at each selection. One element from Π is selected after a selection from any cluster $\Omega_i \subset \Omega$, and this is preferred to picking one element from Π after picking elements from each of Ω^i, $i \in \{1, \ldots, \omega\}$. The reason behind this is that the elements of Π are generally more important for constructing a bounded LP and defining the border where an optimal solution lies, as discussed above.

After the first part of V consisting of elements of Π and Ω is formed, the remaining constraints in $I \setminus V$ are separated into two sets according to whether the angle of their normal vector with c is acute or obtuse. Then, members of the sets are appended to V alternatingly. If for $l \in I \setminus V$, $\cos \alpha_{l,c} \geq 0$, a larger value of $\cos \alpha_{l,c} b_l$ means higher priority. Thus, constraints whose normal vectors have more acute angles with c and whose right-hand-sides are larger have greater priority. For $l \in I \setminus V$ with $\cos \alpha_{l,c} < 0$, a smaller value of $\cos \alpha_{l,c} b_l$ implies higher priority. This function is preferred for assigning higher priorities to approximate parallels of $-c$, so that those with larger right-hand-sides come before others in priority. This way, constraints are ordered in V such that members of Π and Ω come foremost, then the group of approximate parallels of $-c$ follows along with the remaining constraints in I.

In this chapter, efforts until this point were for ordering constraints in order to assure that whenever a small subset of I is considered for constraint selection, this small subset covers most significant constraints for an optimal basis and for

Algorithm 2 Forming Permutation of I that Represents Row Selection Priorities

Input: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, Π, Ω, Ω^i $i \in \{1, \ldots, \omega\}$
Output: V, a permutation of I ordered according to row selection priorities
$S_\Pi \leftarrow \Pi$, $S_\Omega \leftarrow \Omega$, $S_{\Omega^i} \leftarrow \Omega_i$ $i \in \{1, \ldots, \omega\}$
Sort S_Π and S_{Ω^i}, $i \in \{1, \ldots, \omega\}$ according to constraint right-hand-sides
$b_{S_\Pi(o)} \geq b_{S_\Pi(p)}$ if $o < p$
$b_{S_{\Omega^i}(o)} \geq b_{S_{\Omega^i}(p)}$ if $o < p$, $\forall i \in \{1, \ldots, \omega\}$
Form V starting with members of Π and Ω
$odd = true$, $l = 1$, $i = 1$
while $S_\Omega \neq \emptyset$ **do**
  if odd **then**
    if $S_\Pi \neq \emptyset$ **then**
      $V(l) = S_\Pi(1)$
      $S_\Pi \leftarrow S_\Pi \setminus \{S_\Pi(1)\}$
      $l \leftarrow l + 1$
    $odd = false$
  if $odd = false$ **then**
    if $S_{\Omega^i} \neq \emptyset$ **then**
      $V(l) = S_{\Omega^i}(1)$
      $S_{\Omega^i} \leftarrow S_{\Omega^i} \setminus \{S_{\Omega^i}(1)\}$
      $S_\Omega \leftarrow S_\Omega \setminus \{S_{\Omega^i}(1)\}$
      $l \leftarrow l + 1$
      $odd = true$
    $i \leftarrow i - \lfloor \frac{i}{\omega} \rfloor \omega + 1$
Sort $I \setminus V$ according to priorities and append it to V
$S = I \setminus V$
$S_+ = \{o \in S : \cos \alpha_{o,c} \geq 0\}$
$S_- = \{o \in S : \cos \alpha_{0,c} < 0\}$
$\cos \alpha_{S_+(o),c}\, b_{S_+(o)} = A_{S_+(o)\bullet} c\, b_{S_+(o)} \geq \cos \alpha_{S_+(p),c}\, b_{S_+(p)}$ if $o < p$
$\cos \alpha_{S_-(o),c}\, b_{S_-(o)} = A_{S_-(o)\bullet} c\, b_{S_-(o)} \leq \cos \alpha_{S_-(p),c}\, b_{S_-(p)}$ if $o < p$
while $S_+ \cup S_- \neq \emptyset$ **do**
  if $S_- \neq \emptyset$ **then**
    $V(l) = S_-(1)$
    $S_- \leftarrow S_- \setminus \{S_-(1)\}$
    $l \leftarrow l + 1$
  if $S_+ \neq \emptyset$ **then**
    $V(l) = S_+(1)$
    $S_+ \leftarrow S_+ \setminus \{S_+(1)\}$
    $l \leftarrow l + 1$

the definition of the LP feasible region. Comparing promising candidates for constraint selection is important, yet another factor for quick convergence of a row generation algorithm is to shift the objective value as much as possible by each row generation. In the remaining part of this chapter, results of the lower bound analysis in Chapter 2 is used to estimate the change caused in the objective value by a row generation.

For the estimation of the effect of generation of row i_k, consider again the LP problems (D_{k-1}) and (D_k), which are the formulations before and after the kth row generation:

$$
\begin{array}{lll}
 & \text{minimize} & c^T x \\
(D_{k-1}) & \text{subject to} & A_{B_{k-1}\bullet}\, x \;\geq\; b_{B_{k-1}} \\
 & & A_{N_{k-1}\bullet}\, x \;\geq\; b_{N_{k-1}}
\end{array}
$$

After the kth row generation, the LP becomes:

$$
\begin{array}{lll}
 & \text{minimize} & c^T x \\
 & \text{subject to} & A_{B_{k-1}\bullet}\, x \;\geq\; b_{B_{k-1}} \\
(D_k) & & A_{N_{k-1}\bullet}\, x \;\geq\; b_{N_{k-1}} \\
 & & A_{i_k\bullet}\, x \;\geq\; b_{i_k}.
\end{array}
$$

Constraint i_k is considered as a row generation candidate in the discussions in this chapter. A constraint is a row generation candidate only if it is violated by the optimal solution $\hat{x}_{k-1}$ of (D_{k-1}). The set of row generation candidates for the kth row generation, $\bar{V}_{k-1}$ is defined as follows:

$$\bar{V}_{k-1} = \{v \in V_{k-1} : A_{v\bullet}\hat{x}_{k-1} < b_v\}\,. \tag{3.4}$$

The lower bound for the change in the objective value due to a row generation Δ_k, given in (2.30) is used to compare row generation candidates. The letter β and a superscript is used to denote the lower bound for objective value change for a specific row generation candidate $v \in \bar{V}_{k-1}$:

$$\Delta_k^v \geq \beta_k^v = \min_{j \in J_+^v} \frac{c^T \rho_j}{A_{v\bullet}\, \rho_j} \left(b_v - A_{v\bullet}\, \hat{x}_{k-1}\right). \tag{3.5}$$

Here, J_+^v is the set of indices for the edges of $\mathcal{F}_{B_{k-1}}$ that intersect the hyperplane of constraint v.

$$J_+^v = \{j \in J : A_{v\bullet}\, \rho_j > 0\} \tag{3.6}$$

Looking at the lower bound β_k^v, it is possible to verify that several strategies adopted in the ordering of V are actually useful for highlighting more promising row generation candidates. The ideal case of $A_{v\bullet} = c$ guarantees an objective change of $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$, given $J_+^v \neq \emptyset$. 1 is not the highest value possible for $\min_{j \in J_+^v} \dfrac{c^T \rho_j}{A_{v\bullet}\, \rho_j}$, yet the assurance of an increase in the objective value at the level of violation $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$ confirms the strategy of assigning high priority to constraints with normal vector approximately parallel to c.

The advantage of assigning higher priorities to constraints with larger right-hand-sides is obvious in (3.5) by b_v, a larger right-hand-side b_v directly implies a larger lower bound β_k^v.

In a problem with a very large number of constraints, $\bar{V}_{k-1}$ might be a too large set to compare the members of. Conducting computations necessary for the calculation of β_k^v for each member $v \in \bar{V}_{k-1}$ requires an effort that necessitates considering only a small subset of $\bar{V}_{k-1}$. J_+^v is not necessarily a small subset of J, therefore, the heavy part of the effort for the calculation of β_k^v consists of computation of inner-products $A_{v\bullet}\, \rho_j$, $j \in J_+^v$.

Since $\hat{x}_{k-1}$ is at hand by the solution of (D_{k-1}), $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$ can be obtained efficiently for each, $v \in \bar{V}_{k-1}$ by one inner product and one arithmetic operation. Thus, $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$ is the value that is used in combination with the order of V to obtain a small subset $\hat{V}_{k-1}$ of $\bar{V}_{k-1}$ that consists of strong row generation candidates.

The priority order V is exploited by considering only a portion $\bar{\bar{V}}_{k-1} \subset \bar{V}_{k-1}$ of highest priority elements of $\bar{V}_{k-1}$ as row generation candidates. The measure $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$ comes into play for comparison among elements of $\bar{\bar{V}}_{k-1}$. The few elements of $\bar{\bar{V}}_{k-1}$ with largest $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$ values remain to the set $\hat{V}_{k-1}$ to be compared by measures β_k^v, $v \in \hat{V}_{k-1}$. This strategy requires definition of two

more intrinsic parameters ϕ and γ of the algorithm, in addition to α_Π, α_Ω and κ. $\phi \in (0, 1]$ defines $\bar{\bar{V}}_{k-1}$ as the first $\lceil \phi \left| \bar{V}_{k-1} \right| \rceil$ elements of $\bar{V}_{k-1}$. $\min \left\{ \gamma, \left| \bar{\bar{V}}_{k-1} \right| \right\}$ elements of $\bar{\bar{V}}_{k-1}$ that have largest $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$ values constitute $\hat{V}_{k-1}$. This row selection procedure is given in Algorithm 3.

Both of the parameters ϕ and γ should be large enough to allow strong row generation candidates to remain for final comparison, but larger values beyond what achieves this would result in inefficiency due to increase in computational requirements without improving row selection. This study does not cover a discussion of what would be the optimal values of ϕ and γ for highest algorithm efficiency, the values for the intrinsic parameters adopted in computational tests are determined by trial and comparison of several settings for intrinsic parameters α_Π, α_Ω, κ, ϕ and γ.

Algorithm 3 Selection of $\hat{v}$, the Row to be Generated, from the Set V_{k-1}

Input: B_{k-1}, N_{k-1}, V_{k-1}, ϕ, γ
Output: $\hat{v} \in V_{k-1}$, the row to be generated
$\hat{x}_{k-1} = A^{-1}_{B_{k-1}\bullet} b_{B_{k-1}}$
$\bar{V}_{k-1} = \{v \in V_{k-1} : A_{v\bullet} \hat{x}_{k-1} < b_v\}$
$\bar{\bar{V}}_{k-1} = \left\{ \bar{V}_{k-1}(1), \ldots, \bar{V}_{k-1}\left(\lceil \phi \left| \bar{V}_{k-1} \right| \rceil \right) \right\}$
Sort $\bar{\bar{V}}_{k-1}$ according to descending $(b_v - A_{v\bullet}\, \hat{x}_{k-1})$, $v \in \bar{\bar{V}}_{k-1}$
$\hat{V}_{k-1} = \left\{ \bar{\bar{V}}_{k-1}(1), \ldots, \bar{\bar{V}}_{k-1}\left(\min \left\{ \gamma, \left| \bar{\bar{V}}_{k-1} \right| \right\} \right) \right\}$
$\hat{v} = \arg\max_{v \in \hat{V}_{k-1}} \beta^v_k$

The next chapter continues with the discussion about conditions for termination of the algorithm at approximate optimality and warm-start strategies. The presentation of the row generation algorithm as a pseudo-code based on the methods considered in this chapter follows the discussions on approximations and warm-start strategies.

Chapter 4

Approximations, Warmstart and the Algorithm

This chapter begins with the discussion of conditions for terminating the row generation algorithm at approximate optimality, by accepting the optimal solution of the LP (D_k) in the proximity of an optimal solution of (D). Section 4.7 follows with the discussion of warm-start strategies, considering the selection of constraints constructing the initial LP (D_0).

In this chapter the objective vector c and rows $A_{i\bullet}$, $i \in I$ are assumed to be normalized, and the right-hand-side vector b is assumed to be modified accordingly.

4.1 Approximation with Multiplicative Error

In most of the LP problems, especially those arising from real world applications, a solution with a small deviation from optimality is acceptable. Accepting a certain level of error in solution is necessary for another reason: due to the limited precision of computers, error accumulates in the solution vector during the solution process. A multiplicative LP relaxation approach is possible for error

tolerance as follows:

$$\left(D_+^\epsilon\right) \qquad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \geq (1-\epsilon)b \end{array}$$

Here, $b \geq 0$ is assumed, and a relative deviance of $\epsilon \in (0,1)$ from the solution space is accepted. If $\hat{x}$ is an optimal solution of $\left(D_+^\epsilon\right)$, we can write:

$$(1-\epsilon)\, z^* \leq c^T \hat{x} \leq z^* \tag{4.1}$$

where z^* is the optimal value of the original LP (D), which is assumed to be feasible and bounded ($z^* \geq 0$ when the problem is bounded, since by $b \geq 0$, μx, $\mu \geq 1$ is a feasible solution if x is a feasible solution of (D)). The inequality on the left follows from the fact that $\dfrac{\hat{x}}{1-\epsilon}$ is a feasible solution of (D), with objective value $\dfrac{c^T \hat{x}}{1-\epsilon} \geq z^*$. The inequality on the right follows from the fact that the optimal value of a relaxation is at least as good as that of the original LP.

A tolerance of ϵ relative error in objective value is assured by the ϵ multiplicative relaxation as long as the assumption $b \geq 0$ is satisfied. Although a large class of LP problems satisfies this assumption, a general LP in the format used in our analysis may have constraints with positive and negative right-hand-sides together. Therefore, in general, the right-hand-side vector b can be rearranged and separated into two vectors $b_1 \geq 0$ and $b_2 < 0$, resulting in the following LP formulation:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & A_1 x \geq b_1 \\ & A_2 x \geq b_2 \end{array}$$

After multiplicative relaxation is applied, we have:

$$(D^\epsilon) \qquad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & A_1 x \geq (1-\epsilon)\, b_1 \\ & A_2 x \geq (1+\epsilon)\, b_2 \end{array}$$

When the assumption $b \geq 0$ is not satisfied, an ϵ approximation to the solution space does not guarantee ϵ relative error to the optimal value.

Example 1 : Relative error is not bounded for ϵ-relaxation

To show that there exists no bounds for absolute error when ϵ-relaxation is used, let us consider the following LP with right-hand-sides determined by constants b and η such that $0 < \eta < b$.

$$\begin{array}{lrcl} \text{minimize} & x_0 - x_1 & & \\ \text{subject to} & x_0 & \geq & (b+\eta) \\ & -x_1 & \geq & (-b+\eta) \end{array}$$

The constraints are relaxed by a factor ϵ to form the LP:

$$\begin{array}{lrcl} \text{minimize} & x_0 - x_1 & & \\ \text{subject to} & x_0 & \geq & (1-\epsilon)(b+\eta) \\ & -x_1 & \geq & (1+\epsilon)(-b+\eta) \end{array}$$

$z^* = 2\eta$ denotes the optimal value of the original LP, and $z^{**} = 2\eta - 2\epsilon b$ the optimal value of the second LP. Defining relative error as $\bar{\epsilon}$, the relative error can be arbitrarily high depending on the values of b and η.

$$\bar{\epsilon} = \frac{|z^* - z^{**}|}{|z^*|} = \frac{2\epsilon b}{2\eta} = \frac{\epsilon b}{\eta}$$

The error $\bar{\epsilon}$ therefore can take the value of arbitrary positive multiple of ϵ depending on the ratio b/η.

4.1.1 Absolute error in optimal value with multiplicative relaxation

Although a bound on the relative error in the case of multiplicative error for a general LP can not be established, and relative error may not exist when optimal value is 0, a bound on the absolute error exists for an LP with an optimal solution.

The LP problem and its dual are given by:

$$
\begin{array}{rlll}
 & \text{minimize} & c^T x & \\
(D) & \text{subject to} & A_1 x \geq b_1 & \\
 & & A_2 x \geq b_2 &
\end{array}
\qquad
\begin{array}{rlll}
 & \text{maximize} & b_1^T w_1 + b_2^T w_2 & \\
(P) & \text{subject to} & A_1^T w_1 + A_2^T w_2 & = \quad c \\
 & & w_1,\ w_2 & \geq \quad 0
\end{array}
$$

The relaxed LP and its dual are given by:

$$
\begin{array}{rll}
 & \text{minimize} & c^T x \\
(D^\epsilon) & \text{subject to} & A_1 x \geq (1-\epsilon)\, b_1 \\
 & & A_2 x \geq (1+\epsilon)\, b_2
\end{array}
$$

$$
\begin{array}{rll}
 & \text{maximize} & b_1^T w_1 + b_2^T w_2 - \epsilon\, b_1^T w_1 + \epsilon\, b_2^T w_2 \\
(P^\epsilon) & \text{subject to} & A_1^T w_1 + A_2^T w_2 = c \\
 & & w_1,\ w_2 \geq 0
\end{array}
$$

Let z^* be the optimal value of (D), and z^{**} be the optimal value of (D^ϵ), which we know to exist since the feasible regions of (P^ϵ) and (P) are the same and the feasible region of (D^ϵ) covers that of (D^ϵ). Therefore both primal and dual feasible regions for the relaxed problem are non-empty. There is an optimal extreme point w^* of (P), also feasible to (P^ϵ). Then:

$$
\begin{aligned}
z^* &\geq z^{**} \\
&\geq b_1^T w_1^* + b_2^T w_2^* - \epsilon\, b_1^T w_1^* + \epsilon\, b_2^T w_2^* \\
&= z^* - \epsilon\, b_1^T w_1^* + \epsilon\, b_2^T w_2^* \qquad (4.2)
\end{aligned}
$$

Then the absolute error is bounded above:

$$
\begin{aligned}
|z^* - z^{**}| &\leq \epsilon \begin{bmatrix} b_1 \\ -b_2 \end{bmatrix}^T \begin{bmatrix} w_1^* \\ w_2^* \end{bmatrix} \\
&= \epsilon \begin{bmatrix} b_{1,B} \\ -b_{2,B} \end{bmatrix}^T \begin{bmatrix} w_{1,B}^* \\ w_{2,B}^* \end{bmatrix} \qquad (4.3)
\end{aligned}
$$

where $\begin{bmatrix} w^*_{1,B} \\ w^*_{2,B} \end{bmatrix}$ are the dual basic variables, and $\begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix}$ are the corresponding dual objective function coefficients.

$$\begin{bmatrix} w^*_{1,B} \\ w^*_{2,B} \end{bmatrix} = (A_{B^*\bullet})^{-1}c \tag{4.4}$$

where B^* is the optimal basis of (P).

$$\begin{aligned}
|z^* - z^{**}| &\leq \epsilon \begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix}^T (A_{B^*\bullet})^{-1}c \\
&= \epsilon \left| \begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix}^T (A_{B^*\bullet})^{-1}c \right| \\
&\leq \epsilon \left\| \begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix} \right\|_2 \left\| (A_{B^*\bullet})^{-1}c \right\|_2 \\
&\leq \epsilon \left\| \begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix} \right\|_2 \sup_{u\in\mathbb{R}^n:\|u\|_2=1} \left\{ \left\| (A_{B^*\bullet})^{-1}u \right\|_2 \right\} \\
&= \epsilon \left\| \begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix} \right\|_2 \left\| (A_{B^*\bullet})^{-1} \right\|_2 \\
&\leq \epsilon \left\| \begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix} \right\|_2 \max_{B\in\mathcal{B}\left(\left[A_1^T A_2^T\right]\right)} \left\{ \left\| (A_{B\bullet})^{-1} \right\|_2 \right\} \\
&\leq \epsilon\, \beta\, \chi\left(\left[A_1^T A_2^T\right]\right)
\end{aligned} \tag{4.5}$$

Here, β is the largest norm that $\begin{bmatrix} b_{1,B} \\ b_{2,B} \end{bmatrix}$ can attain, i.e., the norm of the vector composed of n elements of b that are larger in absolute value than the remaining entries of b.

$$\beta = \left(\sum_{j=1}^{n} {b_{i_j}}^2\right)^{1/2}, \quad \{i_1, \ldots, i_n\} \subseteq \{1, \ldots, m\} \tag{4.6}$$

$$|b_i| \geq |b_{i'}| \text{ if } i \in \{i_1, \ldots, i_n\}, \quad i' \in \{1, \ldots, m\} \setminus \{i_1, \ldots, i_n\} \tag{4.7}$$

The measure $\chi(A)$ of a matrix A is the maximal determinant of inverted bases of A [83].

4.2 Approximation with Additive Error

Due to the difficulty encountered in multiplicative relaxation when both positive and negative right-hand-side coefficients exist, additive relaxation is considered, resulting in the following LP:

$$\left(D^{\delta}\right) \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & a_i^T x \geq b - \delta \|a_i\| \qquad i \in \{1..m\} \end{array}$$

or, in matrix form:

$$\left(D^{\delta}\right) \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \geq b - \delta \hat{a} \end{array}$$

where $\hat{a} = \begin{bmatrix} \|a_1\| \\ . \\ . \\ . \\ \|a_m\| \end{bmatrix}$, the vector composed of constraint normal vector norms, and δ is a positive real number.

For simplicity, the objective function and the constraint rows of the LP can be normalized. Since the magnitude of the optimal value, but not the optimal

solution, changes by multiplying the objective function coefficient vector by a positive scalar, normalizing this vector does not affect the LP solution. Again, by dividing each row including the right-hand-side by the norm of the constraint vector, the solutions space remains the same, and the LP takes the following form:

$$\begin{aligned} & & \text{minimize} \quad & \frac{c^T}{\|c\|}x \\ (D^\delta) & & & \\ & & \text{subject to} \quad & \frac{a_i^T}{\|a_i\|}x \geq \frac{b}{\|a_i\|} - \delta \qquad i \in \{1..m\} \end{aligned}$$

From this point on, rows of the matrix A, and the vector c are assumed be normalized, and b is assumed to be modified accordingly. After the additive relaxation of constraints, the LP becomes:

$$\begin{aligned} (D^\delta) \quad & \text{minimize} \quad c^T x \\ & \text{subject to} \quad Ax \geq b - \delta \times \vec{1} \end{aligned}$$

where

$$A = \begin{bmatrix} — & \frac{a_1^T}{\|a_1\|} & — \\ & . & \\ & . & \\ & . & \\ — & \frac{a_m^T}{\|a_m\|} & — \end{bmatrix}, \quad b = \begin{bmatrix} \frac{b_1}{\|a_1\|} \\ . \\ . \\ . \\ \frac{b_m}{\|a_m\|} \end{bmatrix},$$

$$\vec{\delta} = \delta \times \vec{1}_{m\times 1} \quad \text{and} \quad \|c\| = 1$$

4.2.1 Error in optimal value with additive relaxation

The LP problem and its dual are given by:

$$(D)\quad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \geq b \end{array} \qquad (P)\quad \begin{array}{ll} \text{maximize} & b^T w \\ \text{subject to} & A^T w = c \\ & w \geq 0 \end{array}$$

The relaxed LP and its dual:

$$\left(D^\delta\right)\quad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \geq b - \delta\vec{1} \end{array} \qquad \left(P^\delta\right)\quad \begin{array}{ll} \text{maximize} & b^T w - \delta\vec{1}^T w \\ \text{subject to} & A^T w = c \\ & w \geq 0 \end{array}$$

In the case where both (D) and (P) are feasible and bounded, let $z^{(D)}$ denote the optimal value of the original problem and $z^{\left(D^\delta\right)}$ denote the optimal value of the relaxed problem. Since the feasible region of (P) and $\left(P^\delta\right)$ are the same, $\left(P^\delta\right)$ is feasible; and again, $\left(D^\delta\right)$ is feasible since its feasible region covers that of (D). Let $w^{(P)}$ be an optimal extreme point of (P), hence an extreme point of $\left(P^\delta\right)$ that is not necessarily optimal for $\left(P^\delta\right)$. Then:

$$z^{(D)} \geq z^{\left(D^\delta\right)} \geq b^T w^{(P)} - \delta\vec{1}^T w^{(P)} = z^{(D)} - \delta\vec{1}^T w^{(P)}. \tag{4.8}$$

In the absence of information about extreme points before attempting the solution of the LP, an upper bound on the additive error $\bar{\delta} = \left| z^{(D)} - z^{\left(D^\delta\right)} \right|$ can be given as follows:

$$\begin{aligned} \bar{\delta} &\leq \max_{w \in V^{(P)}} \left\{ \delta\vec{1}^T w \right\} \\ &\leq \max_{B \in \mathcal{B}(A^T): A_{B\bullet}^{-T} c \geq 0} \left\{ \delta\vec{1}^T A_{B\bullet}^{-1} c \right\} \\ &= \delta \max_{B \in \mathcal{B}(A^T): A_{B\bullet}^{-T} c \geq 0} \left\{ \left\langle \vec{1}, A_{B\bullet}^{-T} c \right\rangle \right\} \end{aligned} \tag{4.9}$$

where $V^{(P)}$ is the set of vertices of the feasible region $\left\{ w : A^T w = c,\, w \geq 0 \right\}$ of (P), and $\mathcal{B}\left(A^T\right)$ is the set of all bases of A^T. This holds since every vertex w_0

of $\left\{w : A^T w = c,\, w \geq 0\right\}$ can be represented by a basis B_0 of A^T as $w_0 = A_{B_0 \bullet}^{-T} c$, and any basis $B \in \mathcal{B}\left(A^T\right)$ satisfying $A_{B\bullet}^{-T} c \geq 0$ represents an extreme point of $\left\{w | A^T w = c\right\}$. As $\|c\| = 1$, we can write:

$$\begin{aligned}
\max_{B \in \mathcal{B}(A^T): A_{B\bullet}^{-T} c \geq 0} \left\{ \left\langle \vec{1}, A_{B\bullet}^{-T} c \right\rangle \right\} &\leq \sqrt{n} \max_{B \in \mathcal{B}(A^T): A_{B\bullet}^{-T} c \geq 0} \left\{ \left\| A_{B\bullet}^{-T} c \right\| \right\} \\
&\leq \sqrt{n} \max_{B \in \mathcal{B}(A^T): A_{B\bullet}^{-T} c \geq 0} \left\{ \sup_{u \in \mathbb{R}^n : \|u\| = 1} \left\{ \left\| A_{B\bullet}^{-T} u \right\|_2 \right\} \right\} \\
&= \sqrt{n} \max_{B \in \mathcal{B}(A^T): A_{B\bullet}^{-T} c \geq 0} \left\{ \left\| A_{B\bullet}^{-T} \right\|_2 \right\} \\
&\leq \sqrt{n} \max_{B \in \mathcal{B}(A^T)} \left\{ \left\| A_{B\bullet}^{-T} \right\|_2 \right\} \\
&= \sqrt{n}\, \chi\left(A^T\right) \qquad (4.10)
\end{aligned}$$

Thus

$$\bar{\delta} = \left| z^{(D)} - z^{\left(D^\delta\right)} \right| \leq \delta\, \chi\left(A^T\right) \sqrt{n}. \qquad (4.11)$$

4.3 A New Approach to Multiplicative Relaxation

We can consider another relaxation for the LP:

The LP problem and its dual given by:

$$(D) \quad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \geq b \end{array} \qquad (P) \quad \begin{array}{ll} \text{maximize} & b^T w \\ \text{subject to} & A^T w = c \\ & w \geq 0 \end{array}$$

The relaxed LP and its dual given by:

$$
\begin{array}{lll}
\left(D^{\lambda}\right) & \text{minimize} & c^T x \\
& \text{subject to} & Ax - b\lambda \geq 0 \\
& & -\lambda \geq -1 - \epsilon \\
& & \lambda \geq 1 - \epsilon
\end{array}
\qquad
\begin{array}{lll}
\left(P^{\lambda}\right) & \text{maximize} & w_2 - w_1 - \epsilon\,(w_1 + w_2) \\
& \text{subject to} & A^T w = c \\
& & -b^T w - w_1 + w_2 = 0 \\
& & w \geq 0; \;\; w_1,\; w_2 \geq 0
\end{array}
$$

In this relaxation scheme, ϵ is assumed to lie in the interval $[0, 1)$. We can show that relative error is bounded by ϵ in this approximation approach.

If (D) is a feasible and bounded LP problem, $\left(D^{\lambda}\right)$ also has a feasible solution with $\lambda = 0$. For a feasible solution w_0 of (P), a modified solution to $\left(P^{\lambda}\right)$ is $(w, w_1, w_2) = (w_0, max(-b^T w_0, 0), max(b^T w_0, 0))$. Since both $\left(D^{\lambda}\right)$ and $\left(P^{\lambda}\right)$ are feasible, the relaxed problem has an optimal solution.

4.3.1 Error in optimal value with λ relaxation

Let $z^{(D)}$ be the optimal value of (D) and $z^{\left(D^{\lambda}\right)}$ be the optimal value of $\left(D^{\lambda}\right)$. We will show that

$$z^{(D)} - \epsilon \left| z^{(D)} \right| \leq z^{\left(D^{\lambda}\right)} \leq z^{(D)}. \tag{4.12}$$

Let $x^{(D)}$ be an optimal solution of (D). Then, $\left(x^{(D)}, 1\right)$ is a feasible solution of $\left(D^{\lambda}\right)$, with objective value $c^T x^{(D)} = z^{(D)}$ implying:

$$z^{\left(D^{\lambda}\right)} \leq z^{(D)}. \tag{4.13}$$

Now let $\left(x^{D^{\lambda}}, \lambda^*\right)$ be an optimal solution of $\left(D^{\lambda}\right)$. Note that $\lambda^* > 0$, since the solution is a feasible solution of $\left(D^{\lambda}\right)$. Then

$$Ax^{D^{\lambda}} - b\lambda^* \geq 0, \tag{4.14}$$

that is,

$$\frac{Ax^{D^{\lambda}}}{\lambda^*} \geq b, \tag{4.15}$$

therefore $\dfrac{x^{D^{\lambda}}}{\lambda^*}$ is a feasible solution of (D) with objective value

$$\frac{c^T x^{D^{\lambda}}}{\lambda^*} = \frac{z^{\left(D^{\lambda}\right)}}{\lambda^*}. \tag{4.16}$$

Then,

$$z^{(D)} \leq \frac{z^{\left(D^{\lambda}\right)}}{\lambda^*} \tag{4.17}$$

$$\lambda^* z^{(D)} \leq z^{\left(D^{\lambda}\right)} \tag{4.18}$$

$$\lambda^* z^{(D)} \leq z^{\left(D^{\lambda}\right)} \leq z^{(D)} \tag{4.19}$$

By inequality (4.19), it can be inferred that $\lambda^* \leq 1$ if $z^{(D)} > 0$ and $\lambda^* \geq 1$ if $z^{(D)} < 0$, so that:

$$z^{(D)} - |1 - \lambda^*| \left|z^{(D)}\right| \leq z^{\left(D^{\lambda}\right)} \leq z^{(D)} \tag{4.20}$$

Noting that $\lambda^* \in [1-\epsilon, 1+\epsilon]$, we have $(1-\lambda^*) \in [-\epsilon, \epsilon]$, which completes the proof:

$$z^{(D)} - \epsilon \left|z^{(D)}\right| \leq z^{\left(D^{\lambda}\right)} \leq z^{(D)}. \tag{4.21}$$

In the case $z^{(D)} = 0$, this holds with an equality; $z^{(D)} = z^{\left(D^{\lambda}\right)}$. Then, we can write

$$\left|z^{(D)} - z^{\left(D^{\lambda}\right)}\right| \leq \epsilon \left|z^{(D)}\right|. \tag{4.22}$$

4.3.2 Absolute error in λ relaxation

The absolute error in the λ relaxation scheme, i.e., $\left|z^{(D)} - z^{\left(D^{\lambda}\right)}\right|$ is bounded above by $\epsilon \left|z^{(D)}\right|$. In the absence of information about the optimal value and the optimal basis, we can construct a looser bound:

$$\begin{aligned} \epsilon \left|z^{(D)}\right| \quad = \quad \epsilon \left|b^T w^*\right| \quad &\leq \quad \epsilon \max_{B \in \mathcal{B}(A^T): A_{B\bullet}^{-T} c \geq 0} \left\{\left|b_B^T A_{B\bullet}^{-T} c\right|\right\} \\ &\leq \quad \epsilon \left\|b_B\right\|_2 \max_{B \in \mathcal{B}(A^T)} \left\{\left\|A_{B\bullet}^{-T}\right\|_2\right\} \|c\| \end{aligned} \tag{4.23}$$

w^* is an optimal basic feasible solution of (P), thus having a corresponding basis in $\mathcal{B}\left(A^T\right)$, the set of bases of A^T, as defined before. When we replace $\chi\left(A^T\right)$ with its definition in the expression, and since c is a unit vector, the bound becomes:

$$\left|z^{(D)} - z^{\left(D^{\lambda}\right)}\right| \leq \epsilon \left|z^{(D)}\right| \quad \leq \quad \epsilon \left\|b_B\right\|_2 \chi\left(A^T\right) \leq \epsilon \beta \chi\left(A^T\right), \tag{4.24}$$

where β was previously defined.

4.4 LP Format Does not Affect The Relaxed Optimal Value

In this section we show that, the relaxation of a generic LP has the same optimal value with the relaxed problem obtained after the generic LP is converted to the format that is used in this study. A generic LP can be received from the user with all types of constraints and variable domains as follows:

(P_{gen})

$$\begin{aligned}
&\text{minimize} && c_1^T x_1 + c_2^T x_2 + c_3^T x_3 && && \\
&\text{subject to} && A_{11}x_1 + A_{12}x_2 + A_{13}x_3 && \geq && b_1 && (4.25)\\
& && A_{21}x_1 + A_{22}x_2 + A_{23}x_3 && \leq && b_2 && (4.26)\\
& && A_{31}x_1 + A_{32}x_2 + A_{33}x_3 && = && b_3 && (4.27)\\
& && x_1 \geq 0,\ x_2 \leq 0,\ x_3 \text{ free} && && , && (4.28)
\end{aligned}$$

where $A_{ij},\ i,j \in \{1,2,3\}$ are matrices and $b_i,\ i \in \{1,2,3\}$ are vectors of appropriate sizes. In such a generic LP format, where all three constraint types are allowed, we can without loss of generality assume that all right-hand side vectors are non-negative. Then this LP can be multiplicatively relaxed by a factor ϵ. Equality constraints are relaxed through both sides of the hyperplane, represented by two constraints, one being less and the other greater-than-or-equal type constraints.

$\left(P_{gen}^{\epsilon}\right)$

$$\begin{aligned}
&\text{minimize} && c_1^T x_1 + c_2^T x_2 + c_3^T x_3 && && \\
&\text{subject to} && A_{11}x_1 + A_{12}x_2 + A_{13}x_3 && \geq && (1-\epsilon)\, b_1 && (4.29)\\
& && A_{21}x_1 + A_{22}x_2 + A_{23}x_3 && \leq && (1+\epsilon)\, b_2 && (4.30)\\
& && A_{31}x_1 + A_{32}x_2 + A_{33}x_3 && \geq && (1-\epsilon)\, b_3 && (4.31)\\
& && A_{31}x_1 + A_{32}x_2 + A_{33}x_3 && \leq && (1+\epsilon)\, b_3 && (4.32)\\
& && x_1 \geq 0,\ x_2 \leq 0,\ x_3 \text{ free} && && && (4.33)
\end{aligned}$$

The following LP is the conversion of (P_{gen}) to the format we use:

(D)

$$
\begin{aligned}
\text{minimize} \quad & c_1^T x_1 + c_2^T x_2 + c_3^T x_3 & & & \\
\text{subject to} \quad & A_{11}x_1 + A_{12}x_2 + A_{13}x_3 & \geq & b_1 & (4.34)\\
& -A_{21}x_1 - A_{22}x_2 - A_{23}x_3 & \geq & -b_2 & (4.35)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 & \geq & b_3 & (4.36)\\
& -A_{31}x_1 - A_{32}x_2 - A_{33}x_3 & \geq & -b_3 & (4.37)\\
& x_1 & \geq & 0 & (4.38)\\
& -x_2 & \geq & 0 & (4.39)
\end{aligned}
$$

The multiplicative relaxation to this problem is:

(D^ϵ)

$$
\begin{aligned}
\text{minimize} \quad & c_1^T x_1 + c_2^T x_2 + c_3^T x_3 & & & (4.40)\\
\text{subject to} \quad & A_{11}x_1 + A_{12}x_2 + A_{13}x_3 & \geq & (1-\epsilon)\, b_1 & (4.41)\\
& -A_{21}x_1 - A_{22}x_2 - A_{23}x_3 & \geq & (1+\epsilon)\,(-b_2) & (4.42)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 & \geq & (1-\epsilon)\, b_3 & (4.43)\\
& -A_{31}x_1 - A_{32}x_2 - A_{33}x_3 & \geq & (1+\epsilon)\,(-b_3) & (4.44)\\
& x_1 & \geq & 0 & (4.45)\\
& -x_2 & \geq & 0 & (4.46)
\end{aligned}
$$

With constraints (4.42) and (4.44) negated, $\left(P^\epsilon_{gen}\right)$ and (D^ϵ) are the same LP problems. Therefore fixing the format of LP input as $min\left\{c^T x : Ax \geq b\right\}$ does not affect the LP relaxation and its optimal value.

Again, the additive relaxation of the LP in generic format and its comparison with its additive relaxation after conversion to the format we use only differ in the multiplication of constraints by -1:

$\left(P^{\delta}_{gen}\right)$

$$\begin{aligned}
\text{minimize} \quad & c_1^T x_1 + c_2^T x_2 + c_3^T x_3 \\
\text{subject to} \quad & A_{11}x_1 + A_{12}x_2 + A_{13}x_3 \geq b_1 - \delta\hat{a}_1 && (4.47)\\
& A_{21}x_1 + A_{22}x_2 + A_{23}x_3 \leq b_2 + \delta\hat{a}_2 && (4.48)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 \geq b_3 - \delta\hat{a}_3 && (4.49)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 \leq b_3 + \delta\hat{a}_3 && (4.50)\\
& x_1 \geq -\delta\vec{1},\ x_2 \leq \delta\vec{1},\ x_3 \text{ free} && (4.51)
\end{aligned}$$

$\left(D^{\delta}\right)$

$$\begin{aligned}
\text{minimize} \quad & c_1^T x_1 + c_2^T x_2 + c_3^T x_3 \\
\text{subject to} \quad & A_{11}x_1 + A_{12}x_2 + A_{13}x_3 \geq b_1 - \delta\hat{a}_1 && (4.52)\\
& -A_{21}x_1 - A_{22}x_2 - A_{23}x_3 \geq -b_2 - \delta\hat{a}_2 && (4.53)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 \geq b_3 - \delta\hat{a}_3 && (4.54)\\
& -A_{31}x_1 - A_{32}x_2 - A_{33}x_3 \geq -b_3 - \delta\hat{a}_3 && (4.55)\\
& x_1 \geq -\delta\vec{1} && (4.56)\\
& -x_2 \geq \delta\vec{1} && (4.57)
\end{aligned}$$

The parameter $\hat{a}$ is defined by:

$$\hat{a}_i = \begin{bmatrix} \|a_{i1}^T\| \\ . \\ . \\ . \\ \|a_{im_i}^T\| \end{bmatrix}, \qquad i \in \{1,2,3\}, \tag{4.58}$$

where

$$[A_{i1}A_{i2}A_{i3}] = \begin{bmatrix} — & a_{i1}^T & — \\ & . & \\ & . & \\ & . & \\ — & a_{im_i}^T & — \end{bmatrix}, \qquad i \in \{1,2,3\}, \tag{4.59}$$

and m_i is the number of rows of A_{i1}, $i \in \{1, 2, 3\}$.

For λ relaxation case, generic format LP and the LP in format adopted in this study are obviously the same except that the rows (4.68), (4.70), (4.72) and (4.74) of $\left(D^{\lambda}\right)$ are in negated form. They are as follows:

$\left(P^{\lambda}_{gen}\right)$

$$\begin{aligned}
\text{minimize} \qquad & c_1^T x_1 + c_2^T x_2 + c_3^T x_3 & & & \\
\text{subject to} \quad & A_{11}x_1 + A_{12}x_2 + A_{13}x_3 - b_1\lambda & \geq & 0 & (4.60)\\
& A_{21}x_1 + A_{22}x_2 + A_{23}x_3 - b_2\lambda & \leq & 0 & (4.61)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 - b_3\lambda & \geq & 0 & (4.62)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 - b_3\lambda & \leq & 0 & (4.63)\\
& \lambda & \geq & 1 - \epsilon & (4.64)\\
& \lambda & \leq & 1 + \epsilon & (4.65)\\
& x_1 \geq 0,\ x_2 \leq 0,\ x_3 \text{ free} & & & (4.66)
\end{aligned}$$

$\left(D^{\lambda}\right)$

$$\begin{aligned}
\text{minimize} \qquad & c_1^T x_1 + c_2^T x_2 + c_3^T x_3 & & & \\
\text{subject to} \quad & A_{11}x_1 + A_{12}x_2 + A_{13}x_3 - b_1\lambda & \geq & 0 & (4.67)\\
& -A_{21}x_1 - A_{22}x_2 - A_{23}x_3 + b_2\lambda & \geq & 0 & (4.68)\\
& A_{31}x_1 + A_{32}x_2 + A_{33}x_3 - b_3\lambda & \geq & 0 & (4.69)\\
& -A_{31}x_1 - A_{32}x_2 - A_{33}x_3 + b_3\lambda & \geq & 0 & (4.70)\\
& x_1 & \geq & 0 & (4.71)\\
& -x_2 & \geq & 0 & (4.72)\\
& \lambda & \geq & 1 - \epsilon & (4.73)\\
& -\lambda & \geq & -1 - \epsilon & (4.74)
\end{aligned}$$

4.5 Termination at (Approximate) Optimality for δ and ϵ Relaxation Schemes

Algorithm 5 states that at least one row from $Ax \geq b$ is generated at each step until termination. An LP has a finite number of constraints, therefore the algorithm terminates at an (approximate) optimal solution, given that it is feasible and bounded. Then, for some k $(0 \leq k \leq m)$, k is the number of constraints of the LP at termination step, and there is an optimal solution $\hat{x}_k$ to the LP:

$$\begin{array}{rll} & \text{minimize} & c^T x \\ (D_k) & \text{subject to} & A_{B_k\bullet} x \geq b_{B_k} \\ & & A_{N_k\bullet} x \geq b_{N_k} \end{array}$$

(D_k) is the termination step if and only if $\hat{x}_k$ satisfies $A_{V_k\bullet}\hat{x}_k \geq b_{V_k} - \vec{\delta}$, in δ relaxation case, for example. After the solution of (D_k), the remaining constraints are checked for whether they are satisfied under the relevant relaxation scheme, δ (additive) or ϵ (multiplicative). If all the remaining constraints are satisfied under relaxation, then $\hat{x}_k$ is a solution that attains the user-defined precision level to the optimal solution of (D). This holds for δ and ϵ relaxation schemes, since

$$z^{(D)} \geq z^{\left(D_k^\delta\right)} = c^T \hat{x}_k \geq z^{\left(D^\delta\right)}.$$

$z^{(D)} \geq z^{\left(D_k^\delta\right)}$ follows from the fact that $\left(D_k^\delta\right)$ (or (D_k^ϵ)) is a relaxation of (D):

$$\begin{array}{rll} & \text{minimize} & c^T x \\ \left(D_k^\delta\right) & \text{subject to} & A_{B_k\bullet} x \geq b_{B_k} \\ & & A_{N_k\bullet} x \geq b_{N_k} \\ & & A_{V_k\bullet} x \geq b_{V_k} - \vec{\delta} \end{array}$$

Similarly, for ϵ relaxation $z^{(D)} \geq z^{\left(D_k^\epsilon\right)}$:

$$
\begin{array}{lll}
\text{minimize} & c^T x & \\
\text{subject to} & A_{B_k\bullet} x & \geq b_{B_k} \\
(D_k^\epsilon) & A_{N_k\bullet} x & \geq b_{N_k} \\
& A_{V_k^1\bullet} x & \geq b_{V_k^1} - \epsilon b_{V_k^1} \\
& A_{V_k^2\bullet} x & \geq b_{V_k^2} + \epsilon b_{V_k^2}
\end{array}
$$

V_k is separated into two sets V_k^1 and V_k^2, indices of non-negative right-hand sides being in V_k^1 and indices of negative right-hand sides being in V_k^2.

We have $z^{\left(D_k^\delta\right)} = c^T \hat{x}_k$ as $\hat{x}_k$ is assured to be feasible to $\left(D_k^\delta\right)$ by checking the constraints left out of (D_k). Since all of its constraints are relaxed, $\left(D^\delta\right)$ is a relaxation of $\left(D_k^\delta\right)$, therefore the optimal value of $\left(D^\delta\right)$ is smaller than that of $\left(D_k^\delta\right)$. We write $z^{\left(D_k^\delta\right)} = c^T \hat{x}_k \geq z^{\left(D^\delta\right)}$. By a similar argument, (D^ϵ) is a relaxation of (D_k^ϵ) and $z^{\left(D_k^\epsilon\right)} = c^T \hat{x}_k \geq z^{(D^\epsilon)}$.

4.6 Termination at (approximate) optimality for λ relaxation scheme

After the LP (D_k) is solved and an optimal solution $\hat{x}_k$ is found, it is not sufficient to check the constraints that are left out to find out whether $\hat{x}_k$ is an approximate solution according to the λ relaxation scheme. λ relaxation does not necessarily loosen every constraint; when the option to relax a constraint is chosen, this might result in tightening of another constraint under λ relaxation. If a value greater than 1 is chosen for λ, this relaxes the constraints with negative right-hand-sides and tightens the constraints with positive right-hand-sides. The opposite happens when a value less than 1 is chosen for λ. If $\hat{x}_k$ satisfies one of the constraints that are left out by slightly relaxing this constraint, this relaxation might be accompanied by the tightening of one of the constraints in (D_k) to a point that it is no longer satisfied. Therefore, after solving (D_k), we have to check every constraint of (D) to see if $\hat{x}_k$ is feasible to a λ relaxation of (D).

On trying to see if $\hat{x}_k$ is feasible to a λ relaxation of (P), we check each constraint of (D) for whether the constraint is satisfied by $\hat{x}_k$, in which case the maximum amount this constraint allows for tightening is investigated. If the constraint is not satisfied, then we find out how much the constraint has to be relaxed in order for $\hat{x}_k$ to satisfy it. Each constraint with a positive right-hand side value determines an upper-bound value for λ, indicating allowance for tightening if this upper-bound value is greater than one, and a need for relaxation if it is less than one. Similarly, investigation of each constraint with a negative right-hand side value determines a lower-bound value for λ, indicating an allowance if it is below one and a need for relaxation if it is above 1. Since constraints with zero right-hand sides are not affected by the λ relaxation, if these are not satisfied by $\hat{x}_k$, then $\hat{x}_k$ is not an approximate solution of (D) under the λ relaxation scheme. Noting these, a procedure to check if $\hat{x}_k$ is feasible for λ relaxation of (D) is given in Algorithm 4.

Algorithm 4 Checking approximate feasibility of $\hat{x}_k$ under λ relaxation

$I_0 = \{i \in I : b_i = 0\}$, $I_+ = \{i \in I : b_i > 0\}$, $I_- = \{i \in I : b_i < 0\}$
if $A_{I_0\bullet}\hat{x}_k \geq b_{I_0}$ **then**
  $\lambda_{min} = max_{i \in I_-} \dfrac{A_{i\bullet}\hat{x}_k}{b_i}$
  $\lambda_{max} = min_{i \in I_+} \dfrac{A_{i\bullet}\hat{x}_k}{b_i}$
  if $1 - \epsilon \leq \lambda_{min} \leq \lambda_{max} \leq 1 + \epsilon$ **then**
    $\hat{x}_k$ is approximately feasible to (D) under λ relaxation
  else
    $\hat{x}_k$ is not approximately feasible to (D) under λ relaxation
else
  $\hat{x}_k$ is not approximately feasible to (D) under λ relaxation

4.7 Warm start Strategies: Construction of initial set of constraints

A Big-M type start is applicable for the dual row generation algorithm. The starting basis could be constructed by additional constraints: $Ix \geq -M$. A finite value for M can be found for each problem so that an optimal solution $\hat{x}$ of (D)

satisfies $I\hat{x} \geq -M$, given that (D) is a feasible and bounded LP problem. Once this initial LP is constructed, the algorithm continues by row generation, if this is necessary due to violation of constraints. The row generation procedure returns an (approximate) optimal solution of (D), since at least one optimal solution exists in the intersection of the feasible region of (D) and $\{x \in \mathbb{R}^n : Ix \geq -M\}$.

Despite the applicability of a Big-M type start, the fact that feasibility is not a requirement for the initial basis, or any basis until the final solution, promotes the opportunity for warm-start strategies. Since feasibility is not a requirement, any linearly independent group of n dual rows is eligible for an initial basis. A smart selection of this group of n linearly independent dual rows provides a warm-start strategy.

In the selection of the initial basis, dual rows with the most acute angle with the objective vector c can be given high priority. The constraints corresponding to these rows might form a lower bound for the objective function, since they designate halfspaces in closely opposite direction to the objective gradient $-c$. With this warm-start strategy, minimizing the gap between the initial objective value and the optimal objective value is attempted. If a quick convergence to the optimal value of (D) is aimed, this strategy can be useful due to shortening the path for convergence. Oppositely, the rows an with acute angle with the objective gradient $-c$ can be preferred. This strategy is likely to lead to an unbounded initial system, which is not critical at initialization, yet might provide an upper-bound for the unknown optimal objective value. This strategy can be used to test efficiency of the algorithm when encountering unboundedness of subproblems (D_k).

If the computational cost is considered bearable, another warm-start strategy can be the clustering of all constraint normal vectors, then picking one constraint from each cluster with largest right-hand side value in order to form an initial basis. This selection can be considered a geometric approximation for the LP polyhedron. For each direction representing a constraint hyperplane, a tight constraint with approximate direction is selected for the initial LP. However, instead of incurring the cost of clustering all of the constraint vectors, the strategy can be

applied through the orthogonal group constraints, which are already clustered. In this case, the initial basis may not provide an overall representation of the polyhedron of the original LP. It might not constitute a bounded LP either. Still, this strategy forms an initial geometry as narrow as possible, so that more accurate estimates of the objective changes are made.

Combining warm-start strategies is possible, and the number of constraints of the initial LP is not limited to a set that is sufficient to form a basis. Therefore, a selection involving several constraints from the parallel group, anti parallel group and each cluster is possible. To sum up, making a constraint selection for warm-start is based on principles similar to those underlying sorting constraint indices in V prior to the solution process so that strong row generation candidates come before others. The warm-start strategy implemented in this study is based on the permuted set V, whose construction is discussed in Chapter 3.

A parameter of the warm-start strategy along with V is s, the number of rows that should be appended to the constraint matrix during the construction of the initial LP (D_0). (D_0) is constructed by selecting a basis from V followed by s row generations according to the priorities.

The formation of an initial basis is by the detection of first n indices from V that correspond to n linearly independent rows. Gaussian elimination with complete pivoting is used to achieve this. The elimination procedure proceeds by selecting as a pivot the row and column pair that contains the element with largest absolute value. The name complete pivoting is due to scanning both rows and columns for the pivot element. Initially, the rows $A_{V(1)\bullet}, \ldots, A_{V(n)\bullet}$ go through the elimination process, but if these are not linearly independent, the elimination continues through the remaining indices of V until n^{th} pivot, i.e., n linearly independent rows, are found. Since V is a large set, it is reasonable to assume that n linearly independent rows can be found among rows corresponding to the indices in V.

Warm-start procedure after the construction of an initial basis follows with the selection of s constraints to be appended to the LP. This part is similar to the row generation procedure. Calculations for the comparison of constraints is

similar to the row generation selection calculations. The difference between this part of the warm-start phase and the row generation phase is that there is no optimization in warm-start phase. Each row generation is followed by search for a feasible solution of the new LP after the row generation. Once a feasible solution is found, an optimal feasible solution of the new LP is searched for. The next row to be generated is selected using the optimal solution $\hat{x}$ of the new LP. In the warm-start phase, a feasible solution of the new LP is sufficient to proceed with the selection of the next constraint. Unlike the row generation phase, a search for an optimal solution of the new LP is not conducted. Thus, $\bar{x}$, a feasible solution of the new LP, is used in the calculations for constraint selection, instead of $\hat{x}$, an optimal solution.

Following the discussion of approximation techniques and warm-start strategies in the previous sections, this chapter ends with the presentation of the row generation algorithm devised in this study.

4.8 The Algorithm

After formation of an initial LP

$$
\begin{array}{lllll}
 & \text{minimize} & c^T x & & \\
(D_0) & \text{subject to} & A_{B_0\bullet}\, x & \geq & b_{B_0} \\
 & & A_{N_0\bullet}\, x & \geq & b_{N_0}
\end{array}
$$

and solving (D_0) for an initial optimal basic feasible solution, at each step k of row generation, the matrix

$$
\begin{bmatrix} A_{B_k\bullet} \\ A_{N_k\bullet} \end{bmatrix}
$$

contains rows of A corresponding to constraints included in the LP solution. V is the set of indices for the remaining rows of A that are not yet considered in the LP solution. The row generation step $k+1$ consists of picking and removing a row index i_{k+1} from V to be added to $B_k \cup N_k$. After the LP is updated accordingly, the optimal solution of the new system with i_{k+1} in the basis, is found by simplex iterations. This continues until one of the conditions for termination

is encountered: optimal solution of (D_{k+1}) is feasible to (D) under the approximation scheme, a direction of unboundedness of (D) is encountered or (D_{k+1}) is infeasible. This procedure is defined in Algorithm 5.

Steps for the formation of the initial LP (D_0) are covered under the warm-start part of Algorithm 5. As mentioned in Section 4.7, these steps are based on the input V and s, the former a permutation of I representing row selection priorities, and the latter is the number of constraints to be added to the LP after the construction of an initial basis. The LP problems formed and solved in these steps have super-scripts:

$$\begin{array}{rlrll} & \text{minimize} & c^T x & & \\ \left(D_0^k\right) & \text{subject to} & A_{B\bullet}x & \geq & b_B \\ & & A_{i_l\bullet}x & \geq & b_{i_l} \quad l \in \{1,\ldots,k\} \end{array}$$

where B is the initial basis, and i_l, $l \in \{1,\ldots,k\}$ are the constraints appended to the LP according to the warm-start strategy until the initial LP (D_0) is constructed.

Due to the existence of an initial basis, a feasible solution of (D_0^0) is known initially. At each initialization step where a new constraint is added to the LP, a feasible basis of the new system is obtained via simplex iterations, if one exists. Otherwise, it is inferred that (D) is an infeasible problem. Warm-start ends if s constraints are added to the LP after the basis formation, or the feasible basis of $\left(D_0^k\right)$, $0 \leq k < s$ violates no constraints in V. The final LP (D_0^s) or $\left(D_0^k\right)$, becomes the initial LP (D_0) for row generation.

In contrast to warm-start, in the row generation part, a basis that is both feasible and optimal to the new LP is required after appending a violated constraint. After the k^{th} row generation, first a feasible basis of (D_k) is sought, and is found, unless the row generation leads to the discovery that (D) is infeasible. Then, an optimal solution of (D_k) is searched by dual simplex iterations. It is possible that an improving direction of (D_k) is encountered during this process. In this case, the solution process continues by the generation of row $\bar{j} \in V$ if constraint $\bar{j}$ bounds the improving direction of (D_k). If no such constraint exists in V, then the algorithm terminates reporting that (D) is unbounded. Unless an

improving direction is encountered, the optimal basis of (D_k) is found. If this basis is approximately feasible to (D) under the preferred approximation scheme, the basis is reported as the optimal basis of (D). Otherwise, the solution procedure continues with the following row generation step by the selection of the violated constraint with highest score, as discussed in Chapter 3. This solution procedure is presented as a pseudo-code in Algorithm 5.

Algorithm 5 The Row Generation Algorithm

Input: A, b, c, V, s
Initialization: Warm-start
$B \leftarrow$ first n linearly independent constraints from V
$V \leftarrow V \setminus B$, $N \leftarrow \emptyset$, $k \leftarrow 0$
while $k < s$ **do**
  if $A_{B\bullet}^{-1} b_B$ approximately feasible to (D) [$Algorithm 4$] **then**
    Go to **(!)**
  else
    $\bar{j} \leftarrow \hat{v}$ [Algorithm 3 (input: B,N,V,ϕ,γ)]
    $k \leftarrow k+1$, $N \leftarrow N \cup \{\bar{j}\}$, $V \leftarrow V \setminus \{\bar{j}\}$
    if $\left(D_0^k\right)$ remains feasible after addition of constraint $\bar{j}$ **then**
      Find $\bar{B}$ a feasible basis of $\left(D_0^k\right)$
      $N \leftarrow N \cup \left(B \setminus \bar{B}\right)$, $B \leftarrow \bar{B}$
    else
      Report (D) to be an infeasible problem.
(!)$(D_0) \leftarrow \left(D_0^k\right)$, $k \leftarrow 0$
Row Generation
while *true* **do**
  if (D_k) is a bounded problem **then**
    Find $\bar{B}$ an optimal basis of (D_k)
    $N \leftarrow N \cup \left(B \setminus \bar{B}\right)$, $B \leftarrow \bar{B}$
  else
    $\exists d_u$, a direction of (D_k) improving objective
    if $\exists \bar{j} \in V$ s.t. $A_{\bar{j}\bullet} d_u < 0$ **then**
      Go to **(!!!)**
    else
      Report (D) to be an unbounded problem.
  if $A_{B\bullet}^{-1} b_B$ approximately feasible to (D) [Algorithm 4] **then**
    Report B as the optimal basis and $A_{B\bullet}^{-1} b_B$ as the optimal solution for (D).
  else
    $\bar{j} \leftarrow \hat{v}$ [Algorithm 3 (input: B,N,V,ϕ,γ)]
  (!!!)$V = V \setminus \{\bar{j}\}$, $N = N \cup \{\bar{j}\}$, $k = k+1$
  if (D_k) remains feasible after addition of constraint $\bar{j}$ **then**
    Find $\bar{B}$ a feasible basis of (D_k)
    $N \leftarrow N \cup \left(B \setminus \bar{B}\right)$, $B \leftarrow \bar{B}$
  else
    Report (D) to be an infeasible problem.

Chapter 5

Computational Results

In this chapter, computational test results of the implementation of Algorithm 5 are presented. Details are discussed about the attributes of the algorithm that are tested, and how the tests are conducted.

An implementation of the row generation algorithm enhanced with a warm-start strategy as discussed in Section 4.7, and the λ relaxation scheme is used in the computational tests. The row generation algorithm follows the largest reduced cost pivoting rule during the simplex iterations. The same rule is used in iterations that are made to find a feasible solution, yet the objective gradient for these iterations is not $-c$, but $A_{\bar{j}\bullet}$, the normal vector of the last constraint that is appended to the LP.

For comparing computation times, two well known variants of the primal simplex method are chosen: primal simplex with Dantzig type pivot selection and primal simplex with steepest edge pivot selection. Again, the difference is due to pivot selection, one being based on largest reduced cost pivoting [86] and the other being the steepest edge pivoting technique presented in [31].

All of the implementations keep and iterate the basis inverse in the elimination form of inverse (EFI). An LU factorization based implementation of storage and update of the basis is used in all algorithms. This basis storage and update

technique due to Bartels and Golub is preferred for its stability and efficiency [7], [8], [9], [70].

Clustering is done by the furthest neighbor approximation algorithm for k-center clustering. This is a 2-approximation algorithm for k-center clustering, namely, it guarantees a solution with maximum cluster radius at most twice that of the optimal k-center clustering. It is an intuitive approach to the k-center clustering problem, achieving efficiency and solution quality in the same time. The approximation coefficient 2 might be misleading at this point, it is thus worth noting that k-center clustering problem is shown to be inapproximable below a factor of 2 [33], [37].

Warm-start of dual row generation algorithms is done by assigning high priorities to the constraints in the parallel group and orthogonal group. Each group and cluster is ordered in itself according to descending right-hand-sides, and in priority one orthogonal group constraint follows one parallel group constraint. The remaining constraints have lower priorities, assigned according to the angles of their normal vectors with c and their right hand side values, as stated in Section 4.7.

For all algorithms, the distinction between a non-zero and zero is limited from below by 10^{-9} for assuring stability of division operations. For the dual row generation variants, the λ approximation scheme is used as discussed in Section 4.3, and the tolerance level ϵ for approximate feasibility is set to 0.05.

Coding is done on GNU Octave 3.2, an open source MATLAB clone. Computational tests are carried out using a PC with Intel Core 2 Duo 2.53 GHz P8700 CPU and 4 GB DDR2 800MHz RAM on a Linux Ubuntu 10.04 operating system. The average run time and iteration count results are obtained by running each algorithm on 30 instances of any given problem size.

For the primal simplex variants, iteration counts are given separately for phase one and phase two, and computation times are given both separately and as a total for phase one and phase two.

For the row generation algorithms, computation time is presented under three titles: preparation, warm-start and row generation. The preparation part, which does not appear in Algorithm 5, covers normalization of vectors, forming of orthogonal group clusters and the parallel group vectors. The computation time for this part is considered separately especially for keeping an account of clustering times. Warm-start and row generation times correspond to the parts with the same name under Algorithm 5.

Iteration counts are given separately for warm-start and row generation phases. There are four subtitles for iteration counts of the row generation phase: number of rows generated, number of simplex iterations to find a feasible solution after a row generation, the number of simplex iterations for improving the objective value, and the number of simplex iterations following the row generation after a direction of unboundedness is encountered. Iteration counts are presented in a similar format for the warm-start phase, but since optimization is not attempted in this phase, there are no simplex iterations improving the objective value, and unboundedness is not encountered.

In parallel to the motivation for the row generation algorithm, data with number of constraints, m, significantly larger than the dimension, n, is generated (from the primal perspective, we say that number of columns significantly larger than number of rows). Data is generated randomly, by uniform distribution applied on an appropriate range for each of b and c. Data generation is based on uniform distribution for A, an adjustment on top of random generation is needed to obtain LP problem instances that are both feasible and bounded. Details about random data generation are discussed in the following section.

5.1 Generation of random data

LP problem instances that satisfy both dual and primal feasibility are generated with random data. When both dual and primal problems are feasible, they are both bounded at the same time, and therefore have optimal solutions. Noting that

number of dual rows is significantly larger than the number of dual columns ($m \ll n$), to achieve primal and dual feasibility together, data is generated according to the following rules:

- Entries of the $m \times n$ coefficient matrix A are generated independently according to uniform distribution in the interval $[-1, 1]$.
- A primal basic feasible solution $\hat{w}$ is constructed by forming a vector of length m whose entries are independent, identically and uniformly distributed in the interval $(0, 1]$, initially. Then, randomly selected $m - n$ entries of this vector are set to 0.
- The vector c takes the value $A^T\hat{w}$, so that the primal problem is feasible: $\hat{w} \in \{w \in \mathbb{R}^m : A^T w = c\}$.
- Two vectors, s_1 and s_2 in $\mathbb{R}^m$, whose entries are independent, identically and uniformly distributed in the interval $(0, 1]$ are generated. Another vector $u \in \mathbb{R}^m$ is generated with all entries equal to 1 except n randomly selected entries that are set to 0. Finally, a vector $s \in \mathbb{R}^m$ is formed, whose entries are equal to the product of the three entries corresponding in s_1, s_2, u.
- Using s a feasible dual basis is obtained by setting $b = A\hat{x} - s$. Elements of $\hat{x}$ are generated uniformly in $[-1, 1]$. $\hat{x}$ satisfies n of the inequalities of the system $Ax \geq b$ by equality, and the others are satisfied with excess.

By these rules a primal and dual feasible LP problem instance is generated.

The results of computational tests with the data generated according to these rules are presented in Tables 5.5-5.4. The next section follows with the discussion on test results.

5.2 Interpreting the results

The first thing obvious from the test results is that it is very difficult to match primal steepest edge simplex method in efficiency. This is true for both dual

row generation algorithms and the primal simplex method. Primal steepest edge method had significantly lower computation times than all other algorithms. Partially, there are implementation based reasons behind this, since the relatively old primal and primal steepest edge simplex algorithms had significant improvements through time. Especially, efficient update techniques for the edge norm squares and reduced cost vector are developed for the primal steepest edge simplex algorithm [31], which are important for the implementation to attain its potential for high efficiency.

From the preparation times given in Table 5.5, it can be concluded that clustering takes significant time for small data, since it constitutes the vast part of the preparation. Preparation times are larger than warm-start times for all data sizes, and both row generation variants. However, the interesting property about preparation times is that there is a very slow growth. Appropriate selection of α_π and α_ω makes possible keeping the sizes of sets Π and Ω constant while problem size increases. This assures that there is a very small growth in preparation times as the problem size becomes larger. An interpretation of this is that the computational cost incurred for clustering pays off better as the number of constraints increases.

From the number of row generations given in Table 5.5 for the row generation algorithm, it is inferred that the growth in this number is slow relative to the increase in the number of constraints for a fixed dimension. That the number of row generations is not too sensitive to the increase in number of total rows is a favorable property of the row generation algorithm, since it is desired that an LP with a high number of rows relative to columns is solved with the generation of minimum number of rows.

Looking at Tables 5.2 and 5.3 for results on primal simplex variants, it is seen that phase two demands twice to three times iterations and computational time, compared to that demanded by phase one. Number of iterations is reduced by steepest edge pivoting, as expected. Interestingly, primal steepest edge simplex method achieves about 10 times faster performance than primal simplex method by a nearly 30% reduction in number of iterations.

Table 5.1: Row Generation Results

Data Size (n x m)	**Warm-Start**		**Row Generation**				**Prep. Time**	**Warm-start Time**	**Row Gener. Time**	**Total Time**
	Rows Gener.	**# of Iterations for Feas.**	**Rows Gener.**	**# of Iterations for**						
				Feas.	**Impr.**	**Unbd.**				
30 x 700	94.8	255.9	0	0	26.5	0	0.065	0.208	0.036	0.289
30 x 7000	205.3	573.7	0	0	34.6	0	0.765	0.82	0.023	1.608
30 x 30000	302.7	921.9	0	0	47.5	0	4.88	3.175	0.04	8.099
30 x 100000	365.7	1104.6	0	0	58.13	0	35.31	10.84	0.068	46.21
50 x 1000	120.6	488.4	0	0	56.53	0	0.161	0.567	0.054	0.781
50 x 15000	234.8	1174	0	0	79.5	0	1.963	2.77	0.09	4.826
50 x 100000	333.2	1793.6	0	0	103.2	0	35.2	22.6	0.19	57.98
50 x 250000	768	3189.2	0	0	115.8	0	203.46	134.36	0.445	338.3

Table 5.2: Primal Results

Data Size (n x m)	# of Iterations		Time		Total Time
	Phase 1	Phase 2	Phase 1	Phase 2	
30 x 700	42.3	111.4	0.0695	0.174	0.244
30 x 7000	40.6	167.5	0.631	2.38	3.01
30 x 30000	44.6	203.8	3.49	16.12	19.6
30 x 100000	39	215.8	10.65	50.04	60.69
50 x 1000	80.6	222.5	0.385	1.025	1.41
50 x 15000	77.8	371.6	6.62	31.77	38.39
50 x 100000	72.8	440.8	42.9	263.2	306.2
50 x 250000	72.5	521	106.8	776.6	883.4

Table 5.3: Primal Steepest Results

Data Size (n x m)	# of Iterations		Time		Total Time
	Phase 1	Phase 2	Phase 1	Phase 2	
30 x 700	35.4	78.6	0.0362	0.0798	0.116
30 x 7000	34.4	117	0.167	0.553	0.720
30 x 30000	35.27	133	0.73	2.594	3.324
30 x 100000	35.8	134.2	2.48	8.63	11.11
50 x 1000	65.13	149.53	0.108	0.245	0.353
50 x 15000	62.8	237.2	0.944	3.396	4.34
50 x 100000	60	281	7.44	33.7	41.1
50 x 250000	58.8	288.8	18	86.9	104.9

Table 5.4: Total Time Comparisons

Data Size (n x m)	Total Time		
	Primal	Primal Steepest	Row Generation
30 x 700	0.244	0.116	0.289
30 x 7000	3.01	0.720	1.608
30 x 30000	19.6	3.324	8.099
30 x 100000	60.69	11.11	46.21
50 x 1000	1.41	0.353	0.781
50 x 15000	38.39	4.34	4.826
50 x 100000	306.2	41.1	57.98
50 x 250000	883.4	104.9	338.3

Table 5.5: Ratio of Rows Generated to Total # of Constraints (%)

	n			
m	700	7000	30000	100000
30	17.8	3.4	1.1	0.4

	n			
m	1000	15000	100000	250000
50	17.1	1.9	0.38	0.32

Chapter 6

Conclusion and Future Research

The focus point of this study was incorporating the minimum number of constraints in the solution of LP problems that have a large number of constraints relative to the problem dimension. The computational test results indicate that the algorithm developed achieves this aim, since less than 18 percent of constraints were sufficient to obtain an optimal solution of the input LP. This percentage reduces for problems with larger number of constraints, and attains a value as small as 0.3 percent. A more favorable result is that, actually a very small growth occurs in the number of row generations as the row number of data increases.

The optimal solution is reached by working on LP problems with a small percentage of rows. Still, there are concerns about the efficiency of the dual row generation algorithm. Although tests indicate slower growth in computation times than the primal steepest edge simplex method, the superior performance of primal steepest edge indicates that the row generation algorithm has several points to be improved.

For an efficient implementation of an LP algorithm based on traversing basic solutions, efficient vector and matrix update is very important. Storage and update of the basis inverse, which is common to all such algorithms, received

significant academical effort until efficient and stable implementations were developed. Similarly, efficient update of the edge norm squares has great importance for the steepest edge simplex algorithms. Where applicable, such efficient implementations are incorporated into the implementation of the row generation algorithm. Yet, many computation and update operations of the row generation algorithm require more efficient implementations. This remains as an issue for further research.

Another reason for slower performance of the current implementation is the difficulty of determining ideal settings for the intrinsic parameters of the dual row generation algorithm. What is the maximum angle that a constraint normal vector should be at with the objective vector, so that it is regarded as being almost parallel to the objective vector? What is the range of angles for including a constraint in the orthogonal group? In how many clusters should the orthogonal group constraints be arranged? Again, seeking the answers to these questions to achieve greater performance remains to future research.

It is not possible to say that one algorithm is a faster general LP solver than another algorithm. For instance, interior point methods are considered to be the most efficient large-scale LP solvers. Yet, there are problem types that the primal simplex method performs significantly better than interior point methods and dual simplex method. Also, there are problem types that the dual simplex method performs significantly better than other LP solution techniques. If the weak points mostly related to implementation can be improved, the growths in the computation times reveal that the dual row generation algorithm is a good approach for the solution of LP problems with a very large number of rows compared to the number of columns.

Bibliography

[1] P. Abel, On the Choice of the Pivot Columns of the Simplex-Method: Gradient Criteria, *Computing* 38, 13-21, 1987.

[2] I. Adler, N. Karmarkar, M.G.C. Resende and G. Veiga, An Implementation of Karmarkar's Algorithm for Linear Programming, *Mathematical Programming* 44, 297-335, 1989.

[3] E.D. Andersen, J. Gondzio, C. Mészáros and X. Xu, Implementation of Interior Point Methods for Large Scale Linear Programming, Technical Report 1996.3, Logilab, University of Geneva, Switzerland, 1996.

[4] H. Arsham, A Hybrid Gradient and Feasible Direction Pivotal Solution Algorithm for General Linear Programs, *Applied Mathematics and Computation* 188, 596-611, 2007.

[5] D. Avis and K. Fukuda, A Pivoting Algorithm for Convex Hulls and Vertex Enumeration of Arrangements and Polyhedra, *Discrete and Computational Geometry* 8(1), 295-313, 1992.

[6] J. Barle and J. Grad, On the Use of Dense Matrix Techniques within Sparse Simplex, *Annals of Operations Research* 43, 3-14, 1993.

[7] R. H. Bartels, A Stabilization of the Simplex Method, *Numer. Math.* 16, 414-434, 1971.

[8] R. H. Bartels and G. H. Golub, Algorithm 350: Simplex Method Procedure Employing LU Decomposition, *Communications of the ACM* 12(5), 275-281, 1969.

[9] R. H. Bartels and G. H. Golub, The Simplex Method of Linear Programming Using LU Decomposition, *Communications of the ACM* 12(5), 266-268, 1969.

[10] K. Belling-Seib, An Improved General Phase-I Method in Linear Programming, *European Journal of Operations Research* 36, 101-106, 1988.

[11] J. Bisschop and A. Meeraus, Matrix Augmentation and Partitioning in the Updating of the Basis Inverse, *Mathematical Programming* 13, 241-254, 1977.

[12] R. E. Bixby, Implementing The Simplex Method: The Initial Basis, Department of Mathematical Sciences, Rice University, Houston, Texas,1991.

[13] R. E. Bixby, Progress in Linear Programming, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1993.

[14] R. E. Bixby, Solving Real-World Linear Programs: A Decade and More of Progress, *Operations Research* 50(1), 3-15, 2002.

[15] R.E. Bixby, J.W. Gregory, I.J. Lustig, R.E. Marsten, D.F. Shanno, Very Large-Scale Linear Programming: A Case Study in Combining Interior Point and Simplex Methods, *Operations Research* 40(5), 885-897, 1992.

[16] R. G. Bland, New Finite Pivoting Rules for The Simplex Method, *Mathematics of Operations Research* 2(2), 103-107, 1977.

[17] R. K. Brayton, F. G. Gustavson and R. A. Willoughby, Some Results on Sparse Matrices, *Mathematics of Computation* 24(112), 937-954, 1970.

[18] H.-D. Chen, P. M. Pardalos and M. A. Saunders, The Simplex Algorithm with a New Primal and Dual Pivot Rule, *Operations Research Letters* 16(3), 121-127, 1994.

[19] G.B. Dantzig, Linear Programming, *Operations Research* 50(1), 50th Anniversary Issue, 42-47, 2002.

[20] S. Eldersveld and T. S. Munson, A Block-LU Update For Large-Scale Linear Programming, *SIAM J. Matrix Anal. Appl.* 13(1), 191-201, 1992.

[21] A. M. Erisman, R. G. Grimes, J. G. Lewis, W. G. Poole, Jr., A Structurally Stable Modification of Hellerman-Rarick's P4 Algorithm for Reordering Unsymmetric Sparse Matrices, *SIAM Journal on Numerical Analysis* 22(2), 369-385, 1985.

[22] R. Fletcher and S.P.J. Matthews, Stable Modification of Explicit LU Factors for Simplex Updates, *Mathematical Programming* 30, 267-284, 1984.

[23] J.J.H. Forrest and J.A. Tomlin, Updated Triangular Factors of the Basis to Maintain Sparsity in the Product Form Simplex Method, *Mathematical Programming* 2, 263-278, 1972.

[24] J.J.H. Forrest and J.A. Tomlin, Vector Processing in Simplex and Interior Methods for Linear Programming, *Annals of Operations Research* 22, 71-100, 1990.

[25] J. J. Forrest and D. Goldfarb, Steepest-edge Simplex Algorithms for Linear Programming, *Mathematical Programming* 57, 341-374, 1992.

[26] S. I. Gass and S. Vinjamuri, Cycling in Linear Programming Problems, *Computers & Operations Research* 31, 303-311, 2004.

[27] A. M. Geoffrion, Elements of Large-Scale Mathematical Programming: Part I: Concepts, *Management Science* 16(11), 652-675, 1970.

[28] P. E. Gill and W. Murray, A Numerically Stable Form of the Simplex Algorithm, *Linear Algebra and Its Applications* 7, 99-138, 1973.

[29] P. E. Gill, W. Murray, M. A. Saunders and M. H. Wright, Maintaining LU Factors of a General Sparse Matrix, *Linear Algebra and its Applications* 88-89, 239-270, 1987.

[30] P. E. Gill, G. H. Golub, W. Murray and M. A. Saunders, Methods for Modifying Matrix Factorizations, *Mathematics of Computation* 28(126), 505-535, 1974.

[31] D. Goldfarb and J.K Reid, A Practicable Steepest-Edge Simplex Algorithm, *Mathematical Programming* 12, 361-377, 1977.

[32] D. Goldfarb and W.Y. Sit, Worst Case Behavior of the Steepest Edge Simplex Method, *Discrete Applied Mathematics* 1, 277-285, 1979.

[33] T. F. Gonzalez, Clustering to Minimize the Maximum Intercluster Distance, *Theoretical Computer Science* 38, 293-306, 1985.

[34] W.W. Hager, Updating the Inverse of a Matrix, *SIAM Review* 31(2), 221-239, 1989.

[35] P.M.J. Harris, Pivot Selection Methods of the Devex LP Code, *Mathematical Programming* 5, 1-28, 1973.

[36] E. Hellerman and D. Rarick, Reinversion with the Preassigned Pivot Procedure, *Mathematical Programming* 1, 195-216, 1971.

[37] W.L. Hsu, G.L. Nemhauser, Easy and Hard Bottleneck Location Problems, *Discrete Applied Mathematics* 1(3), 209-215, 1979.

[38] T. Illés and T. Terlaky, Pivot versus Interior Point Methods: Pros and Cons, *European Journal of Operations Research* 140, 170-190, 2002.

[39] B. Jaumard, C. Meyer and T. Vovor, How to Combine a Column and Row Generation Method with a Column or Row Elimination Procedure - Application to a Channel Assignment Problem, Mobile Networks and Computing, DIMACS Workshop, 1999.

[40] Y. Jin and B. Kalantari, A Procedure of Chvátal for Testing Feasibility in Linear Programming and Matrix Scaling, *Linear Algebra and its Applications* 416, 795-798, 2006.

[41] J. E. Kalan, Aspects of Large-Scale In-Core Linear-Programming, *ACM Annual Conference, Proceedings of the 1971, 26th Annual Conference*, 304-313, 1971.

[42] N. Karmarkar, A New Polynomial-Time Algorithm for Linear Programming, *Combinatorica* 4(4), 373-395, 1984.

[43] L. Khachiyan, A Polynomial Algorithm in Linear Programming, *Doklady Akademii Nauk SSSR* 244:S, 1093-1096, 1979.

[44] L. Khachiyan, On the Complexity of Approximating Extremal Determinants in Matrices, *Journal of Complexity* 11, 138-153, 1995.

[45] M. Kojima, N. Megiddo and S. Mizuno, A Primal-Dual Infeasible-Interior-Point Algorithm for Linear Programming, *Mathematical Programming* 61(1-3), 263-280, 1993.

[46] T.D. Lin and R.S.H. Mah, Hierarchical Partition - A New Optimal Pivoting Algorithm, *Mathematical Programming* 12, 260-278, 1977.

[47] I. J. Lustig, An Analysis of an Available Set of Linear Programming Test Problems, *Computers & Operations Research* 16(2), 173-184, 1989.

[48] T. L. Magnanti and J. B. Orlin, Parametric Linear Programming and Anti-Cycling Pivoting Rules, *Mathematical Programming* 41, 317-325, 1988.

[49] I. Maros, A General Phase-I Method in Linear Programming, *European Journal of Operations Research* 23, 64-77, 1986.

[50] I. Maros, A Generalized Dual Phase-2 Simplex Algorithm, *European Journal of Operations Research* 149, 1-16, 2003.

[51] I. Maros, A Practical Anti-Degeneracy Row Selection Technique in Network Linear Programming, *Annals of Operations Research* 47, 431-442, 1993.

[52] I. Maros and M. H. Khaliq, Advances in Design and Implementation of Optimization Software, *European Journal of Operations Research* 140, 322-337, 2002.

[53] I. Maros, An Enhanced Piecewise Linear Dual Phase-1 Algorithm for the Simplex Method, Departmental Technical Report, Department of Computing, Imperial College, London, 2002.

[54] I. Maros and G. Mitra, Simplex Algorithms, Advances in Linear and Integer Programming, J.E. Beasley, Oxford Science Publications, 1996.

[55] R. D. McBride, A Bump Triangular Dynamic Factorization Algorithm for the Simplex Method, *Mathematical Programming* 18, 49-61, 1980.

[56] L. F. McGinnis, Implementation of a Primal-Dual Algorithm for the Assignment Problem, *Operations Research* 31(2), 277-291, 1983.

[57] N. Megiddo, S. Mizuno and T. Tsuchiya, A Modified Layered-Step Interior-Point Algorithm for Linear Programming, *Mathematical Programming* 82, 339-355, 1998.

[58] B. A. Murtagh and M. A. Saunders, Large-Scale Linearly Constrained Optimization, *Mathematical Programming* 14, 41-72, 1978.

[59] K. G. Murty and Y. Fathi, A Feasible Direction Method for Linear Programming, *Operations Research Letters* 3(3), 121-127, 1984.

[60] P.-Q. Pan, A Basis-Deficiency-Allowing Variation of the Simplex Method for Linear Programming, *Computers & Mathematics with Applications* 36(3), 33-53, 1998.

[61] P.-Q. Pan, A Dual Projective Simplex Method for Linear Programming, *Computers & Mathematics with Applications* 35(6), 119-135, 1998.

[62] P.-Q. Pan, A Largest-Distance Pivot Rule for the Simplex Algorithm, *European Journal of Operations Research* 187, 393-402, 2008.

[63] P.-Q. Pan, The Most-Obtuse-Angle Row Pivot Rule for Achieving Dual Feasibility: A Computational Study, *European Journal of Operations Research* 101, 164-176, 1997.

[64] P.-Q. Pan, Practical Finite Pivoting Rules for the Simplex Method, *OR Spektrum* 12, 219-225, 1990.

[65] K. Paparrizos, N. Samaras and G. Stephanides, A New Efficient Primal Dual Simplex Algorithm, *Computers & Operations Research* 30, 1383-1399, 2003.

[66] K. Paparrizos, An Exterior Point Simplex Algorithm for (General) Linear Programming Problems, *Annals of Operations Research* 47, 497-508, 1993.

[67] K. Paparrizos, N. Samaras and G. Stephanides, An Efficient Simplex Type Algorithm for Sparse and Dense Linear Programs, *European Journal of Operations Research* 148, 323-334, 2003.

[68] S. H. Paskov, Termination Criteria for Linear Problems, *Journal of Complexity* 11, 105-137, 1995.

[69] A. F. Perold, A Degeneracy Exploiting LU Factorization for the Simplex Method, *Mathematical Programming* 19, 239-254, 1980.

[70] J. K. Reid, A Sparsity-Exploiting Variant of the Bartels-Golub Decomposition for Linear Programming Bases, *Mathematical Programming* 24, 55-69, 1982.

[71] M. A. Saunders, The Complexity of LU Updating in the Simplex Method, The Complexity of Computational Problem Solving, R.S. Anderssen and R.P. Brent, University of Queensland Press, 214-230, 1976.

[72] R. Shamir, The Efficiency of the Simplex Method: A Survey, *Management Science* 33(3), 301-334, 1987.

[73] S. W. Sloan, A Steepest Edge Active Set Algorithm for Solving Sparse Linear Programming Problems, *International Journal for Numerical Methods in Engineering* 26, 2671-2685, 1988.

[74] D. A. Spielman and S.-H. Teng, Smoothed Analysis of Algorithms: Why the Simplex Algorithm Usually Takes Polynomial Time, *Journal of the ACM* 51(3), 385-463, 2004.

[75] U. H. Suhl and L. M. Suhl, Computing Sparse LU Factorizations for Large-Scale Linear Programming Bases, *ORSA Journal on Computing* 2(4), 325-336, 1990.

[76] L. M. Suhl and U. H. Suhl, A Fast LU Update for Linear Programming, *Annals of Operations Research* 43, 33-47, 1993.

[77] T. Terlaky and S. Zhang, A Survey on Pivot Rules for Linear Programming, Technical Report 91-99, Delft University of Technology, 1991. ISSN 0922-5641.

[78] M.E. Thomadakis, Implementation and Evaluation of Primal and Dual Simplex Methods with Different Pivot-Selection Techniques in the LPBench Environment, A Research Report, Texas A & M University, College Station, Texas, 1994.

[79] M.E. Thomadakis and J.-C. Liu, An Efficient Steepest-Edge Simplex Algorithm for SIMD Computers, International Conference on Supercomputing, Proceedings of the 10th International Conference on Supercomputing, Philadelphia, Pennsylvania, United States, 286-293, 1996. ISBN: 0-89791-803-7.

[80] M. J. Todd, Polynomial Expected Behavior of a Pivoting Algorithm for Linear Complementarity and Linear Programming Problems, Technical Report No. 595, Cornell University, Ithaca, New York, 1983.

[81] M. J. Todd, L. Tunçel and Y. Ye, Characterizations, Bounds, and Probabilistic Analysis of Two Complexity Measures for Linear Programming Problems, *Mathematical Programming* 90, 59-69, 2001.

[82] M. J. Todd, The Many Facets of Linear Programming, *Mathematical Programming* 91, 417-436, 2002.

[83] L. Tunçel, Approximating the Complexity Measure of Vavasis-Ye Algorithm is NP-hard, *Mathematical Programming* 86(1), 219-223, 1999.

[84] L. Tunçel, On The Condition Numbers for Polyhedra in Karmarkar's Form, *Operations Research Letters* 24, 149-155, 1999.

[85] R. J. Vanderbei, M. S. Meketon and B. A. Freedman, A Modification of Karmarkar's Linear Programming Algorithm, *Algorithmica* 1, 395-407, 1986.

[86] R. J. Vanderbei, Linear Programming: Foundations and Extensions, Department of Operations Research and Financial Engineering, Princeton University , Princeton, New Jersey, 2001.

[87] S. A. Vavasis and Y. Ye, An Accelerated Interior Point Method Whose Running Time Depends on Only on A (Extended Abstract), Annual ACM Symposium on Theory of Computing, Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 512-521, 1994.