COMBINATORIAL OPTIMZATION

BITS PILANI KK BIRLA GOA CAMPUS

YEAR 2016-2017

DEVVRIT (2015A7PS053G)

SARTHAK MOORJANI (2015A7PS104G)

ABSTRACT

Many problems of both practical and theoretical importance concern themselves with the choice of a “best” configuration or set of parameters to achieve some goal. Over the past few decades a hierarchy of such problems has emerged, together with a corresponding collection of techniques for their solution. The topic of combinatorial optimization is of paramount significance in the field of problems asking for an “optimal” solution with linear constraints. In this report, an introduction to what combinatorial optimization problems are given, followed by one of the most important algorithms for solving such problems, known as Simplex algorithm. After that a property of the simplex algorithms, namely Duality, is touched upon in the the third chapter.

Acknowledgement

We would like to express our heartfelt gratitude to our Professor and Project guide DR Bharat M. Deshpande for his invaluable support, encouragement and useful suggestions throughout the project. His continuous guidance has enabled us to successfully carry out my project work.

We are indebted to our family and friends for their unceasing love, support and concern that kept us motivated.

Contents

Optimization Problems

1.1 General Nonlinear programming problem-

A problem in which the solution is x and the given constraints are of the form-

minimize f(x), subject to

$g_i(x) \geq 0 \qquad i = 1,2.....,m$

$h_j(x) = 0 \qquad j=1,2,.....,p$

where f,g and h are general functions of the parameter x. When f is convex, g is concave and h is linear, we call this problem a convex programming problem. This problem has the most convenient property that local optimality satisfies the global optimality.

When f,g and h are all linear, the problem is called the linear programming problem. If any problem reduces to the selection of a solution from among a finite set of possible solutions, then that problem can be called as a combinatorial problem.

Optimization problems

These set of problems have 2 sub categories, one with continuous variables and other with discrete variables. In the continuous problems, the solutions can be a set of real numbers or a function defined on real number. However, the one with the discrete variables are called combinatorial optimization problems and the solution set is an object from a finite or possibly countably infinite set.

Instance of an optimization problem

An instance of an optimization problem is a pair(F,c) where F is any set, the domain of feasible points; c is the cost function which is to be maximized or minimized and a mapping:

$$c: F \rightarrow R$$

The problem is to find an $f \in F$ for which

$$c(f) \leq c(y) \text{ for all } y \in F$$

Such a point f is called the globally optimum solution to the given instance or simply an optimal solution.

Optimization Problem

An optimization problem is a set of instances of an optimization problem. In an instance, we are given the input data and we have to obtain the solution using the given information which is sufficient. An optimization problem is a collection of instances. For example, an instance of a travelling salesman problem has a given instance matrix but generally speaking, the travelling salesman problem is the collection of all instances associated with all possible distance matrices.

Travelling salesman problem

The Travelling Salesman Problem (often called TSP) is a classic algorithmic problem in the field of computer science. It is focused on optimization. In this context better solution often means a solution that is cheaper. The Travelling Salesman Problem describes a salesman who must travel between N cities. The order in which he does so is something he does not care about, as long as he visits each one during his trip, and finishes where he was at first. Each city is connected to other close by cities, or nodes, by airplanes, or by road or railway. Each of those links between the cities has one or more weights (or the cost) attached. The cost describes how "difficult" it is to traverse this edge on the graph, and may be given, for example, by the cost of an airplane ticket or train ticket, or perhaps by the length of the edge, or time required to complete the traversal. The salesman wants to keep both the travel costs, as well as the distance he travels as low as possible.

In an instance of the Travelling Salesman Problem we are given an integer n>0 and the distance between every pair of n cities which he needs to travel in the form of an n×n matrix which is called the distance matrix D. An entry D(i,j) represents the distance from city i to city j. A tour is defined as a closed path that visits every city exactly once. The problem is to find a tour with minimal total length. The problem is basically to find a spanning tree on n vertices that has minimal total length. In an instance of an optimization problem, we choose

F = {all spanning trees (V,E) with V = {1,2. . . . ,n}}

c: (V,E) → $\sum_{i,j \in E} \; d_{i,j}$

1.2 Neighbourhoods

Given an optimization problem with instances (F,c) a neighbourhood is a mapping

$$N: F \rightarrow 2^F$$

defined for each instance.

If $F = R^n$ the set of points within a fixed Euclidian distance provide a natural neighborhood. In combinatorial problems, the choice of N may depend critically on the structure of F.

For example, in the Travelling salesman problem we may define a neighborhood 2-change by

$N_2(f)$ = {g: $g \in F$ and g can be obtained from f as follows: Remove any 2 edges from the tour, and then replace them with 2 edges}

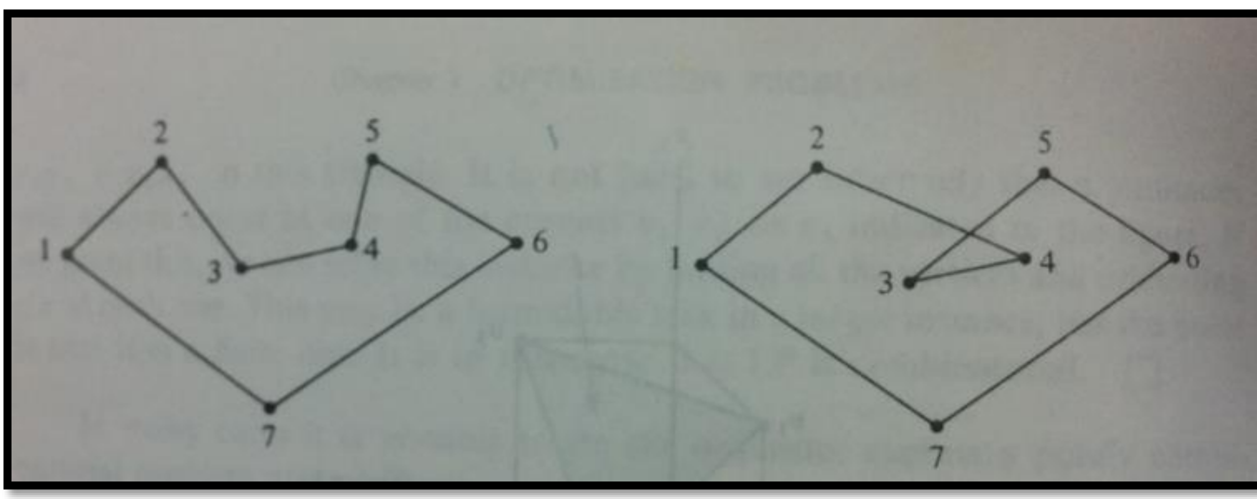


Fig 1.0

Exact Neighborhood

Given an optimization problem with feasible set F and a neighborhood N, if whenever f ∈ F is locally optimal with respect to N, it is also globally optimal, then we say that the neighborhood N is exact.

Local and global optima

Given an instance (F,c) of an optimization problem and a neighbourhood N, a feasible solution f∈F is called locally optimal with respect to N if

c(f) ≤ c(g) for all g∈ N(f)

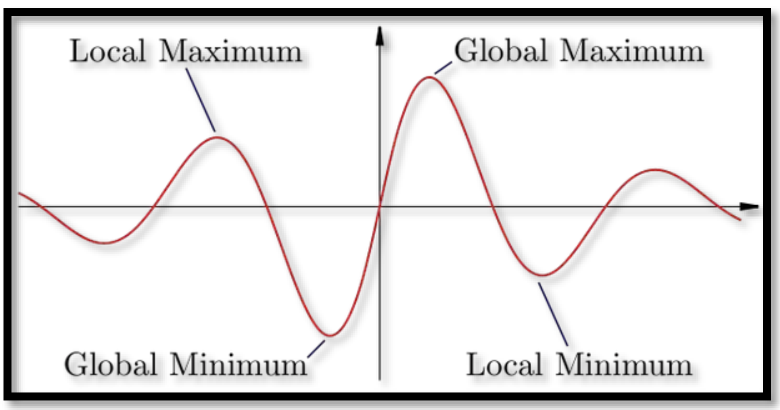


Fig 1.1

1.3 Convex Sets

A subset A of a vector space V is said to be convex if

$z = \lambda x + (1-\lambda)y$ for all vectors $z \in A$, and all scalars $\lambda \in [0,1]$.

Via induction, this can be seen to be equivalent to the requirement that

$\lambda_1 x_1 + \cdots \ldots . + \lambda_n x_n \in A$ for all vectors $x_1, x_2, \ldots . x_n \in A$,

and for all scalars $\lambda_1, \lambda_2, \ldots . \lambda_n \geq 0$ such that

$\sum \lambda_i = 1$

With the above restrictions on the λ_i, an expression of the form $\lambda_1 x_1 + \cdots \ldots . + \lambda_n x_n$ is said to be a convex combination of the vectors $x_1, x_2, \ldots . x_n$.

Few Results:

- A set S $\subset$ R^n is convex if it contains all convex combinations of pairs of points x,y $\in$ S.
- The entire set R^n is convex.
- In R, any interval is convex and any convex set is an interval.

The intersection of any number of convex sets is convex.

If x and y are two points in the intersection of the convex sets, then obviously they are present in each of the sets. This means that every convex combination of them is present in each of the sets, which further means that every convex combination of them is also present in the intersection of the sets.

1.4 Convex Functions

Let S$\subset$$R^n$ be a convex set. Then the function

$$c: S \rightarrow R$$

is convex in S if for any two points x,y $\in$ S,

$$c(\lambda x + (1-\lambda)y) \leq \lambda c(x) + (1-\lambda)c(y), \qquad \lambda \in [0,1]$$

If S = R^n, then c is convex.

It is pertinent to add that any linear function is convex in any convex set S.

Theorem

Let c(x) be a convex function on a convex set S. Then for any real number t, then the set

$$S_t = \{ x : c(x) \leq t;\ x \in S \}$$

is convex.

Proof

Let x and y be 2 points in S_t. Then the convex combination $c(\lambda x + (1-\lambda)y)$ is in S and

$$c(\lambda x + (1-\lambda)y) \leq \lambda c(x) + (1-\lambda)c(y), \qquad \lambda \in [0,1]$$

$$c(\lambda x + (1-\lambda)y) \leq \lambda t + (1-\lambda)t$$

$$c(\lambda x + (1-\lambda)y) \leq t$$

which clearly shows that the convex combination $(\lambda x + (1-\lambda)y)$ is also in S_t.

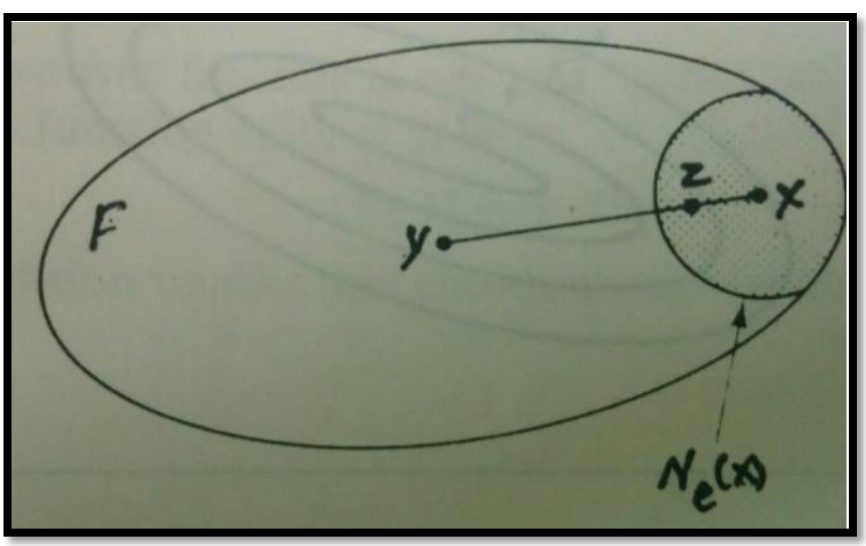
1.5 Convex Programming Problems

Theorem

Consider an instance of an optimization problem (F,c) where $F \subset R_n$ is a convex set and c is a convex function. Then the neighborhood defined by Euclidean distance

$$N_e(x) = \{y : y \in F \text{ and } ||x-y|| \leq \varepsilon$$

is exact for every $\varepsilon \geq 0$.



Let x be a local optimum with respect to N for any fixed $\varepsilon > 0$ and let y ∈ F be any other feasible point, not necessarily in N(x). We can always choose a λ sufficiently close to 1 such that the strict convex combination

$$z = \lambda x + (1 - \lambda)y$$

lies within the neighborhood of N(x). Evaluating the cost function c at this point, we get, by the convexity of c,

$$c(z) = c(\lambda x + (1 - \lambda)y) \leq \lambda c(x) + (1 - \lambda)c(y)$$

Hence,

$$c(y) \geq \frac{c(z) - \lambda c(x)}{1 - \lambda}$$

But since z ∈ N(x),

$$c(z) \geq c(x)$$

So, we get

$$c(y) \geq \frac{c(x) - \lambda c(x)}{1 - \lambda} = c(x)$$

Hence, the neighborhood is exact.

The convex feasible region will always be defined by a set of inequalities involving a concave function. Such problems are called as convex programming problems.

The Simplex Algorithm

2.1 General form of Linear Programming Problem

Given an $m \times n$ integer matrix A with rows a_i. Let M be the set of row indices corresponding to the equality constraints and let M' be those corresponding to inequality constraints. Similarly, let $x \in R^n$ and let N be the column indices corresponding to the constrained variables and N' be those corresponding to the unconstrained variables. Then an instance of a general LP is given by-

$$\min c'x$$

$$a_i x = b_i \qquad i \in M$$

$$a_i x \geq b_i \qquad i \in M'$$

$$x_j \geq 0 \qquad j \in N$$

Canonical form of Linear Programming Problem

A linear programming problem is said to be in canonical form if it has the following constraints

$$\min c'x$$

$$a_i x \geq b_i \qquad i \in M'$$

$$x_j \geq 0 \qquad j \in N$$

Standard form of Linear Programming Problem

A linear programming problem is said to be in standard form if it has the following constraints

$$\min c'x$$

$$a_i x = b_i \qquad i \in M'$$

$$x_j \geq 0 \qquad j \in N$$

It is trivial to prove that the canonical, standard and general forms are all equivalent. An instance in one form can be converted to one in another form by a simple transformation in such a way that the two instances have the same solution. The canonical and standard forms are both already in the general form, so it is sufficient to prove that a general form problem can be put in canonical and standard forms.

2.2 Basic Feasible Solutions

It is always convenient to take a LP in standard form i.e.

$$\min c'x$$

$$a_i x = b_i \qquad i \in M'$$

$$x_j \geq 0 \qquad j \in N$$

It is assumed that the rank of A is m.

A basis of A is a linearly independent collection B = { A_{j1},. . . , A_{jm} }. The basic solution corresponding to B is a vector $x \in R^n$ with

- $x_j = 0 \; for \; A_j \; not \; in \; B$
- $x_{jk} = the \; k^{th} \; component \; of \; B^{-1}b.$

Thus a basic solution x can be found by the following procedure:

1. Choose a set B of linearly independent columns of A.
2. Set all components of x corresponding to columns not in B equal to 0.
3. Solve the m resulting equations to determine the remaining components of x. These are the basic variables.

2.3 The geometry of Linear Programs

Linear and Affine spaces

Consider a vector space R^d. A linear subspace S of R^d is a subset of R^d closed under the vector addition and scalar multiplication. Equivalently, a subspace S of R^d is the set of points in R^d that satisfy a set of homogenous linear equations:

$$S = \{ x \in R^d : a_{j1}x_1 + \ldots\ldots + a_{jd}x_d = 0, where\ j = 1, \ldots\ldots, m \}$$

It is well known that every subspace S has a dimension, dim(S) equal to the maximum number of linearly independent vectors in it. Equivalently, dim(S) = d – rank(a_{ij}). Because for every extra equation that we are adding, dimension of the subspace reduces by one. For example, a plane in 3D say x+y+z = 2 has dimension 2 whereas a given line is 1 dimensional.

An affine subspace A of R^d is a linear subspace S translated by a vector u: A = {u+x}. The dimension of A is that of S. An affine space can also be defined as a subspace A of R^d that is the set of all points satisfying a set of non homogenous equations:

$$A = \{ x \in R^d : a_{j1}x_1 + \ldots\ldots + a_{jd}x_d = b_j, where\ j = 1, \ldots\ldots, m \}$$

The dimension of any subset of R^d is the smallest dimension of any affine subspace which contains it. For example, any line segment has dimension 1; any set of k points where k ≤ d+1, has dimension atmost k-1. Hence, the dimension of the set F defined by the linear programming problem

$$\min c'x$$

$$a_i x = b_i \qquad i \in M'$$

$$x_j \geq 0 \qquad j \in N$$

Is therefore at most d – m.

2.4 Convex Polytopes

An affine subspace R^d of dimension d-1 is called a hyperplane. Alternatively, a hyperplane is a set of points x satisfying

$$a_1x_1 + \ldots\ldots + a_dx_d = b$$

With all a_i's not equal to zero. A hyperplane defined two closed half spaces,

$$a_1x_1 + \ldots\ldots + a_dx_d \geq b$$

$$a_1x_1 + \ldots\ldots + a_dx_d \leq b$$

A halfspace is a convex set. Therefore, the intersection of halfspaces is also convex. The intersection of a finite number of halfspaces, when it is bounded and non empty, is called a convex polytope.

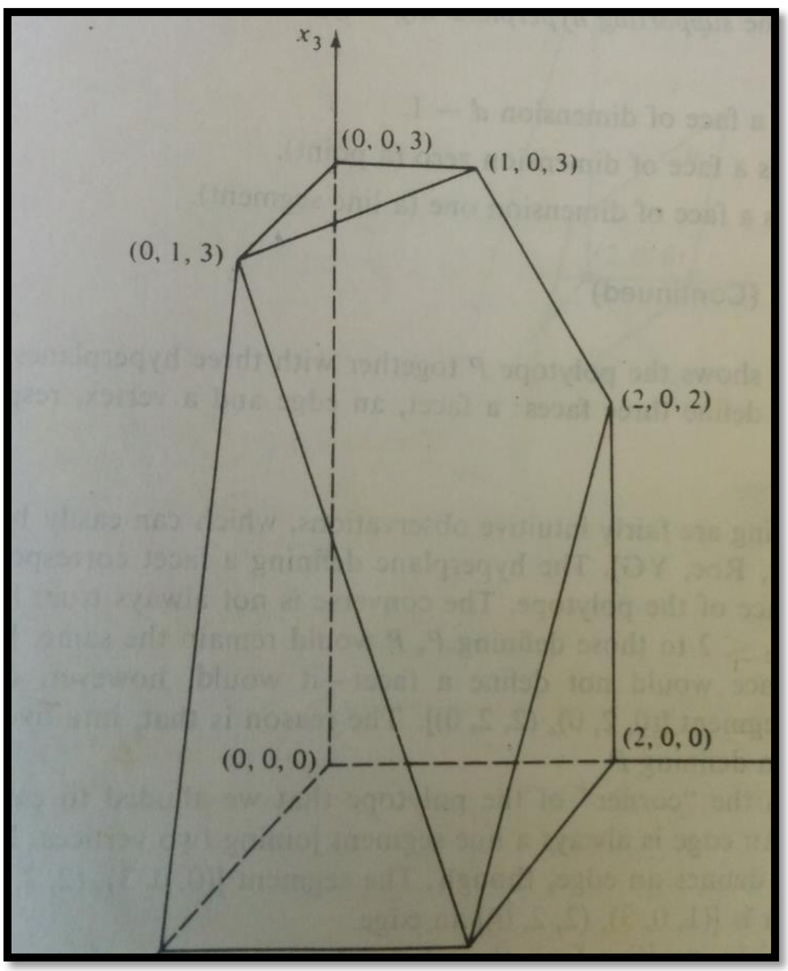


Fig 2.1

2.5 Algebraic aspect of polytope

Let

$$Ax = b$$

$$x_i \geq 0$$

be the set of equations and inequalities of the feasible region F of an LP. Since rank(A)=m, where A is an m×n matrix, we can assume that the equations Ax = b are of the form

$$x_i = b_i - \sum_{j=1}^{n-m} a_{ij}x_j \qquad i = n-m+1, \ldots\ldots., n$$

The above equation is equivalent to the inequalities

$$b_i - \sum_{j=1}^{n-m} a_{ij}x_j \geq 0 \qquad i = n-m+1, \ldots\ldots.., n$$

$$x_j \geq 0 \qquad j = 1, \ldots\ldots, n-m$$

We can clearly see that the above set of equations is the intersection of n half spaces which are bounded. Hence they describe a convex polytope P ⊂ R^{n-m}.

We can also say something about the converse.

Let P be a polytope in R^{n-m}. Then the n half spaces defining P can be expressed by the inequalities

$$h_{i1}x_1 + \ldots\ldots + h_{i,n-m}x_{n-m} + g_i \leq 0 \qquad i = 1, \ldots\ldots.. ., n$$

We can easily assume that the first n-m inequalities are of the form

$$x_i \geq 0 \qquad i = 1, \ldots\ldots\ldots, n-m.$$

Let H be the matrix of the coefficients of the remaining inequalities. We can introduce m slack variables $x_{n-m+1}, \ldots\ldots.., x_n$ to obtain

Ax = b

X ≥ 0

Where the m×n matrix A = [H|I] and $x \in R^n$. Thus, any polytope can be alternatively viewed as the feasible region F of an LP via a simple transformation. Any point

$\dot{x} = (x_1, \ldots\ldots, x_{n-m}) \in P$ can be transformed into $\dot{x} = (x_1, \ldots\ldots, x_{n-m}) \in F$ by simply defining a transformation:

$$x_i = -g_i - \sum_{j=1}^{n-m} h_{ij} x_j \qquad\qquad i = n - m + 1, \ldots\ldots\ldots, n$$

Theorem

Let P be a convex polytope, F = {x: Ax = b, x ≥ 0} be the corresponding feasible set of an LP, and $\dot{x} = (x_1, \ldots\ldots, x_{n-m}) \in P$. Then the following are equivalent:

- The point ẋ is a vertex of P.
- If $\dot{x} = \lambda\dot{x}' + (1-\lambda)\dot{x}'', with\ \dot{x}', \dot{x}'' \in P\ with\ 0 < \lambda < 1, then\ \dot{x}' = \dot{x}'' = \dot{x}$. This means that ẋ cannot be a strict convex combination of the points of P.
- The corresponding vector x in F is a bfs of F.

Consider the matrix A:

$$\begin{matrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 3 & 1 & 0 & 0 & 0 & 1 \end{matrix}$$

And the matrix B:

$$\begin{matrix} 4 \\ 2 \\ 3 \\ 6 \end{matrix}$$

With respect to figure 2-1:

Consider the bases $B = \{ A_1, A_2, A_3, A_6 \}$ and $B' = \{ A_1, A_2, A_4, A_6 \}$.

Both have $B^{-1}b = \{ 2,2,0,0,0,3,0 \}$.

Now, to calculate the vertex corresponding to B, according to our procedure as described before we first set $x_4 = x_5 = x_7 = 0$, which means that the corresponding 3 inequalities must be satisfied by equality determining the vertex (2,2,0) by in the intersection of 3 facets. Now in B', if we replace the constraint $x_1 + x_2 + x_3 \leq 4$ by the constraint $x_3 \geq 0$. But we know that $x_3 = 0$ also happens to pass through the same vertex (2,2,0) and so the solution still remains the same. Thus, a vertex like this must lie on more than n-m = 3 facets. Generalizing, the bfs must have more than n-m zeros. A bfs and the corresponding vertex is called degenerate if it contains more than n-m zeros.

Few More Results

- If two distinct bases correspond to the same bfs x, then x is degenerate.
- There is an optimal bfs in any instance of LP. Furthermore, if more than one bfs's are optimal, then their convex combination is also optimal.

2.6 Moving from bfs to bfs:

Theorem:

Given a bfs x_0 with basic components x_{i0}, I =1, . . , m and basis B=$\{A_{B(i)}: i = 1,..,m\}$, let j be such that $A_j \notin B$. Then the new feasible solution is determined by :

$$\theta_0 = \min i_{such\ that\ xij>0} \frac{x_{i0}}{x_{ij}} = \frac{x_{l0}}{x_{lj}} \qquad (1)$$

$$x'_{i0} = \begin{cases} x_{i0} - \theta_0 x_{ij}\ i \neq l \\ \theta_0\ i = l \end{cases} \qquad (2)$$

is a bfs with basis B' defined by

$$B'(i) = \begin{cases} B(i) i \neq l \\ i\,, i = l \end{cases}$$

When there is a tie in the min operation of equation (1), the new bfs is degenerate.

$Proof$: We need to show that x'_0 with components given by equation (2) is basic, since it is a feasible solution. Thus we must show that the set of columns B'(i) is linearly independent.

Suppose then that for some constants d_i we have:

$$\sum_{i=1}^{m} d_i A_{B'(i)} = d_l A_j + \sum_{i=1,i\neq l}^{m} d_i A_{B(i)} = 0 \qquad (3)$$

Substituting

$$A_j = \sum_{i=1}^{m} x_{ij} A_{B(i)}$$

this becomes

$$\sum_{i=1,i\neq l}^{m} \left(d_i x_{ij} + d_i\right) A_{B(i)} + d_i x_{lj} A_{B(l)} = 0$$

This is a linear combination of the original basis vectors, so all the coefficients must be zero; in particular $d_i x_{lj} = 0$, and hence $d_l = 0$. Equation (3) then implies that the remaining d_i are 0 and hence that the new bases is in fact linearly independent.

We conclude the proof by noting that if a tie occurs in the min operation of equation (1), the corresponding entries in x'_0 is degenerate. ■

2.7 Choosing a profitable Column

Now before we move further, we need to introduce some some terminology that we would be using in further proofs. The cost of a bfs with basis B is

$$z_0 = \sum_{i=1}^{m} x_{i0} c_{B(i)}$$

Now consider the process of bringing Column A_j into the basis: We write A_j In terms of the basis columns as

$$A_j = \sum_{i=1}^{m} x_{ij} A_{B(i)}$$

This can be interpreted as meaning that for every unit of the variable x_j that enters the basis, an amount x_{ij} of each of the variable $x_{B(i)}$ must leave. Thus a unit increase in the variable in the variable x_j results in a net change in the cost equal to

$$c_j - \sum_{i=1}^{m} x_{ij} C_{B(i)}$$

The summation quantity can be assigned it's own symbol, z_j; and we call the difference

$$c'_j = c_j - z_j$$

the $relative\ cost$ of Column j. It is thus profitable to bring the column j into the basis exactly when $c'_j < 0$. Furthermore, when all $c'_j \geq 0$, we are at global optimum.

We write the tableau X as

$$X = B^{-1}A$$

and the vector $z = col\ (z_1, \ldots, z_n)$ from its definition as

$$z' = c'_B X = c'_B B^{-1} A$$

Theorem (Optimality Criteria):

At a bfs x_0, a pivot step in which x_j enters the basis changes the cost by the amount

$$\theta_0 c'_j = \theta_0(c_j - z_j) \qquad (1)$$

If

$$c' = c - z \geq 0$$

then x_0 is optimal.

$Proof$: From the previous Theorem, the new solution is :

$$x'_{i0} = \begin{cases} x_{i0} - \theta_0 x_{ij} & i \neq l \\ \theta_0 & i = l \end{cases}$$

so the new cost is

$$z'_0 = \sum_{i=1, i\neq l}^{m} (x_{i0} - \theta_0 x_{ij}) c_{B(i)} + \theta_0 c_j$$

$$= z_0 + \theta_0(c_j - z_j)$$

which establishes equation (1)

To show that $c' \geq 0$ implies that x_0 is optimal, let y be any feasible vector whatsoever, not necessarily basic. That is,

$$Ay = b$$

and

$$y \geq 0$$

Since $c' = c - z \geq 0$, the cost of y is

$$c'y \geq z'y = c'_B B^{-1} A y = c'_B B^{-1} b = c' x_0$$

which shows that the cost of y can never be less than that of x_0.

■

Now that we know about Simplex, we can write an informal program of the algorithm as:

procedure simplex

begin

op:='no', unbounded:='no'

(**comment**: when either becomes 'yes' the algorithm terminates)

while opt = 'no' **and** unbounded='no' **do**

if $c_j \geq 0$ for all j **then** opt:='yes'

else begin

choose any j such that $c_j < 0$;

if $x_{ij} \leq 0$ for all i **then** unbounded:='yes'

else

find $\theta_0 = \min i_{such\ that\ xij>0} \frac{x_{i0}}{x_{ij}} = \frac{x_{k0}}{x_{kj}}$

and pivot on x_{kj}

end

end

One should note that there's a possibility of this algorithm to run in loops, meaning that the pivots being chosen (columns chosen at each iteration to include in the Basis), might form a cycle. To avoid this, we just set out some stricter rules to choose the column (and the row if minimum θ is obtained at more than one row) so that it doesn't form a cycle. We wrote the rules here as a theorem given by R.G. Bland:

Theorem (Bland's Anticycling algirithm) Suppose in the simplex algorithm we choose the column to enter the basis by

$$j = \min\{c_j - z_j < 0\}$$

(Choose the lowest numbered favorable column), and the row by

$$B(i) = \min\left\{B(i): x_{ij} > 0 \; and \; \frac{x_{i0}}{x_{ij}} \leq \frac{x_{k0}}{x_{kj}} \; for \; every \; k \; with \; x_{kj} > 0\right\}$$

(choose in case of tie the lowest numbered column to leave the basis). Then the algorithm terminates after a finite number of pivots.

□

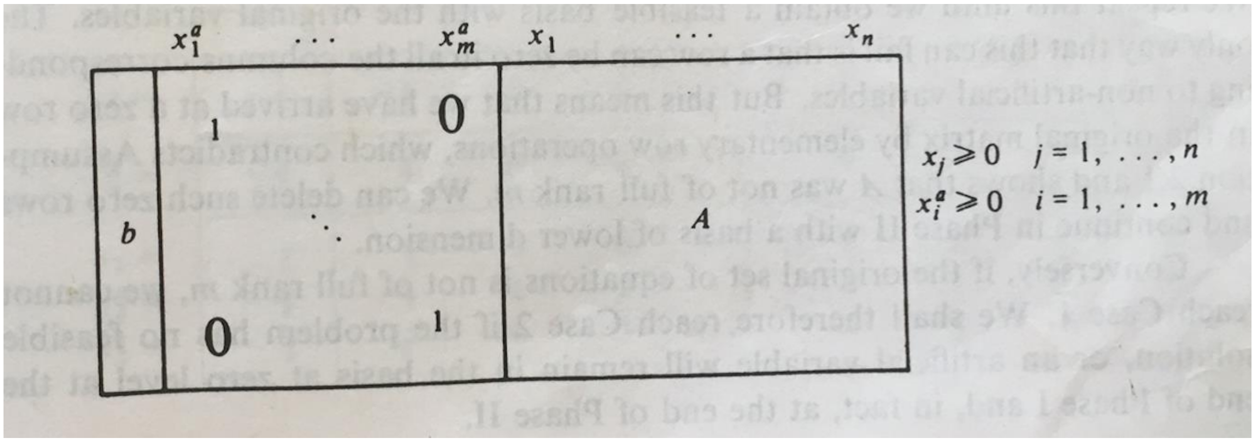
2.8 Beginning the simplex algorithm

We will now see how to begin the simplex algorithm. The main problem in the algorithm that have been explained till now is that we don't have a basis in the beginning to start with. In order to fix this problem, we will introduce the $artificial\ variable$, or $two-phase$, method. In this method, we simply append new, "artificial" variables $x_i^a, i = 1, .., m$ to the left of the tableau as given in the figure. Then we start the simplex algorithms in two phases:

$Phase\ I$: we minimize the cost function:

$$E = \sum_{i=1}^{m} x_i^a$$

Hence after the Phase I, we would have a basis in the original tableau, and the artificial variables will be 0. This would give us a basis to start with in the original tableau. After this, we go for $Phase\ II$, in which we apply the simplex algorithm to the original tableau.



2.9 Geometric Aspect of Pivoting

First we introduce a definition here:

Definition : Two vertices x’ and y’ of a polytope are called to be $adjacent$ if the line segment [x’, y’] is an edge of the polytope. Two distinct bfs’s x and y of an LP $Ax = b, x \geq 0$ are called to be $adjacent$ if there exists bases B_x, B_y such that $B_y = (B_x - \{A_j\}) \cup \{A_k\}$ and $x = B_x^{-1}b, y = B_y^{-1}b$. □

We present a geometric aspect of the relation between adjacent vertices of a polytope and adjacent bfs of corresponding feasible set.

Theorem: Let P be a polytope, $F = \{x: Ax = b, x \geq 0\}$ the corresponding feasible set, and $x' = (x_1, \ldots, x_{n-m}), y' = (y_1, \ldots, y_{n-m})$ be distinct vertices of P. Then the following are equivalent.

1. The segment [x’, y’] is an edge of P.
2. For every $z' \in [x', y'], if\ z' = \lambda z'' + (1-\lambda)z''' with$ $0 < \lambda < 1$ and $z'', z''' \in P$, the $z'', z''' \in [x', y']$.
3. The corresponding vectors x, y of F are adjacent bfs’s.

Duality

3.1 Introduction

Consider an LP in general form:

$$\begin{array}{ll} \min c'x & \\ a_i x = b_i & i \in M \\ a_i x \geq b_i & i \in M' \\ x_j \geq 0 & j \in N \\ x_j \geq 0 \ or \ \leq 0 & j \in N' \end{array}$$

We wish to use the optimality criterion, so we convert this into standard form. For each inequality M', we create a surplus variable $x_i^s, i \in M'$; for each unconstrained variable variable x_j , j ∈ N', create two new nonnegative variables by x_j = x_j^+ - x_j^- , and replace column A_j by two columns A_j and -A_j. This yields the LP

$$\begin{aligned} \min c''x' & \\ A'x' = b & \qquad (3.1) \\ x' \geq 0 & \end{aligned}$$

where

$$A' = \left[A_j, j \in N \,\middle|\, (A_j, -A_j), j \in N' \,|\, 0, i\ i \in M\ or - 1, i \in M'\right]$$

and

$$x' = col\left(x_j, j \in N \,\middle|\, \left(x_j^+, x_j^-\right), j \in N \mid x_i^s, i \in M'\right)$$

$$c'' = \ col\big(c_j, j \in N \,\big|\big(c_j, -c_j\big), j \in N' |\ 0\ \big)$$

We know from the optimality criteria $c'' \geq 0$ and the simplex algorithm that if there is an optimal solution to (3.1), then there exists a basis B' for the LP in Eq 3.1 such that

$$c'' - (c''_B B'^{-1})A' \geq 0$$

Thus, Π' = $c''_B B'^{-1}$ is a feasible solution to the linear constraints

$$\Pi' A' \leq c'' \qquad (3.2)$$

where $\Pi \in R^m$, and m is the number of rows in the original A. These inequalities have three parts, depending on which set of columns of A_j is involved. The first set yields simply

$$\Pi' A_j \leq c_j,\ \ j \in N \qquad (3.3)$$

The next set corresponds to the unconstrained $x_j, j \in N'$, and comes in pairs:

$$\Pi' A_j \leq c_j,\ \ j \in N'\ ]$$

$$-\Pi' A_j \leq -c_j,\ \ j \in N'$$

which is equivalent to

$$\Pi' A_j = c_j,\ \ j \in N' \qquad (3.4)$$

The final set corresponds to the inequalities $i \in M'$:

$$-\pi_i \leq 0, i \in M'$$

which is equivalent to saying:

$$\pi_i \geq 0, i \in M' \qquad (3.5)$$

Equations 3.3, 3.4, and 3.5 define the constraints of a new LP, called the $dual$ of the staring LP; the starting LP is called as the $primal$. The value of Π' = $c_B''B'^{-1}$ is feasible in the dual. If we define the cost function of the dual as max Π'b. then Π' is not only feasible, but even optimal. We summarize this here:

Definition 3.1: Given an LP in general form, called the primal the dual is defined as follows:

PRIMAL		DUAL
$\min c'x$		max Π'b
$a_i x = b_i$	$i \in M$	$\pi_i \geq 0 \; or \; \leq 0$
$a_i x \geq b_i$	$i \in M'$	$\pi_i \geq 0$
$x_j \geq 0$	$j \in N$	$\pi' A_j \leq c_j$
$x_j \geq 0 \; or \; \leq 0$	$j \in N'$	$\pi'^{A_j} = c_j$

Theorem 3.1: If an LP has an optimal solution so does its dual, and at optimality their costs are equal.

$Proof$: Let x and π be feasible solutions to the primal and dual, respectively. Then

$$c' \geq \pi' A x \geq \pi' b$$

That is, the cost in the primal always dominates the cost in the dual. Since we assume the primal has a feasible solution, the dual cannot have a solution unbounded in cost. The dual has the solution π' discussed above, so by the smplex algorithm, it has an optimum. We note that the cost of this π' is

$$\pi' b = c_B^{''} B^{'-1} b = c_B^{''} x'_0$$

which is optimal cost in the primal. Therefore, by (3.7), this π' is optimal in the dual. ■

The next theorem is quite obvious, which we state here:

Theorem 3.2: The dual of the dual is the primal.

Now, what we can observe is, that as both primal and the dual are LPs, so three possibilities hold for both of them: either having a finite optimal solution, or having and unbounded feasible set, or having infeasible solution. We will see now that there exist only one of three possibilities may occur among the combination of them:

Theorem 3.3: Given a primal-dual pair, exactly one of the three situations occur, as indicated in fig 3.1.

$Explanation$:

1. If one of them has a finite optimum solution, the other is bound to have a finite optimal solution too, because one can find out the solution of the dual form the primal, or of the primal from the dual.
2. If one of them is unbounded, the other can't be unbounded, because the unbounded solution of the primal can be made large in that direction (meaning, making the cost of the primal infinitely negative), and as we saw in Theorem 3.1, the cost of dual is always less than the primal solution, so it's cost will also keep decreasing, never letting it to attain a maximum value. Hence there will not exist

any feasible solution to the dual. This means the unboundedness of one the LPs will lead to infeasible solution in the other.

3. When one of the LP has in feasible solution, we may either have an unbounded solution as above mention, or we may also have an infeasible solution in the other LP as well. So both can be Infeasible.

Primal \ Dual	Finite optimum	Unbounded	Infeasible
Finite optimum	1	X	X
Unbounded	X	X	3
Infeasible	X	3	2

Figure 3–1 Possible categories of a primal–dual pair.

3.2 Farkas' Lemma.

Farkas' Lemma is a fundamental fact about vectors in R^n that in a sense captures the essence of duality. We first introduce a useful definition.

Definition 3.2: Givne a set of vectors $a_i \in R^n, i = 1, \ldots, m$, the cone generated by the set $\{a_i\}$, denoted by C(a_i), is

$$C(a_i) = \{x \in R^n : x = \sum_{i=1}^{m} \pi_i a_i, \pi_i \geq 0, i = 1, \ldots, m\} \qquad \square$$

We now see the following result as a direct consequence of Theorem 3.1.

Theorem 3.5: **(Farkas' Lemma)** Given vectors $a_i \in R^n, i = 1, \ldots, m$, and another another $c \in R^n$, then

$$(y'^{a_i} \geq 0 \; for \; all \; i \Rightarrow y'c \geq 0) \;\; \Leftrightarrow c \in C(a_i)$$

$Proof$: We first se the if part, which is really trivial. If

$$c = \sum_{i=1}^{m} \pi_i a_i, \pi_i \geq 0$$

then

$$y'c = \sum_{i=1}^{m} \pi_i (y'a_i) \geq 0$$

if the $y'a_i \geq 0$.

The $only \; if$ part is the heart of the matter. Consider the LP

$$\min c'y$$

$$a'_i y \geq 0 \;\; i = 1, \ldots, m$$

$$y \leq 0 \; or \geq 0$$

This program is feasible, because y=0 is a feasible point. It is also bounded by the hypothesis that $a'_i y \geq 0$ for all I implies $y'c \geq 0$. Therefore the dual

$$\max 0$$

$$\pi' A_j = c_j, \; j = 1, \ldots, m$$

$$\pi \geq 0$$

has a feasible solution, where

$$A_j = col\big(a_{ij}, i = 1, \ldots, m\big) \in R^m$$

if

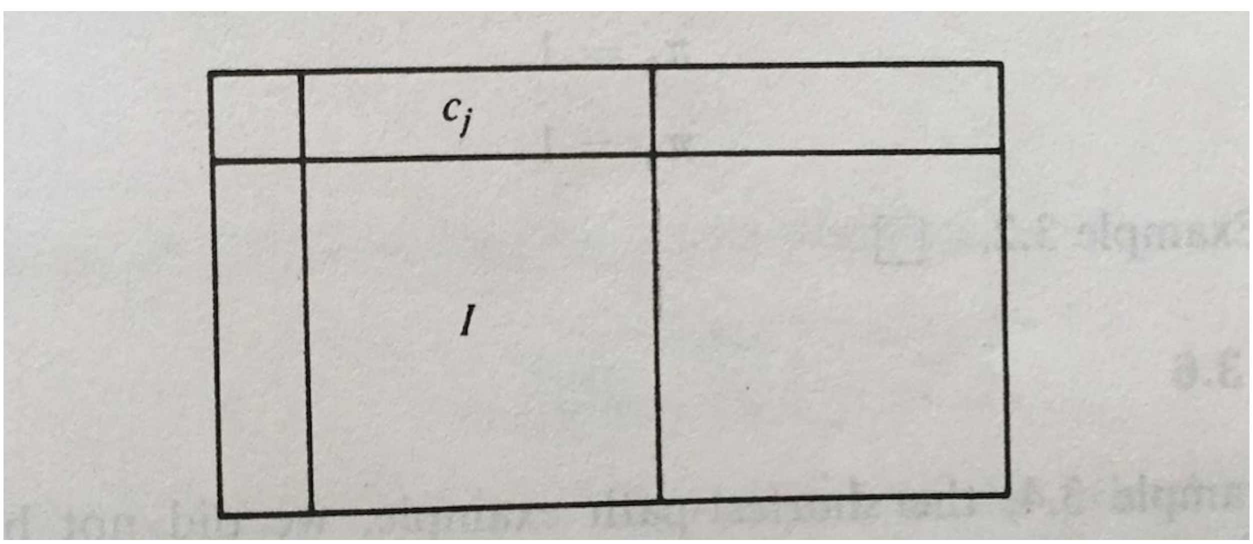
$$a_i = col(a_{ij}, i = 1, \ldots, n) \in R^n$$

Thus there is a Π such that

$$c = \sum_{i=1}^{m} \pi_i a_i, \pi_i \geq 0 \qquad \blacksquare$$

3.3 Dual Information in the Tableau:

We know that the final tableau in the simplex algorithm gives us the optimal solution of the primal, and hence we can also get the optimal solution of the dual. We usually begin with an identity matrix at the left of the tableau; this usually corresponds to artificial or slack variables. The following figure shows the initial tableau:



At the termination of the simplex algorithm at an optimal solution we have in effect multiplied the tableau below the cost row by B^{-1}, where B is the set of ciolumns in the original tableau corresponding to the original optimal bfs. Furthermore the row cost at the optimality becomes

$$c'_j = c_j - \pi' A_j \geq 0$$

where Π is an optimal solution to the dual, as we saw in the proof of Theorem 3.1. In the columns 1 to m, where we assumed we started with an identity matrix, A_j is the unit vector e_j, so that

$$c'_j = c_j - \pi_j, j = 1, \ldots, m$$

Hence, we may obtain an optimal dual from the final tableau by

$$\pi_j = c_j - c'_j \; j = 1, \ldots, m$$

We also note that in the final tableau the position if the initial identity matrix is occupied by B^{-1}. Also, we usually start with $c_j = 0$ in the (actual z) cost row. Thus, in the final tableau, we may get

$$\pi_j = -c'_j, j = 1, .., m$$