COMP4600

Linear Programming

Weifa Liang

Research School of Computer Science

The Australian National University
Canberra, ACT 0200

COMP4600

Textbook: Introduction to Algorithms.
T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. The MIT Press, 3rd Edition, 2009 or later.

Contact: A/Prof Weifa Liang, Room N334, CSIT building.
`Email: wliang@cs.anu.edu.au`, Tel: (02)-6125-3019 (O).

Chapter 29. Linear Programming

Many problems of practical significance take the form of maximizing or minimizing an objective, subject to limited computing resource constraints. If we can specify the objective as a linear function of constraint variables, and the constraints on resources as equalities or inequalities on those variables, then we have **a linear programming problem**. For example,

$$\begin{array}{llllll}
\textit{minimize} & 5x_1 & +4x_2 & -3x_3 & +10x_4 & \text{(objective)} \\
\\
\text{subject to} & -2x_1 & +8x_2 & & +10x_4 \geq 50 & \\
 & 5x_1 & +2x_2 & +5x_3 & -4x_4 \geq 120 & \\
 & 3x_1 & +4x_2 & -7x_3 & +2x_4 \geq 10 & \\
 & \multicolumn{5}{l}{x_i \geq 0,\ i = 1, \ldots, 4.}
\end{array}$$

If we add to a linear program the additional requirement that all variables take integers values, then, that linear program is an **integer linear program** (or ILP for short).

Chapter 29. Linear Programming

Example: Diet Problem, feeding an army.

4 menu choices: Fish, Pizza, Hamburger, and Burrito.

Each choice has benefits (nutrients) and a cost (calories).

	A	C	D	Calories
F	200	90	100	600
P	75	80	250	900
H	275	80	520	550
B	500	95	300	450
RDA	2000	300	475	

The problem is to:

determine least-caloric diet that satisfies RDA of nutrients.

Chapter 29. Linear Programming

Let x_F, x_P, x_H, x_B for the amount consumed of each food.
The objective is to **minimize**

$$600x_F + 900x_P + 550x_H + 450x_B$$

Each nutrient constraint can be expressed as a linear function as follows.

$200x_F + 75x_P + 275x_H + 500x_B \geq 2000$
$90x_F + 80x_P + 80x_H + 95x_B \geq 300$
$100x_F + 250x_P + 520x_H + 300x_B \geq 475$

All food quantities consumed must be non-negative:
$x_F \geq 0$; $x_P \geq 0$; $x_H \geq 0$; $x_B \geq 0$.

This is a Linear Program (or LP).

Chapter 29. Linear Programming

Example:

- A factory has 3 types of machines A, B, C.
- 4 product choices: 1, 2, 3, 4.
- Resource consumed (how much machine time a product takes) and profit per unit

	A	B	C	Profit
1	2	0.5	1.5	3.5
2	2	2	1	4.2
3	0.5	1	3	6.5
4	1.5	2	2	4
Avail.	20/wk	35/wk	17/wk	

The problem is to **determine most profitable product mix** per week.

Chapter 29. Linear Programming

Let x_1, x_2, x_3, x_4 for the units of different products manufactured.
The objective is to **maximize**

$$3.5x_1 + 4.2x_2 + 6.5x_3 + 4x_4$$

Each of the machine constraints can be expressed as a linear function:

$$2x_1 + 2x_2 + 0.5x_3 + 1.5x_4 \leq 20$$
$$0.5x_1 + 2x_2 + x_3 + 2x_4 \leq 30$$
$$1.5x_1 + x_2 + 3x_3 + 2x_4 \leq 15$$

All products must be non-negative:

$$x_1 \geq 0;\ x_2 \geq 0;\ x_3 \geq 0;\ x_4 \geq 0.$$

This is a linear program.

29.1 Notations and notions

Given a set of real numbers $a_1, a_2, \ldots, a_n$ and a set of variables $x_1, x_2, \ldots, x_n$, we define a linear function f on those variables by

$$f(x_1, x_2, \ldots, x_n) = a_1x_1 + a_2x_2 + \ldots + a_nx_n = \sum_{i=1}^{n} a_ix_i.$$

If b is a real number and f is a linear function, then

$$f(x_1, x_2, \ldots, x_n) = b$$

is a linear equality,

and

$$f(x_1, x_2, \ldots, x_n) \leq b$$

or,

$$f(x_1, x_2, \ldots, x_n) \geq b$$

are inequalities.

29.1 Notations and notions

We use the general term linear constraints to denote either linear equalities or linear inequalities.

In linear programming we do not allow **strict inequalities**. Formally,
a linear programming problem is the problem of either minimizing or maximizing a linear function, subject to **a finite set** of linear constraints.

- Minimization linear program
- Maximization linear program

29.2 Overview of linear programming

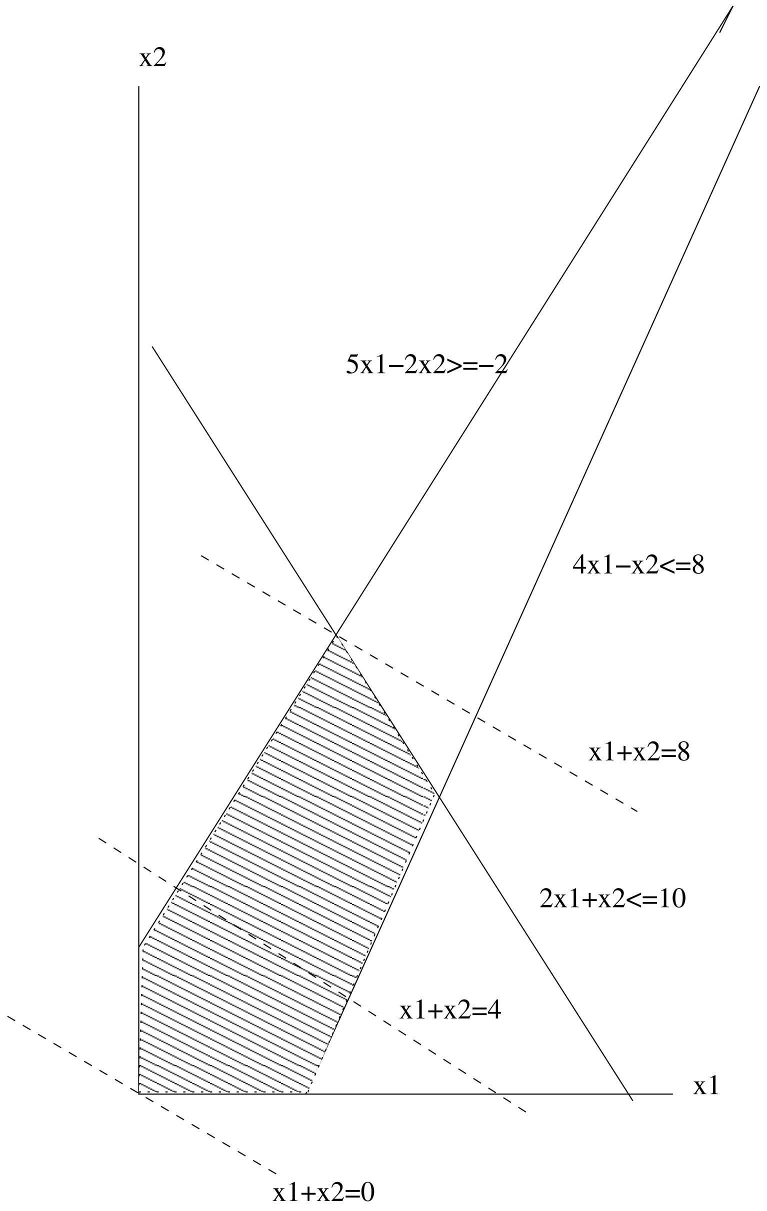
An example: Let us consider an example of linear program.

$$
\begin{array}{ll}
\textit{maximize} & x_1 + x_2 \\
 & \\
\text{subject to} & 4x_1 - x_2 \le 8 \\
 & 2x_1 + x_2 \le 10 \\
 & 5x_1 - 2x_2 \ge -2 \\
 & x_1,\ x_2 \ge 0.
\end{array}
$$

If assigning a pair of values for x_1 and x_2 satisfies all the constraints, the pair of values is refereed to as a feasible solution to the linear program.

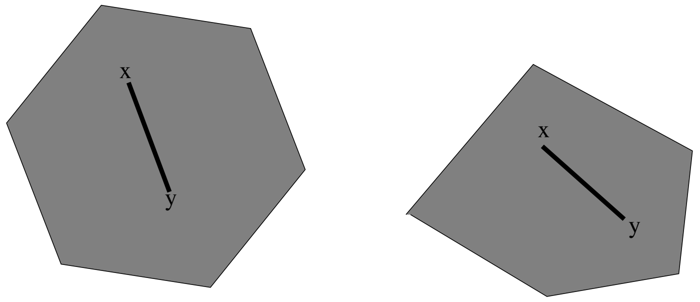
The set of feasible solutions (shaded in the figure) forms a convex region.

If the assignment fails to at least one constraint, then the solution is an infeasible solution to the linear program.

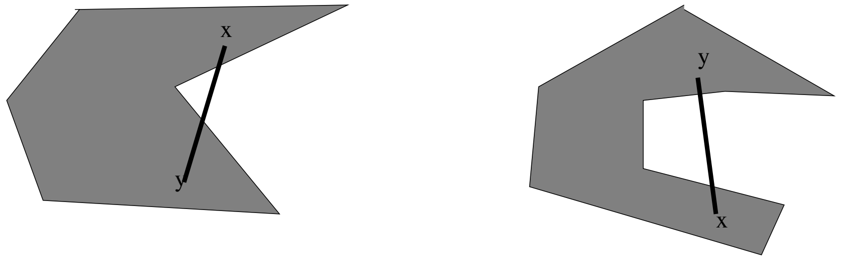

x2
5x1−2x2>=−2
4x1−x2<=8
x1+x2=8
2x1+x2<=10
x1+x2=4
x1
x1+x2=0

29.2.1 Convexity

- A set S is convex if for any two points $x, y \in S$, the line segment xy also lies in S.
- That is, $\alpha x + (1 - \alpha) y \in S$ if $x, y \in S$ for all $\alpha \in [0, 1]$.



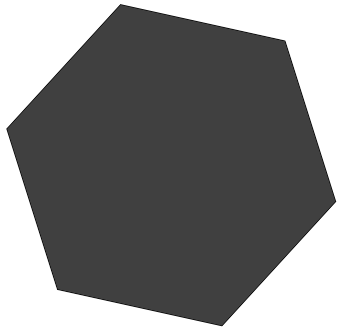
Convex sets

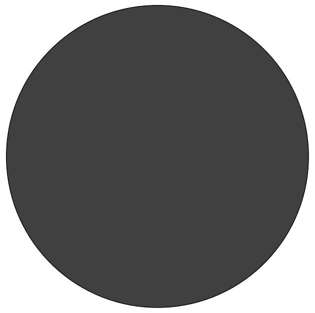


Non–Convex Sets

29.2.2 Extreme Points

- A point x is **an extreme point** of S if it is not a strict convex combination of two other points in S.
- Corners of a polyhedron are extreme. All boundary points of a sphere are extreme.
- Let $S = \{x : Ax = B, x \geq 0\}$, where A is a $m \times n$ matrix of rank $m \leq n$.
- A point x is an extreme point of S iff x is the intersection of n linearly independent hyperplanes.


Extreme Points


All Boundary points are extreme poi

29.2 Overview of linear programming (Cont.)

Algorithms for solving linear programming

- The simplex algorithm (not polynomial algorithm!)
- The ellipsoid algorithm (the 1st polynomial algorithm)
- The interior-point algorithm

29.3 Standard and slack forms

To describe properties of algorithms for linear programs, we use the **standard form** and **slack form** to express a linear programming

- **Standard form:** is the maximization of a linear function, subject to linear inequalities.
- **Slack form:** is the maximization of a linear function subject to linear equalities.

Standard Form:

maximize $\sum_{j=1}^{n} c_j x_j$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \quad \text{for } i = 1, 2, \ldots, m,$$
$$x_j \ge 0, \quad \text{for } j = 1, 2, \ldots, n. \text{ (nonnegativity constraints)}$$

29.3 Standard and slack forms

Converting linear programs into standard form. A linear program may be not in the standard form, because

- The objective function is a minimization (not maximization!)
- There might be variables without nonnegativity constraints
- There might be **equality constraints**
- There might be **inequality constraints** ($\geq$, not $\leq$)

29.3 Standard and slack forms

For example.

minimize $-2x_1 + 3x_2$

subject to

$$x_1 + x_2 = 7$$
$$x_1 - 2x_2 \leq 4$$
$$x_1 \geq 0.$$

In the standard form, we are given n real numbers $c_1, c_2, \ldots, c_n$; m real numbers $b_1, b_2, \ldots, b_m$; and mn real numbers a_{ij} for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.
We wish to find n real numbers $x_1, x_2, \ldots, x_n$ that

maximize $\sum_{j=1}^{n} c_j x_j$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \text{ for } i = 1, 2, \ldots, m$$
$$x_j \geq 0 \text{ for } j = 1, 2, \ldots, n.$$

29.3 Standard and slack forms

Converting it to the standard one:

maximize $2x_1 - 3x_2$
subject to

$$x_1 + x_2 = 7$$
$$x_1 - 2x_2 \leq 4$$
$$x_1 \geq 0.$$

For some variable x_j which does not have a nonnegativity constraint, replace its each appearance by $x'_j - x''_j$ and add the nonnegativity constraints $x'_j \geq 0$ and $x''_j \geq 0$.

Convert the greater-than-or-equal-to constraints to the less-than-or-equal-to constraints thorough by -1. Thus,
maximize $2x_1 - 3x'_2 + 3x''_2$
subject to

$$x_1 + x'_2 - x''_2 = 7$$
$$x_1 - 2x'_2 + 2x''_2 \leq 4$$
$$x_1, x'_2, x''_2 \geq 0.$$

29.3 Standard and slack forms

maximize $2x_1 - 3x_2' + 3x_2''$

subject to

$$x_1 + x_2' - x_2'' \leq 7$$
$$-(x_1 + x_2' - x_2'') \leq -7$$
$$x_1 - 2x_2' + 2x_2'' \leq 4$$
$$x_1, x_2', x_2'' \geq 0.$$

replace x_2' with x_2 and x_2'' with x_3, then

maximize $2x_1 - 3x_2 + 3x_3$

subject to

$$x_1 + x_2 - x_3 \leq 7$$
$$-x_1 - x_2 + x_3 \leq -7$$
$$x_1 - 2x_2 + 2x_3 \leq 4$$
$$x_1, x_2, x_3 \geq 0.$$

29.3 Slack form

Converting linear programs into slack forms, i.e., covert inequalities into equalities.
Let

$$\sum_{j=1}^{n} a_{ij}x_j \le b_i$$

be an inequality constraint.

We introduce a new variable x_{n+i} and rewrite the inequality as the two constraints.

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij}x_j$$

$$x_{n+i} \ge 0.$$

We call x_{n+i} a **slack variable**. The variable x_{n+i} on the left-hand side of the equality is referred to as the **basic variable**,
while the variables in the right-hand side are referred to as the **non-basic variables**.

29.3 Slack form

maximize $2x_1 - 3x_2 + 3x_3$

subject to

$$\begin{aligned} x_4 &= 7 - x_1 - x_2 + x_3 \\ x_5 &= -7 + x_1 + x_2 - x_3 \\ x_6 &= 4 - x_1 + 2x_2 - 2x_3 \\ x_1, x_2, x_3, x_4, x_5, x_6 &\geq 0. \end{aligned}$$

In the slack form we may omit the words " maximize" and "subject to" as well as the explicit nonnegativity constraints. We call the resulting format the **slack form**.

$$\begin{aligned} z &= 2x_1 - 3x_2 + 3x_3 \\ x_4 &= 7 - x_1 - x_2 + x_3 \\ x_5 &= -7 + x_1 + x_2 - x_3 \\ x_6 &= 4 - x_1 + 2x_2 - 2x_3 \\ x_1, x_2, x_3, x_4, x_5, x_6 &\geq 0. \end{aligned}$$

29.3 Slack form

We use N to denote the set of indices of the **Nonbasic variables**,
B to denote the set of indices of the **Basic variables**,
and have $|N| = n$ and $|B| = m$, and $N \cup B = \{1, 2, \ldots, m+n\}$.

We also use v to denote an optional constant term in the objective term. Thus, we can concisely define the slack form by a tuple (N, B, A, b, c, v), denoting the slack form.

$$z = v + \sum_{j=1}^{n} c_j x_j$$
$$x_i = b_i - \sum_{j \in N} a_{ij} x_j \text{ for } i \in B.$$

29.4 Formulating problems as linear programs

Example one: A single source shortest path problem: Given a directed graph $G = (V, E)$ with a source node s and a destination node t,find the length of the shortest path d_t from s to t.

maximize d_t

subject to

$$d_v \leq d_u + w(u,v), \text{ for each edge } (u,v) \in E$$
$$d_s = 0.$$

If we minimize the objective function, then $\bar{d}_v = 0$ for all $v \in V$. We maximize because an optimal solution to the shortest path problem sets each $\bar{d}_v$ to $\min_{u:(u,v)\in E}\{\bar{d}_u + w(u,v)\}$, so that $\bar{d}_v$ is the largest value that is less than or equal to all of the values in the set $\{\bar{d}_u + w(u,v)\}$.

29.4 Formulating problems as linear programs

Example two: Maximum flow problem: Given a directed graph $G = (V, E)$ in which each edge $(u, v) \in E$ has a nonnegative capacity $c(u, v) \geq 0$, two distinguished nodes: source node s and destination node t, find a maximum flow from s to t such that the flow conservation is satisfied.

maximize $\sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs}$

subject to

$f_{uv} \leq c(u, v)$ for each $(u, v) \in E$

$\sum_{v \in V} f_{uv} = \sum_{v \in V} f_{vu}$ for each $u \in V - \{s, t\}$;

$f_{uv} \geq 0$ for each $u, v \in V$.

29.4 Formulating problems as linear programs

Example three: Minimum cost flow: In addition to capacity $c(u,v)$ for each edge (u,v), we are given a real-valued cost $a(u,v) \geq 0$, a source and destination node s and t with demand d from the source node.

Find a flow from s to t to transform the demand from the source node to its destination node such that the total cost is minimized.

minimize $\sum_{(u,v)\in E} a(u,v) f_{uv}$
subject to

$$f_{uv} \leq c(u,v) \text{ for each } (u,v) \in E$$
$$\sum_{v\in V} f_{uv} - \sum_{v\in V} f_{vu} = 0 \text{ for each } u \in V - \{s,t\};$$
$$\sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs} = d$$
$$f_{uv} \geq 0 \text{ for each } u,v \in V.$$

29.4 Formulating problems as linear programs

Example four: Multi-commodity flow: Given a directed graph $G = (V,E)$ in which each edge $(u,v) \in E$ has a nonnegative capacity $c(u,v) \geq 0$, we are given k different commodities $K_1, K_2, \ldots, K_k$ where we specify commodity i by a triple $(s_i, t_i; d_i)$. Here s_i is the source node, t_i is the destination node and d_i the demand for commodity i.

Let f_i be a flow for commodity i, f_{iuv} is the flow of commodity i from node u to node v to be a real-valued function that satisfies the flow-conservation and capacity constraints. Let f_{uv} be the aggregate flow on edge (u,v) so $f_{uv} = \sum_{i=1}^{k} f_{iuv}$ which is no more then the capacity of the edge (u,v), as the problem is to determine whether there is a flow such that each commodity can be routed to its destination.

29.4 Formulating problems as linear programs

The optimization objective is null in this case.

minimize 0

subject to

$\sum_{i=1}^{k} f_{iuv} \leq c(u,v)$ for each $(u,v) \in E$

$\sum_{v\in V} f_{iuv} - \sum_{v\in V} f_{ivu} = 0$ for each $i = 1,2,\ldots,k$ and for each $u \in V - \{s_i, t_i\}$;

$\sum_{v\in V} f_{i,s_i,v} - \sum_{v\in V} f_{i,v,s_i} = d_i$ for each $i = 1,2,\ldots,k$

$f_{iuv} \geq 0$ for each $u, v \in V$ and for each $i = 1,2,\ldots,k$.

The only known polynomial-time algorithm for this problem expresses it as a linear program.

29.5 The Simplex Algorithm

The simplex algorithm is a classical method for linear program. Although its running time is **not polynomial in the worst case**, it is often remarkably fast in practice.

Similar to Gaussian elimination, the simplex algorithm begins with a system of linear equalities whose solution is unknown. In each iteration we rewrite the system in **an equivalent form** that has some additional structure. After some number of iterations, we have rewritten the system so that the solution is simple to obtain.

The main idea of each iteration: Associated with each iteration, there is a **basic solution** that can easily be obtained from the slack form of the linear program:

Set each non-basic variable to 0 and compute the values of the basic variables from the equality constraints. An iteration coverts one slack form into an equivalent slack form. **The objective value of the associated basic feasible solution will be no less than that at the previous iteration**.

29.5 The Simplex Algorithm

- We choose a non-basic variable such that if we were to increase that variable's value from 0, then the objective value would increase too. The amount of the increase is limited by other constraints. We raise it until some basic variables become 0.
- If a basic solution is feasible, we call it a **basic feasible solution**. Our goal in each iteration is to reformulate the linear program so that the basic solution has a greater objective value.
- We choose a non-basic variable x_e whose coefficient in the objective function is positive and we increase x_e as much as possible without violating any of the constraints. The variable x_e becomes basic, and some other variable x_l becomes non-basic.

29.5 The Simplex Algorithm

A **pivot** operation is to choose a non-basic variable x_e called **entering variable** and a basic variable x_l called the **leaving variable**, and exchange their roles.

For example:

$$
\begin{aligned}
z &= 3x_1 + x_2 + 2x_3 \\
x_4 &= 30 - x_1 - x_2 - 3x_3 \\
x_5 &= 24 - 2x_1 - 2x_2 - 5x_3 \\
x_6 &= 36 - 4x_1 - x_2 - 2x_3. \quad (*)
\end{aligned}
$$

Let us increase the value of x_1 (its value range is $\min\{30, 12, 9\}$), then x_4, x_5, and x_6 all decrease. We switch x_1 with x_6 in (*), then

$$x_1 = 9 - x_2/4 - x_3/2 - x_6/4$$

.

29.5 The Simplex Algorithm

Using x_1 to recalculate x_4,

$$x_4 = 30 - (9 - x_2/4 - x_3/2 - x_6/4) - x_2 - 3x_3 = 21 - 3x_2/4 - 5x_3/2 + x_6/4.$$

We then have the following equivalent slack form

$$\begin{aligned} z &= 27 + x_2/4 + 2x_3/4 - 3x_6/4 \\ x_1 &= 9 - x_2/4 - x_3/2 - x_6/4 \\ x_4 &= 21 - 3x_2/4 - 5x_3/2 + x_6/4 \\ x_5 &= 6 - 3x_2/2 - 4x_3 + x_6/2 \quad (**). \end{aligned}$$

We then switch x_3 with x_5 in (**) (ANOTHER PIVOT OPERATION), The value range of x_3 is $\min\{18, 42/5, 3/2\}$.

$$x_3 = 3/2 - 3x_2/8 - x_5/4 + x_6/8.$$

29.5 The Simplex Algorithm

We then have the following equivalent slack form:

$$\begin{aligned}
z &= 111/4 + x_2/16 - x_5/8 - 11x_6/16 \\
x_1 &= 33/4 - x_2/16 + x_5/8 - 5x_6/16 \\
x_3 &= 3/2 - 3x_2/8 - x_5/4 + x_6/8 \qquad (^{***}) \\
x_4 &= 69/4 + 3x_2/16 + 5x_5/8 - x_6/16.
\end{aligned}$$

We finally switch x_2 with x_3 in (***)(ANOTHER PIVOT OPERATION), the value range of x_2 is $\min\{132, 4, \infty\}$, then

$$\begin{aligned}
z &= 28 - x_3/6 - x_5/6 - 2x_6/3 \\
x_1 &= 8 + x_3/6 + x_5/6 - x_6/3 \\
x_2 &= 4 - 8x_3/3 - 2x_5/3 + x_6/3 \; (^{***}) \\
x_4 &= 18 - x_3/2 + x_5/2.
\end{aligned}$$

The optimal solution thus is $(8, 4, 0, 18, 0, 0)$.

29.5 The Simplex Algorithm (2nd example)

Consider LP: max $z = 2x_1 + 3x_2$
subject to

$$\begin{aligned} x_1 - 2x_2 + x_3 &= 4 \\ 2x_1 + x_2 + x_4 &= 18 \\ x_2 + x_5 &= 10 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0. \end{aligned}$$

- Initial basic solution: $(x_3, x_4, x_5) = (4, 18, 10)$ and $(x_1, x_2) = (0, 0)$. Objective function value z is zero
- Entering variable: pick a x_i with $c_i > 0$ in the objective and make it basic (e.g., x_2)
- Leaving variable. Increase x_1 from zero, until some basic variable becomes zero (and non-basic)
- Repeat until all variables in objective function have negative coefficients.

29.5 The Simplex Algorithm

Consider LP max $z = 2x_1 + 3x_2$

subject to

$$\begin{aligned} x_3 &= 4 - x_1 + 2x_2 \\ x_4 &= 18 - 2x_1 - x_2 \\ x_5 &= 10 - x_2 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0. \end{aligned}$$

- Set $x_1 = x_2 = 0$,$x_3 = 4$,$x_4 = 18$, $x_5 = 10$.
- Entering variable is x_2.
- Max value of x_2 is $\min\{18, 10, \infty\}$
- Leaving variable is x_5

29.5 The Simplex Algorithm

Rearrange the linear system:

$$x_2 = 10 - x_5$$
$$x_3 = 24 - x_1 - 2x_5$$
$$x_4 = 8 - 2x_1 + x_5$$

- Set $x_1 = x_5 = 0$ and $x_2 = 10$, $x_3 = 24$, $x_4 = 8$.
- Objective function becomes: $z = 30 + 2x_1 - 3x_5 = 30$.
- Entering variable is x_1
- Max value of x_1 is $\min\{\infty, 24, 4\}$
- Leaving variable is x_4.

29.5 The Simplex Algorithm

Rearrange the linear system:

$$x_1 = 4 - \tfrac{1}{2}x_4 + \tfrac{1}{2}x_5$$
$$x_2 = 10 - x_5$$
$$x_3 = 20 + \tfrac{1}{2}x_4 - \tfrac{5}{2}x_2$$

- Set $x_4 = x_5 = 0$ and $x_1 = 4$, $x_2 = 10$, $x_3 = 20$.
- Objective function becomes: $z = 38 - x_4 - 2x_5 = 38$.
- Entering variable is x_1
- This is optimal because all coefficients are negative.

29.5 The Simplex Algorithm

```
SIMPLEX($A, b, c$)
1   $(N, B, A, b, c, v) = Initialize - SIMPLEX(A, bc)$ (see Textbook, Page 887, 3rd)
2   Let $\Delta$ be a new vector of length $m$
3   while some index $j \in N$ has $c_j > 0$ do
4       choose an index $e \in N$ for which $c_e > 0$
5       for each index $i \in B$ do
6           if $a_{ie} > 0$
7               $\Delta_i \leftarrow b_i / a_{ie}$
8           else $\Delta_i \leftarrow \infty$;
9           choose an index $l \in B$ that minimizes $\Delta_l$
10          if $\Delta_l == \infty$
11              return "unbounded"
12          else $(N, B, A, b, c, v) = PIVOT(N, B, A, b, c, v, l, e)$;
13  for $i \leftarrow 1$ to $n$ do
14      if $i \in B$
```

```
                $\bar{x}_i \leftarrow b_i$;
            **else** $\bar{x}_i \leftarrow 0$
    **return** $(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$.
```

Where the Initialize procedure determines whether the linear program is infeasible or returns a slack form for which the basic solution is feasible. **If all coefficients in the objective function are negative, the while loop will terminate**.

29.5 The Simplex Algorithm (the initial basic feasible solution)

We here describe how to test if a linear program is feasible, if it is, how to produce a slack form for it. A linear program can be feasible, yet the initial basic solution may not be feasible. Consider the following LP:

maximize $2x_1 - x_2$

subject to

$$2x_1 - x_2 \leq 2$$
$$x_1 - 5x_2 \leq -4$$
$$x_1, x_2 \geq 0$$

If we were to convert this LP to slack form, the basic solution would set $x_1 = 0$ and $x_2 = 0$. This solution violates one of the two constraints!, so it is not a feasible solution.

29.5 The Simplex Algorithm (the initial basic feasible solution)

In order to determine whether there is a solution to the LP, we formulate an **auxiliary linear program**, for this auxiliary LP, we are able to find a slack form from which the basic solution is feasible. Furthermore, the solution of this auxiliary LP will determine whether the initial LP is feasible.

Lemma Let L be a linear program in the standard form. Let L_{max} be the following linear program with $n+1$ variables:

maximize $-x_0$

subject to

$\sum_{j=1}^{n} a_{ij}x_j - x_0 \leq b_i$ for $i = 1, 2, \ldots, m$,

$x_j \geq 0$ for $j = 0, 1, 2, \ldots, n$.

Proof. Assume that L has a feasible solution $\bar{x} = (\bar{x_1}, \bar{x_2}, \ldots, \bar{x_n})$. Then, the solution $\bar{x_0} = 0$ combined with $\bar{x}$ is a feasible solution to L_{max} with objective value 0. Since

$x_0 \geq 0$ is a constant of L_{max} and the objective function is to maximize $-x_0$, this solution must be optimal for L_{max}.

Conversely, suppose that the optimal objective value of L_{max} is 0. Then $\bar{x} = 0$, and the value of the remaining variables $\bar{x}$ satisfy the constraints of L. .

Rewrite the original LP using the auxiliary LP as follows.

maximize $-x_0$

subject to

$$2x_1 - x_2 - x_0 \leq 2$$

$$x_1 - 5x_2 - x_0 \leq -4$$

$$x_1, x_2, x_0 \geq 0$$

29.5 The Simplex Algorithm (the initial basic feasible solution)

We write the auxiliary LP in slack form as follows.

$$\begin{aligned} z &= -x_0 \\ x_3 &= 2 - 2x_1 + x_2 + x_0 \\ x_4 &= -4 - x_1 + 5x_2 + x_0 \text{ (*)} \end{aligned}$$

Perform a Pivot operation by swapping x_4 and x_0.

$$\begin{aligned} z &= -4 - x_1 + 5x_2 - x_4 \\ x_3 &= 6 - x_1 - 4x_2 + x_4 \text{ (**)} \\ x_0 &= 4 + x_1 - 5x_2 + x_4 \end{aligned}$$

After a series of Pivot operations we have

$$\begin{aligned} z &= 4/5 + 9x_1/5 - x_4/5 \\ x_2 &= 4/5 + x_1/5 + x_4/5 \\ x_3 &= 14/5 - 9x_1/5 + x_4/5 \end{aligned}$$

29.6 Duality

We have shown that under certain assumptions, SIMPLEX terminates. We have not yet shown that it actually finds an optimal solution. In order to do so, we introduce a powerful concept called **linear programming duality**. Duality enables us to show that a solution is indeed optimal.

Recall in the proof of the maximum flow and minimum cut theorem, when we find a flow f with value $|f|$, how do we know whether f is a maximum flow? By the theorem, if we can find a cut whose value is also $|f|$, then we have verified that f is indeed a maximum flow. This relationship provides an example of duality: **Given a maximization problem, define a related minimization problem such that the two problems have the same optimal objective values**.

29.6 Duality (Cont.)

Given a **primal linear program** (the original linear program) in the standard form:

maximize $\sum_{j=1}^{n} c_j x_j$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \text{ for } i = 1, 2, \ldots, m$$
$$x_j \geq 0 \text{ for } j = 1, 2, \ldots, n.$$

The **dual linear program** of the linear program is defined as follows.

minimize $\sum_{i=1}^{m} b_i y_i$

subject to

$$\sum_{i=1}^{m} a_{ij} y_i \geq c_j \text{ for } j = 1, 2, \ldots, n$$
$$y_i \geq 0 \text{ for } i = 1, 2, \ldots, m.$$

29.6 Duality (Cont.)

An example: Given a primal linear program in the standard form:

maximize $3x_1 + x_2 + 2x_3$

subject to

$$\begin{aligned} &x_1 + x_2 + 3x_3 \leq 30 \\ &2x_1 + 2x_2 + 5x_3 \leq 24 \\ &4x_1 + x_2 + 2x_3 \leq 36 \\ &x_1, x_2, x_3 \geq 0. \end{aligned}$$

The dual linear program of the linear program is defined as follows.

minimize $30y_1 + 24y_2 + 36y_3$

subject to

$$\begin{aligned} &y_1 + 2y_2 + 4y_3 \geq 3 \\ &y_1 + 2y_2 + y_3 \geq 1 \\ &3y_1 + 5y_2 + 2y_3 \geq 2 \\ &y_1, y_2, y_3 \geq 0. \end{aligned}$$

29.6 Duality (Cont.)

Let $\bar{x}$ be any feasible solution to the primal linear program and let $\bar{y}$ be any feasible solution to the dual linear program. Then, we have

$$\sum_{j=1}^{n} c_j \bar{x}_j \leq \sum_{i=1}^{m} b_i \bar{y}_i.$$

$$\begin{aligned}
\sum_{j=1}^{n} c_j \bar{x}_j &\leq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} \bar{y}_i\right) \bar{x}_j && (1)\\
&= \sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} \bar{x}_j\right) \bar{y}_i && (2)\\
&\leq \sum_{i=1}^{m} b_i \bar{y}_i. && (3)
\end{aligned}$$

Lemma: Let $\bar{x}$ be any feasible solution to the primal linear program (A, b, c), and let $\bar{y}$ be any feasible solution to the dual linear program. If $\sum_{j=1}^{n} c_j \bar{x}_j = \sum_{i=1}^{m} b_i \bar{y}_i$, then, $\bar{x}$ and $\bar{y}$ are optimal solutions to the primal and dual linear programs, respectively.

29.6 Duality (Cont.)

- **Weak Duality:** if $\bar{x}$ and $\bar{y}$ are feasible solutions of the primal and dual LPs, then

$$\bar{x} \leq \bar{y}$$

  Following the definition of dual LP, each feasible solution of dual is **an upper bound** on primal.

- **Strong Duality:** if primal LP has a finite optimization solution $\bar{x}^*$, then the dual also has a finite optimal solution $\bar{y}^*$ and

$$\bar{x}^* = \bar{y}^*$$

29.6 Duality (Cont.)

Theorem: Suppose that SIMPLEX returns values $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ for the primal linear program (A, b, c). Let N and B denote the non-basic and basic variables for the final slack form, let c' denote the coefficients in the final slack form:

$z = v' + \sum_{j \in N} c'_j x_j$
$x_i = b'_i - \sum_{j \in N} a'_{ij} x_j$ for $i \in B$.

and let $\bar{y} = (\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_m)$ be defined as follows.

$$\bar{y}_i = \max \begin{cases} -c'_{n+i} & \text{if } n+i \in N, \\ 0 & \text{otherwise.} \end{cases}$$

Then, $\bar{x}$ is an optimal solution to the primal linear program and $\bar{y}$ is an optimal solution to the dual linear program, and

$$\sum_{j=1}^{n} c_j \bar{x}_j = \sum_{i=1}^{m} b_i \bar{y}_i$$

.

29.6 Duality (Cont.)

Proof: Suppose we run SIMPLEX on a primal linear program, the algorithm proceeds through a series of slack forms until it terminates with a final slack form with objective function

$z = v' + \sum_{j \in N} c'_j x_j$.

Since SIMPLEX terminated with a solution, we know that $c'_j \leq 0$ for all $j \in N$. If we define $c'_j = 0$ for all $j \in B$. We rewrite the equation as

$$\begin{aligned} z &= v' + \sum_{j \in N} c'_j x_j && (4) \\ &= v' + \sum_{j \in N} c'_j x_j + \sum_{j \in B} c'_j x_j \quad \text{because } c'_j = 0 \text{ for all } j \in B && (5) \\ &= v' + \sum_{j=1}^{m+n} c'_j x_j && (6) \end{aligned}$$

29.6 Duality (Cont.)

For the base solution $\bar{x}$ associated with this final slack form, $\bar{x}_j = 0$ for all $j \in N$, and $z = v'$. Since all slack forms are equivalent, if we evaluate the original objective function $\bar{x}$, we must have the same objective value

$$\begin{aligned}
\sum_{j=1}^{n} c_j \bar{x}_j &= v' + \sum_{j=1}^{m+n} c'_j \bar{x}_j && (7)\\
&= v' + \sum_{j \in N} c'_j \bar{x}_j + \sum_{j \in B} c'_j \bar{x}_j && (8)\\
&= v' + \sum_{j \in N} (c'_j \times 0) + \sum_{j \in B} (0 \times \bar{x}_j) && (9)\\
&= v'. && (10)
\end{aligned}$$

We now show $\bar{y}$ is a feasible for the dual linear program and its objective value $\sum_{i=1}^{m} b_i \bar{y}_i$ equals $\sum_{j=1}^{n} c_j \bar{x}_j$. In general, all slack forms are equivalent, then $\sum_{j=1}^{n} c_j x_j = v' + \sum_{j=1}^{m+n} c'_j x_j$.

29.6 Duality (Cont.)

$$\begin{align}
\sum_{j=1}^{n} c_j\bar{x}_j &= v' + \sum_{j=1}^{m+n} c'_j\bar{x}_j \tag{11}\\
&= v' + \sum_{j=1}^{n} c'_j\bar{x}_j + \sum_{j=n+1}^{m+n} c'_j\bar{x}_j \tag{12}\\
&= v' + \sum_{j=1}^{n} c'_j\bar{x}_j + \sum_{i=1}^{m} c'_{n+i}\bar{x}_{n+i} \tag{13}\\
&= v' + \sum_{j=1}^{n} c'_j\bar{x}_j + \sum_{i=1}^{m} (-\bar{y}_i)\bar{x}_{n+i} \tag{14}\\
&= v' + \sum_{j=1}^{n} c'_j\bar{x}_j + \sum_{i=1}^{m} (-\bar{y}_i)(b_i - \sum_{j=1}^{n} a_{ij}\bar{x}_j) \tag{15}\\
&= v' + \sum_{j=1}^{n} c'_j\bar{x}_j - \sum_{i=1}^{m} b_i\bar{y}_i + \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}\bar{x}_j\bar{y}_i \tag{16}\\
& \tag{17}
\end{align}$$

29.6 Duality (Cont.)

$$= v' + \sum_{j=1}^{n} c'_j \bar{x}_j - \sum_{i=1}^{m} b_i \bar{y}_i + \sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij}\bar{y}_i\bar{x}_j \quad (18)$$

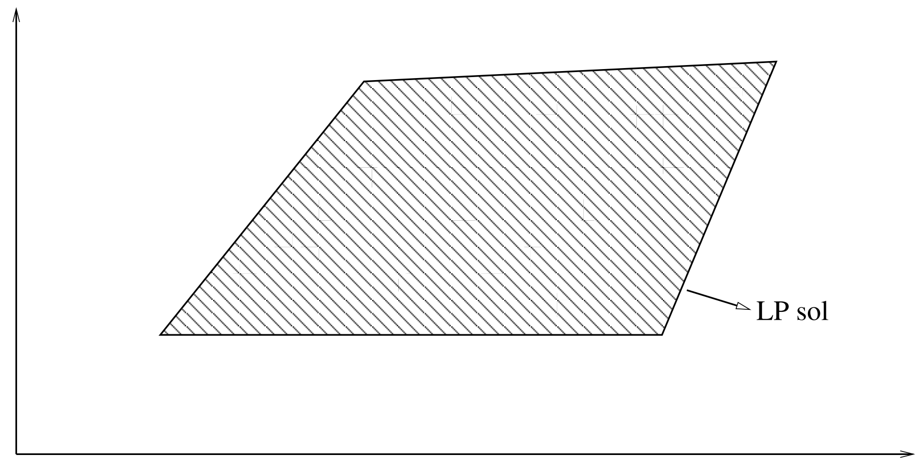
$$= (v' - \sum_{i=1}^{m} b_i \bar{y}_i) + \sum_{j=1}^{n} (c'_j + \sum_{i=1}^{m} a_{ij}\bar{y}_i)\bar{x}_j \quad (19)$$

$$= (0) + \sum_{j=1}^{n} (c_j)\bar{x}_j \quad (20)$$

$$(21)$$

Thus, $v' = \sum_{i=1}^{m} b_i\bar{y}_i$. The rest is to show $\bar{y}$ is a feasible solution to the dual linear program. We have $c'_j \le 0$ for all $j = 1, 2 \ldots, m+n$. Thus, for any $j = 1, 2, \ldots, n$, $c_j = c'_j + \sum_{i=1}^{m} a_{ij}\bar{y}_i \le \sum_{i=1}^{m} a_{ij}\bar{y}_i$.

29.7 Integer Programming



● A feasible IP solution

maximize cx

subject to

$$Ax \leq b \qquad\qquad x \text{ integers (only difference from LP)}$$

IP feasible space is discrete, and smaller than LP feasible space which is continuous.

29.7 Integer Programming

- One might think that restricting variables makes the IP problems easier to solve. In fact IP is NP-Complete

- Reduction from 3-SAT problem to it as follows.
  For example: $\Phi = (x_1 \vee \bar{x_3} \vee x_4) \wedge (\bar{x_2} \vee x_3 \vee \bar{x_4}) \wedge \ldots$
  minimize x_1
  subject to

$$
\begin{aligned}
& x_1 + (1 - x_3) + x_4 \geq 1 \\
& (1 - x_2) + x_3 + (1 - x_4) \geq 1 \\
& \vdots \\
& x_i \in \{0, 1\}.
\end{aligned}
$$

29.7 Integer Programming

Examples:

A. Knapsack Problem: n items, weight a_i with benefit c_i, bin capacity b.

maximize $\sum_{i=1}^{n} c_i x_i$
subject to

$$\sum_{i=1}^{n} a_i x_i \leq b$$
$$x_i \in \{0,1\}.$$

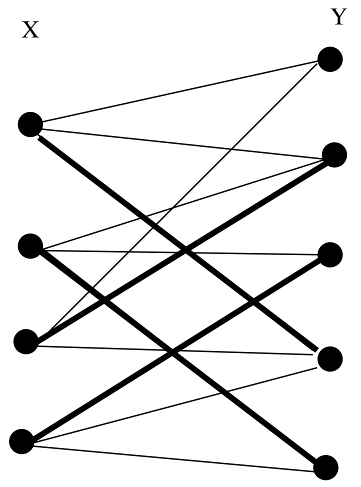
B. Vertex Cover Problem: $G = (V, E)$

minimize $\sum_{i=1}^{n} x_i$
subject to

$$x_i + x_j \geq 1 \quad \text{for each edge } (i, j) \in E$$
$$x_i \in \{0,1\}.$$

29.7 Integer Programming

C. Assignment Problem: $G = (X, Y, E)$, $c(i, j) = c_{ij}$ for each edge $(i, j) \in E$ with $1 \leq i \leq n$ and $1 \leq j \leq m$.



maximize $\sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij}$

subject to

$$\sum_{i:(i,j)\in E} x_{ij} = 1 \quad \text{for all } i \in X$$
$$\sum_{j:(i,j)\in E} x_{ij} = 1 \quad \text{for all } j \in Y$$
$$x_{ij} \in \{0, 1\}.$$

29.8 LP Relaxation of IP

An IP formulation with intergrality constraint removed is called **the LP relaxation**. If IP is max cx with an optimal solution z_I, and z_L is the optimal solution of the relaxed LP, then

$$z_I \leq z_L.$$

It follows, because the IP feasible space is a subset of the LP feasible space.