#50

Mini report 6: Taro's problem

```
var x1 binary;
var x2 binary;
var x3 binary;
var x4 binary;
var x5 binary;
var x6 binary;
var x7 binary;
var x8 binary;
var x9 binary;

minimize cost: 86 * x1 + 64 * x2 + 86 * x3 + 43 * x4 + 86 * x5 + 86 * x6 + 108 * x7 + 216 * x8 + 248 * x9;

subject to
 c1: 0.1 * x1 + 1.1 * x2 + 0.2 * x3 + 0.2 * x4 + 1.2 * x5 + 0.2 * x6 + 0.3 * x7 + 1.8 * x8 + 2.0 * x9 >= 2.0;
 c2: 0.2 * x1 + 0 * x2 + 0.6 * x3 + 0.1 * x4 + 0.2 * x5 + 0 * x6 + 0 * x7 + 0.1 * x8 + 0.1 * x9 >= 1.0;
 c3: 2.0 * x1 + 1.0 * x2 + 2.4 * x3 + 0.1 * x4 + 2.1 * x5 + 4.0 * x6 + 5.0 * x7 + 2.8 * x8 + 3.5 * x9 >= 7;
 c4: 2.0 * x1 + 1.0 * x2 + 2.4 * x3 + 0.1 * x4 + 2.1 * x5 + 4.0 * x6 + 5.0 * x7 + 2.8 * x8 + 3.5 * x9 <= 10;
 c5: 0.1 * x1 + 0.2 * x2 + 0.1 * x3 + 1.5 * x4 + 0.3 * x5 + 0 * x6 + 0 * x7 + 1.0 * x8 + 1.6 * x9 <= 2.5;
```

Notation of vectors, where the superscript T shows the transpose (row <-> column).

$$x = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9)^T \in \{0,1\}^9$$
$$c = (86, 64, 86, 43, 86, 86, 108, 216, 248)^T \in \mathbb{R}^9$$
$$r = (0.1, 1.1., 0.2, 0.2, 1.2, 0.2, 0.3, 1.8, 2.0)^T \in \mathbb{R}^9$$
$$g = (0.2, 0, 0.6, 0.1, 0.2, 0, 0, 0.1, 0.1)^T \in \mathbb{R}^9$$
$$b = (2.0, 1, 0, 2.4, 0.1, 2.1, 4.0, 5.0, 2.8, 3.5)^T \in \mathbb{R}^9$$
$$s = (0.1, 0.2, 0.1, 1.5, 0.3, 0, 0, 1.0, 1.6)^T \in \mathbb{R}^9$$

Notation: $c \cdot x = c^T x = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$

=> Taro's problem: $\min c^T x$

subject to
$$r^T x \geq 2.0$$
$$g^T x \geq 1.0$$
$$b^T x \geq 7$$
$$b^T x \leq 10$$
$$s^T x \leq 2.5$$
$$x \in \{0,1\}^9$$

Notations: $b = (2.0, 1.0, 7, -10, -2.5)^T \in \mathbb{R}^5$

$$A = (r, g, b, -b, -s)^T \in \mathbb{R}^{5\times 9}$$

=> Taro's problem: $\min c^T x$

subject to
$$Ax \geq b$$
$$x \in \{0,1\}^9$$

#50

Mini report 6: Taro's problem

```
# Model specification

param m;
param n;
param c{1..n};
param b{1..m};
param A{1..m, 1..n};

var x{1..n} binary;

minimize cost:
    sum{i in 1..n}c[i] * x[i];

subject to constraint {i in 1..m}:
    sum{j in 1..n} A[i,j] * x[j] >= b[i];

# Data specification

data;

param n := 9;
param m := 5;

param c :=
    1 86
    2 64
    3 86
    4 43
    5 86
    6 86
    7 108
    8 216
    9 248 ;

param b :=
    1 2.0
    2 1.0
    3 7.0
    4 -10
    5 -2.5 ;

param A : 1 2 3 4 5 6 7 8 9 :=
    1   0.1  1.1  0.2  0.2  1.2  0.2  0.3  1.8  2.0
    2   0.2  0.0  0.6  0.1  0.2  0.0  0.0  0.1  0.1
    3   2.0  1.0  2.4  0.1  2.1  4.0  5.0  2.8  3.5
    4  -2.0 -1.0 -2.4 -0.1 -2.1 -4.0 -5.0 -2.8 -3.5
    5  -0.1 -0.2 -0.1 -1.5 -0.3 -0.0 -0.0 -1.0 -1.6 ;

end;
```

#51

* Linear Programming (LP) and its dual problem

A standard form of LP

(P) $\max\{ c^T x \mid Ax \le b,\ x \ge 0 \}$

$x, c \in \mathbb{R}^n,\ b \in \mathbb{R}^m,\ A \in \mathbb{R}^{m \times n}$

Dual problem

(D) $\min\{ b^T y \mid A^T y \ge c,\ y \ge 0 \}$ $\quad y \in \mathbb{R}^m$

Theorem (Weakly duality theorem)

For any x and y that are feasible to (P) and (D) respectively,

$$c^T x \le b^T y.$$

Proof.

$$c^T x = x^T c \le x^T A^T y = (Ax)^T y$$

$$= y^T (Ax) \le y^T b = b^T y \quad \blacksquare$$

Q: Under what condition can the equality hold?

A: $\left\{ \begin{array}{l} x^T c = x^T A^T y \\ y^T A x = y^T b \end{array} \right. \iff \left\{ \begin{array}{l} x_i = 0 \text{ or } c_i = (A^T y)_i \text{ or both } \quad \forall i \\ y_j = 0 \text{ or } (Ax)_j = b_j \text{ or both } \quad \forall j \end{array} \right.$

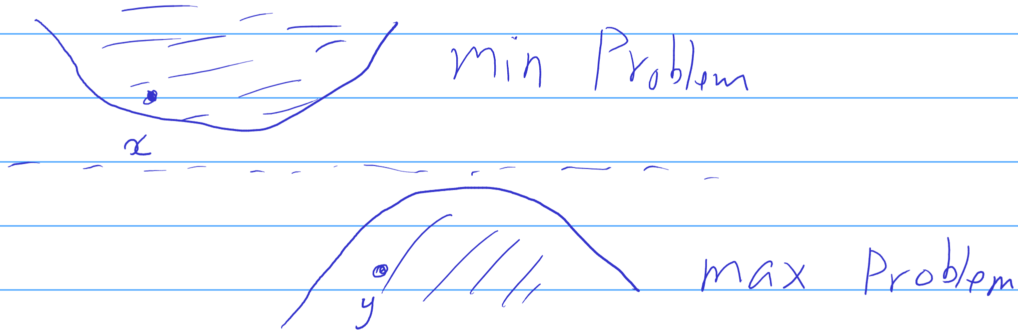
(P) (D)

$$\left\{ \begin{array}{l} x_i \left((A^T y)_i - c_i \right) = 0 \quad \forall i, j \\ y_j \left((Ax)_j - b_j \right) = 0 \end{array} \right.$$

(D) (P)

Complementary slackness

* Illustration of the (weakly) duality theorem



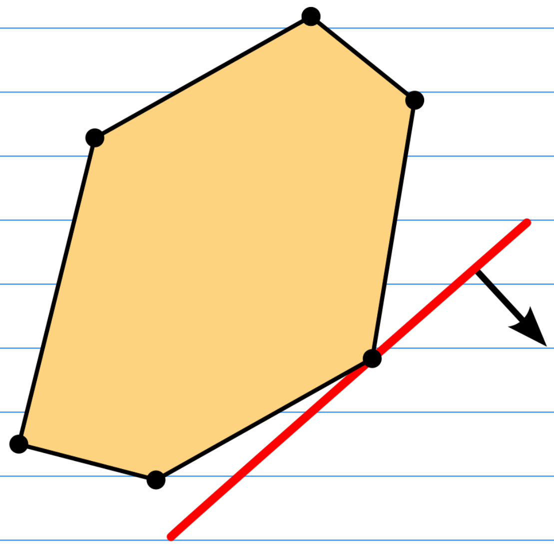
Corollary:

For any x and y feasible to (P) and (D) respectively, they are optimal solutions if $c^T x = b^T y$.

Theorem (Strong duality theorem)

If (P) (or (D)) has an optimal solution x* (or y*), then (D) (or (P)) has a feasible (optimal) solution y* such that $c^T x^* = b^T y^*$.

Illustration of LP and the simplex method for LP



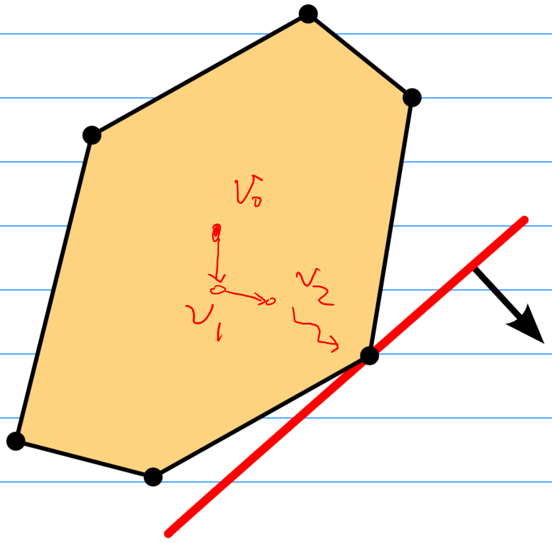
"A pictorial representation of a simple linear program with two variables and six inequalities. The set of feasible solutions is depicted in yellow and forms a polygon, a 2-dimensional polytope. The linear cost function is represented by the red line and the arrow: The red line is a level set of the cost function, and the arrow indicates the direction in which we are optimizing."

https://en.wikipedia.org/wiki/Linear_programming

Simplex method: Start from some vertex; Find a descending neighbor vertex (for minimization problem); Repeat until no such a vertex can be found.

vertex : basic solutions of some linear equations.

* Interior-point method



Start from some interior point;

Find a descending direction and a interior point along that direction;

Repeat until no such a direction (point) can be found.

Advantage: Gradient descent or even Newton's method can be used.

Transform to the standard formulation

(1) $\min c^T x \iff \max (-c)^T x$

(2) $a^T x \geq b \iff (-a)^T x \leq (-b)$

(3) $a^T x = b \iff a^T x \geq b$ and $a^T x \leq b$

$\iff (-a)^T x \leq (-b)$ and $a^T x \leq b$

e.g., $Ax = b \iff Ax \leq b$ and $(-A)x \leq (-b)$

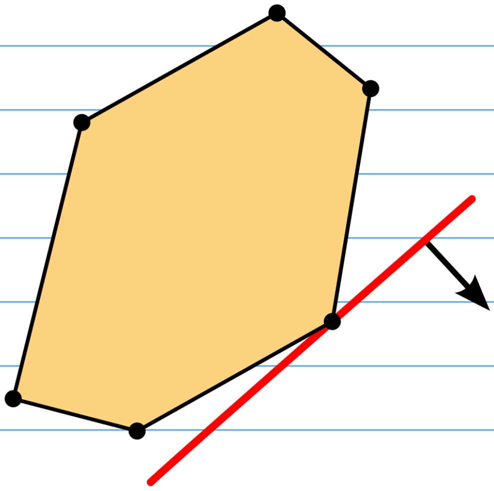
$A \in \mathbb{R}^{m \times n}$

$x \in \mathbb{R}^n$

$b \in \mathbb{R}^m$

$$\iff \underbrace{\begin{pmatrix} A \\ -A \end{pmatrix}}_{= A' \in \mathbb{R}^{2m \times n}} x \leq \underbrace{\begin{pmatrix} b \\ -b \end{pmatrix}}_{= b' \in \mathbb{R}^{2m}}$$

Integer Programming



$$\text{(IP)} \quad \max \; c^T x \quad \text{s.t.} \quad Ax \le b, \quad x \in \mathbb{Z}^n$$

($x \in \{0,1\}^n \Rightarrow$ binary programming
or 0-1 ")

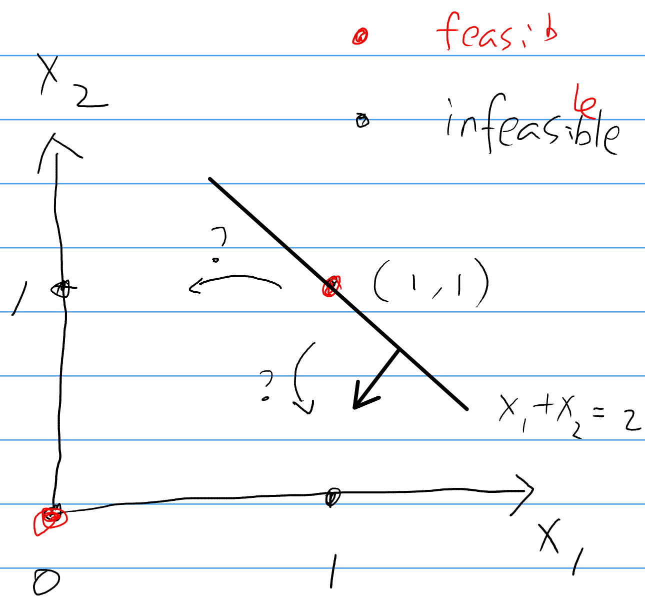
In generl, the problem becomes much difficult when limited to integer solutions.

Ex. $\min \; x_1 + x_2$

s.t. $2x_1 - 2x_2 \le 1$

$2x_1 - 2x_2 \ge -1$

$x_1, x_2 \in \{0, 1\}$



Suppose we start from (1,1). We know the descending direction
but we cannot find a feasible neighbor vertex from (1,1).

* In general, IP is NP-hard whereas LP is P-hard.

* P-hard problems can be solved in polynomial time (of the input size),
NP-hard problems are considered not (however, it has not been proved).

P vs NP

* Decision problems are those problems with answers YES or NO.

* P problems can be solved in polynomial time w.r.t. the input size.

  Ex1: Given a graph and two nodes s and t, is there a path connects s and t?

  Ex2: Decide if a given graph is an Euler graph.

  Ex3: Decide if an LP problem has a solution with objective value <= a given threshold.

* NP problems are those problems for which we can verify if a given solution is feasible in polynomial time w.r.t. the input size.

  Ex1: P problems belong to NP class.

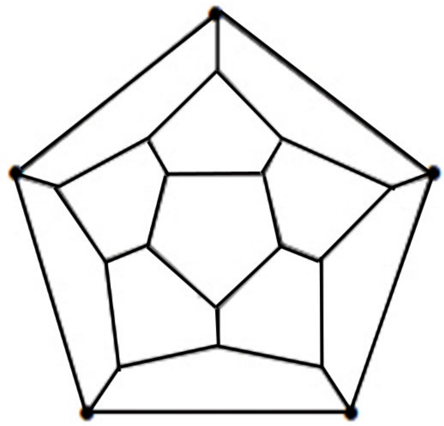
  Ex2: Decide if a given graph is a Hamilton graph (i.e., there exists a cycle that visits all nodes exactly once).

  Ex3: Decide if an IP problem has a solution with objective value <= a given threshold.

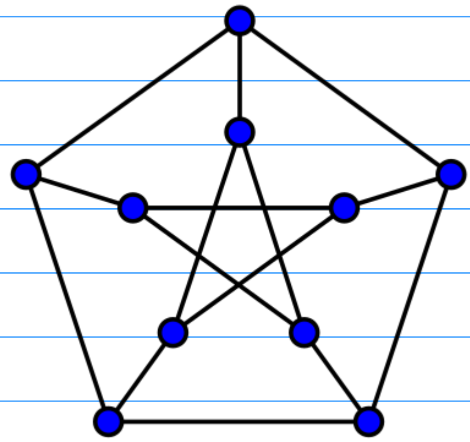
* P-hard or NP-hard problem is the optimization version of a P or an NP decision problem.

* NP-hard problems can be solved by binary search if its decision problem can be solved.

Whether P = NP is the biggest unsolved problem in CS.



https://www.geeksforgeeks.org/mathematics-euler-hamiltonian-paths/



Petersen graph