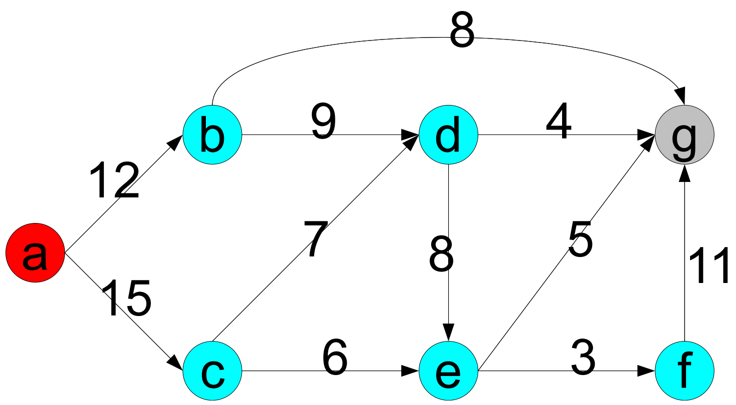
CS5321 Numerical Optimization Homework 6

Due May 21

1. (50%) Many of famous computer science problems can be written as a linear programming problem. Considering the following weighted directed graph, which has 7 nodes. Let node `a` be the *source* and node `g` be the *sink*.



   (a) Let the weights be capacity. Formulate the *maximum flow problem* as a linear programming problem. (Hint: capacity constraints and flow-balanced constraint.)

   (b) Let the weight be capacity. Formulate the *minimum cut problem* as a linear programming problem.

   (c) Show the problems in (b) and (c) are dual to each other.

   (d) Let the weights be distances. Formulate the *shortest path problem* as a linear programming problem.

   (e) What is the dual problem of the shortest path problem?

   (f) (Bonus 20%) Ignore the direction of edges. Formulate the *minimum spanning tree problem* as an integer linear programming problem.

2. (50%) Implement the simplex method for linear programming. The pseudo code is in Figure 1. The calling interface will be like

```
[x, case] = mysimplex(c, A, b, x0)
```

which solves

$$\begin{aligned} \min_{\vec{x}} \quad & \vec{c}^T\vec{x} \\ \text{subject to} \quad & A\vec{x} = \vec{b} \\ & \vec{x} \geq 0 \end{aligned}$$

The return value `case` should be 0, 1, or 2, which means (0) solved, (1) unbounded, (2) infeasible. You can assume $\vec{x}_0$ is a feasible point. Also, print out each $\vec{x}_i$ during the computation.

---

(1) Given a basic feasible point $\vec{x}_0$ and the corresponding index set $\mathcal{B}_0$ and $\mathcal{N}_0$.

(2) For $k = 0, 1, \ldots$

(3) Let $B_k = A(:, \mathcal{B}_k), N_k = A(:, \mathcal{N}_k)$, $\vec{x}_B = \vec{x}_k(\mathcal{B}_k), \vec{x}_N = \vec{x}_k(\mathcal{N}_k)$, and $\vec{c}_B = \vec{c}_k(\mathcal{B}_k), \vec{c}_N = \vec{c}_k(\mathcal{N}_k)$.

(4) Compute $\vec{s}_k = \vec{c}_N - N_k^T B_k^{-1} \vec{c}_B$ (pricing)

(5) If $\vec{s}_k \geq 0$, return the solution $\vec{x}_k$. (found optimal solution)

(6) Select $q_k \in \mathcal{N}_k$ such that $\vec{s}_k(i_q) < 0$, where i_q is the index of q_k in $\mathcal{N}_k$

(7) Compute $\vec{d}_k = B_k^{-1} A_k(:, q_k)$. (search direction)

(8) If $\vec{d}_k \leq 0$, return `unbounded`. (unbounded case)

(9) Compute $[\gamma_k, i_p] = \min\limits_{i, \vec{d}_k(i) > 0} \dfrac{\vec{x}_B(i)}{\vec{d}_k(i)}$ (ratio test)
(The first return value is the minimum ratio; the second return value is the index of the minimum ratio.)

(10) $x_{k+1}\begin{pmatrix} \mathcal{B} \\ \mathcal{N} \end{pmatrix} = \begin{pmatrix} \vec{x}_B \\ \vec{x}_N \end{pmatrix} + \gamma_k \begin{pmatrix} -\vec{d}_k \\ \vec{e}_{i_q} \end{pmatrix}$
($\vec{e}_{i_q} = (0, \ldots, 1, \ldots, 0)^T$ is a unit vector with i_qth element 1.)

(11) Let the i_pth element in $\mathcal{B}$ be p_k. (pivoting)
$\mathcal{B}_{k+1} = (\mathcal{B}_k - \{p_k\}) \cup \{q_k\}$, $\mathcal{N}_{k+1} = (\mathcal{N}_k - \{q_k\}) \cup \{p_k\}$

Figure 1: The simplex method for solving (minimization) linear programming