Mathematics for OR - Examples 2

1. Consider a minimum cost network flow problem in which we have a constraint on flows of $m_{ij} \le f_{ij} \le M_{ij}$ for every arc (i,j). Construct an equivalent network flow problem in which constraints are $0 \le \bar{f}_{ij} \le \bar{M}_{ij}$. Hint: Let $\bar{f}_{ij} = f_{ij} - m_{ij}$ and construct a new network for the arc flows $\bar{f}_{ij}$. How should b_i be changed?

2. Consider a transportation problem in which all shipping costs c_{ij} are positive. Suppose we increase the supply at some sources and increase the demands at some sinks. To maintain feasibility we ensure that the total supply remains equal to the total demand. Is it true that the value of the optimal cost will increase? Prove this, or provide a counterexample.

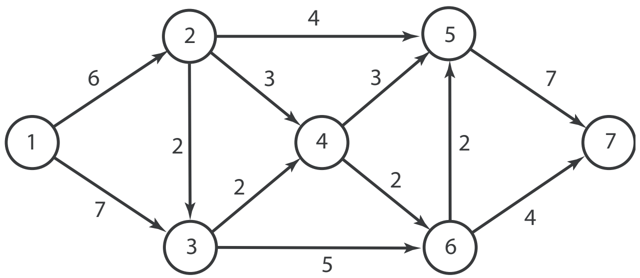
3. Each of n teams plays against each other team k games, and so $kn(n-1)/2$ games are played in all. Assume that every game ends in a win or a loss (no draws) and let x_i be the total number of wins of team i. Let X be the set of all possible outcome vectors $(x_1, \dots, x_n)$. Given an arbitrary vector $(x_1, \dots, x_n)$ we want to determine whether it belongs to X, that is, whether it is a possible tournament outcome vector. Find a network formulation of this problem such that the minimum cost flow attains a certain value if and only if $(x_1, \dots, x_n) \in X$.

4. Consider the following algorithm, applied to a graph with n nodes in which the distance between nodes i and j is c_{ij}.

1. Set $k = 1$. Arbitrarily choose a node, say 1, and let $N_1 = \{1\}$.
2. For each $j \notin N_k$, find the node $i \in N_k$ that minimizes c_{ij}. Call this node $v_k(j)$.
3. Let j_k^* be the node that minimizes $c_{j,v_k(j)}$ over $j \notin N_k$. Set $N_{k+1} = N_k \cup \{j_k^*\}$.
4. Stop if $k+1 = n$; otherwise set $k = k+1$ and goto step 1.

What problem does this algorithm solve? Show its running time is $O(n^2)$.

5. Find the maximal flow from node 1 to node 7 in the following network. The numbers by the arcs are their capacities.



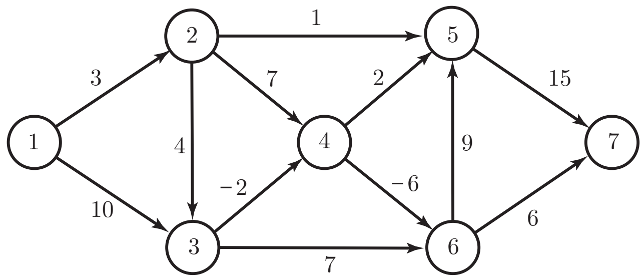
6. In the multicommodity flow problem there are $k \ge 1$ commodities, each of which has its own source s_i, sink t_i and demand D_i. We wish to simultaneously route D_i units of flow from s_i to t_i so that the total amount of commodities passing through any edge is no greater than its capacity. Define the max flow, as the maximum fraction ϕ such that we can simultaneously route flows of ϕD_i. Define the min cut as

$$\min_{U \subset N} \frac{C(U,\bar{U})}{D(U,\bar{U})}$$

where N is the set of nodes, U is a subset of the nodes, $C(U,\bar{U})$ is the sum of capacities passing from U to $\bar{U} = N \setminus U$, and $D(U,\bar{U})$ is the sum of demands D_i such that $s_i \in U$ and $t_i \in \bar{U}$. Show that max-flow is upper bounded by min-cut.

Consider the network of Question 5, with $k = 2$, $(s_1, t_1, D_1) = (1, 7, 11)$, $(s_2, t_2, D_2) = (2, 6, 3)$. Does min cut equal max flow?

7. Use the Bellman-Ford algorithm to find the least cost path from every node to node 7 in the following network. Note that some arcs have negative cost.



Use Dijkstra's algorithm on a modified network to solve the all-pairs problem.

8. Given a graph $G = (V, E)$ the minimum vertex cover problem is that of finding a smallest set of vertices such that every edge is incident to at least one vertex in this set. Show that this is the integer linear program

$$\text{minimize } \textstyle\sum_{i \in V} x_i, \text{ s.t. } x_i \in \{0,1\} \text{ and } x_i + x_j \ge 1 \text{ for all } (i,j) \in E.$$

Why is the linear programming relaxation in which we take $x_i \in [0,1]$ not likely to produce a good lower bound?

Formulate a semidefinite programming relaxation of the problem. Hint: recall the SDP relaxation for the max-cut problem in Lecture 12; take $x_i \in \{0,1\}$ as equivalent to $x_i^2 = x_i$, and use constraints of the form $x_i x_i = x_0 x_i$ for all $i \in V$, $(x_0 - x_i)(x_0 - x_j) = 0$, for all $(i,j) \in E$, and $x_0 x_0 = 1$.

9. A hiker wishes to choose items to take on a journey such that the total value of the items is at least 9, but the total weight is a minimum. Solve this problem using a branch and bound approach. Use as a lower bound for any node for which the total value of included items is less than 9, the total weight of included items plus the smallest weight amongst items not included. There are five items from which to choose, with values v_i and weights w_i.

i	1	2	3	4	5
v_i	5	5	4	2	3
w_i	5	6	3	1	2

10. Let $M = \{1, 2, \ldots, n\}$ be a set of n items and let $M_1, M_2, \ldots, M_m$ be a collection of m subsets of M. let F be a subset of $\{1, \ldots, m\}$. We say that a F is a **cover** if $\cup_{i \in F} M_i = M$ and say that F is a **packing** if $M_i \cap M_j = \emptyset$ for all $i, j \in F$. We are given a weight c_i for each M_i, and the weight of F is $\sum_{i \in F} c_i$. Let $A_{ij} = 1$ or 0 as M_i does or does not contain item j. Let x_i be 1 or 0 as i is or is not in F. Express as an integer linear programs the problems of finding: (a) the cover of minimum weight, and (b) the packing of maximum weight.

Show that the LP relaxation of (a) has the dual problem:

$$\text{(c): maximize } \sum_{j=1}^{n} y_j\,, \quad \text{subject to } \sum_{j=1}^{n} A_{ij} y_j \leq c_i, \quad \text{for all } i, \text{ and } y \geq 0.$$

Suppose all c_j are integers. Consider the problem

$$\text{(d): maximize } |S|\,, \quad \text{subject to } S \subseteq M \text{ and } |S \cap M_i| \leq c_i \text{ for all } i\,,$$

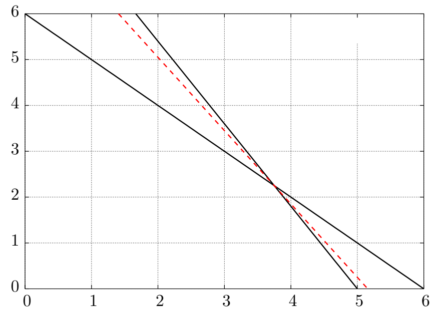
where $|S|$ denotes the number of items in S. Show that the solution value to (d) is never more than the solution value to (a). Can the inequality be strict?

Explain how (d) could be used in a branch and bound algorithm to solve (a).

11. Use Dakin's method to solve the following ILP.

$$\begin{aligned} \text{maximize } z = {} & 8x_1 + 5x_2 \\ \text{subject to } \quad & x_1 + \ \ x_2 \leq 6 \\ & 9x_1 + 5x_2 \leq 45 \\ & x_1, x_2 \geq 0 \\ & x_2, x_2 \text{ integer} \end{aligned}$$

Explain explain how Dakin's method will proceed by referring to the following plot. (You need not carry out iterations of the simplex method.)



12. Given these distances between 5 towns, find a minimum spanning tree.

	A	B	C	D	E
A		9	7	5	7
B	9		9	9	8
C	7	9		7	6
D	5	9	7		6
E	7	8	6	6	

By removing town B and finding a minimum spanning tree for the remaining towns, show that a lower bound on the solution to the travelling salesman problem for this graph is $9 + 8 + 17 = 34$.

Compare this to what you obtain by starting at town A and using a greedy algorithm (i.e. always visiting the closest town not yet visited).

13. Suppose A is a heuristic for the TSP. Let $A(I)$ be the length of the tour produced by A on an instance I and let $\mathrm{OPT}(I)$ be the length of the optimal tour. Suppose A is a ϵ-approximation algorithm, running in polynomial time, so that $\mathrm{A}(I) \leq (1 + \epsilon)\,\mathrm{OPT}(I)$, for all I.

Suppose we have a graph $G = (N, A)$, with n nodes, and want to decide if it contains a Hamiltonian cycle (i.e. a tour that visits every node exactly once). Let us construct an instance of TSP in which $c_{ij} = 1$ or $c_{ij} = (1 + \epsilon)n$ as G does or does not contain the edge (i, j). Show that, applied to this instance, $\mathrm{A}(I) \leq (1 + \epsilon)n$ if G has a Hamiltonian cycle, and $\mathrm{A}(I) \geq (n - 1) + (1 + \epsilon)n$ if G does not. Hence show that A can be used to decide the Hamiltonian cycle decision problem in polynomial time.

Given that the Hamiltonian cycle decision problem is $\mathcal{NP}$-complete, what do you conclude about the plausibility that there exists an algorithm like A? Why does this not invalidate the 1-approximation algorithm derived in Lecture 14?

Richard Weber 20/10/10