Department of Mathematical Sciences
CARNEGIE MELLON UNIVERSITY

OPERATIONS RESEARCH II 21-393

Homework 1: Due Monday September 7.

Q1 Solve the following knapsack problem:

$$\begin{array}{lcccccccc} \text{maximise} & 5x_1 & + & 8x_2 & + & 15x_3 & & \\ \text{subject to} & & & & & & & \\ & 3x_1 & + & 4x_2 & + & 5x_3 & \leq & 19 \end{array}$$

$$x_1, x_2, x_3 \geq 0 \text{ and integer.}$$

Solution

w	f_1	δ_1	f_2	δ_2	f_3	δ_3
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	5	1	5	0	5	0
4	5	1	8	1	8	0
5	5	1	8	1	15	1
6	10	1	10	0	15	1
7	10	1	13	1	15	1
8	10	1	16	1	20	1
9	15	1	16	1	23	1
10	15	1	18	1	30	1
11	15	1	21	1	30	1
12	20	1	24	1	30	1
13	20	1	24	1	35	1
14	20	1	26	1	38	1
15	25	1	29	1	45	1
16	25	1	32	1	45	1
17	25	1	32	1	45	1
18	30	1	34	1	50	1
19	30	1	37	1	53	1

Solution: $x_1 = 0, x_2 = 0, x_3 = 3$. Maximum $= 39$.
Start with $x_1 = x_2 = x_3 = 0$. $\delta_3(16) = 1$ and so we add one to x_3. We have used up 5 units of the knapsack. There are 11 units left. $\delta_3(11) = 1$ and so we add one to x_3. We use up another 5 units and so we are left with 5. $\delta_3(6) = 1$. We add one more to x_3. There are now 1 units in the knapsack. $\delta_3(1) = 0$ and so we move to column 2. $\delta_2(1) = 0$ and so we move to column 1. $\delta(1) = 0$ and we are done.

Q2 Consider a 2-D map with a horizontal river passing through its center. There are n cities on the southern bank with x-coordinates $a(1)...a(n)$ and n cities on the northern bank with x-coordinates $b(1)...b(n)$. You want to connect as many north-south pairs of cities as possible with bridges such that no two bridges cross. When connecting cities, you can only connect city i on the northern bank to city i on the southern bank. Construct a Dynamic Programming solution to this problem. (You can assume that $a(1) < a(2) < \cdots < a(n)$, but you **cannot** assume that $b(1) < b(2) < \cdots < b(n)$. If both sequences are increasing, then the problem is trivial).
Solution: Let $f(j)$ be the maximum number of bridges choosable if we only use $(a(i), b(i),\ i \geq j)$. Then

$$f(j) = \max \begin{cases} f(j+1) & \text{do not choose } (a(j), b(j)) \\ 1 + f(\min\{k > j :\ b(k) > b(j)\}) & \text{choose } (a(j), b(j)) \end{cases}.$$

Q3 We are given $2n$ sets $D_1, D_2, \ldots D_n$ and $R_1, R_2, \ldots, R_n$ where n is even. Also, $|D_i| + |R_i| = m$ for $i = 1, 2, \ldots, n$. Find an algorithm that will check to see if the following is possible: Find a set $I \subseteq [n]$, $|I| = n/2$ such that

$$\sum_{i \in I} |D_i| \geq \sum_{i \in I} |R_i| \text{ and } \sum_{i \notin I} |D_i| \geq \sum_{i \notin I} |R_i|.$$

Your algorithm should run in time polynomial in m, n.
Solution: For $I \subseteq [n]$ let $D_I = \sum_{i \in I} |D_i|$ and $R_I = \sum_{i \in I} |R_i|$. Then let

$$f_{k,\ell}(a, b, c, d) = \begin{cases} 1 & \exists I \subseteq [k] : |I| = \ell \text{ and } D_I = a, D_{[k] \setminus I} = b, R_I = c, R_{[k] \setminus I} = d \\ 0 & otherwise \end{cases}$$

Then we have the recurrence

$$f_{k+1,\ell}(a,b,c,d) =$$
$$\begin{cases} 1 & f_{k,\ell-1}(a-|D_{k+1}|,b,c-|R_{k+1}|,d)+f_{k,\ell}(a,b-|D_{k+1}|,c,d-|R_{k+1}|)\geq 1 \\ 0 & otherwise \end{cases}$$

Having computed $f_{n,n/2}$ we can check to see whether or not there exist $a \geq c, b \geq d$ such that $f_{n,n/2}(a,b,c,d) = 1$. This takes $O((mn)^4 n^2)$ time, since this is the number of function evaluations we need to compute. Note that $D_{[n]} + R_{[n]} = mn$. (We can save a bit of time by only evaluating $f_{k,\ell}$ when $a+b+c+d = km$.)