Computer Science C73 — December 6, 2021
Scarborough Campus — University of Toronto

Solutions for Homework Assignment #8

Answer to Question 1.

a. We construct a flow network with graph $G = (V, E)$, source and sink nodes, and edge capacities as follows. The node set V of the graph consists of:

- A source s and a sink t;
- For each subscriber i, three nodes labeled S_i, F_i, and N_i. Intuitively, node S_i represents the books borrowed by subscriber i; these are split into the fiction and nonfiction books borrowed by the subscriber, represented by F_i and N_i respectively.
- For each book j, a node T_j.

The edge set E of the graph and their capacities are:

- For each subscriber i, an edge (s, S_i) with capacity 12 (the maximum number of books that the subscriber may borrow).
- For each subscriber i, edges (S_i, F_i) and (S_i, N_i), each with capacity 8 each (the maximum number of function and nonfiction books that the subscriber may borrow).
- For each subscriber i and each book j such that $j \in B_i$ (i.e., j is a book that subscriber i is interested in borrowing) and j is a fiction (respectively, nonfiction) book, an edge (F_i, T_j) (respectively, (N_i, T_j)) with capacity 1. This represents the possibility that subsriber i borrows a copy of book j.
- For each book j, an edge (T_j, t) with capacity b_j (the number of copies of book j that the library owns).

There is a natural correspondence between feasible assignments on one hand and integral flows in the above flow network on the other: Given a feasible assignment, we define a flow as follows: If subscriber i is is assigned to borrow k books, put a flow of k on the edge (s, S_i); if k' of these are fiction books, put a flow of k' on edge (S_i, F_i) and a flow of $k - k'$ on edge (S_i, N_i). If subscriber i is assigned to borrow fiction (respectively, nonfiction) book j, put a flow of 1 on edge (F_i, T_j) (respectively (N_i, T_j)). If a total of ℓ subscribers are assigned to borrow copies of book j, put of flow of ℓ on edge (T_j, t). It is easy to verify that this is indeed a flow (i.e., it satisfies the capacity and conservation constraints). Conversely, from an integral flow, we can obtain a feasible assignment by interpreting each unit of flow along an s-to-t path going through nodes S_i and T_j as meaning that subscriber i is assigned to borrow a copy of book j.

Thus, a maximum flow in this graph corresponds to a feasible assignment of subscribers to books that maximizes the number of books that subscribers are assigned to borrow, and hence maximizes the library's revenue from government subsidies. The algorithm then is as follows:

```
1  use input to construct flow network 𝓕 as described above
2  f := MaxFlow(𝓕)
3  A := {(i, j) : f(F_i, T_j) > 0 or f(N_i, T_j) > 0}
4  return A
```

It is easy to see that the running time of this algorithm is dominated by the time required for line 2. Let $k = |S|$ and $\ell = |B|$. The graph of the flow network has at most $k\ell + k + \ell = O(k\ell)$ edges, and the sum of the capacities of the edges out of s is $12k$. So the running time of the algorithm is $O(k^2\ell)$.

b. The tricky part here is to express the objective function. Let $B = F \cup N$.

We introduce variables x_{ij}, for each subscriber $i \in S$ and book $j \in B$. The intention is that, in a feasible solution of the 0-1 LP,

$$x_{ij} = \begin{cases} 1, & \text{if subscriber } s_i \text{ borrows book } j \\ 0, & \text{otherwise} \end{cases}$$

We start with the statement of the constraints that any assignment of subscribers to books must satisfy, according to the specification of the problem.

- For every $i \in S$ and $j \in B$, add the nonlinear constraint $x_{ij} \in \{0, 1\}$.
- For every $i \in S$ and $j \in B$, if $j \notin B_i$, add the constraint $x_{ij} = 0$. (Do not lend to a subscriber a book in which he/she is not interested.)
- For every $i \in S$, add the constraint $\sum_{j \in B} x_{ij} \leq 12$. (Each subscriber can borrow at most 12 books.)
- For every $i \in S$, add the constraints $\sum_{j \in F_i} x_{ij} \leq 8$ and $\sum_{j \in N_i} x_{ij} \leq 8$. (Each subscriber can borrow at most 8 fiction and at most 8 nonfiction books.)
- For every $j \in B$, add the constraint $\sum_{i \in S} x_{ij} \leq b_j$. (The number of copies of a book borrowed does not exceed the number of copies of that book that the library owns.)

To define the objective function, we introduce a variable y_i for every subscriber i. The intention is that

$$y_i = \begin{cases} 1, & \text{if subscriber } i \text{ has borrowed at least 10 books} \\ 0, & \text{otherwise} \end{cases}$$

The objective function then becomes:

- Maximize $\sum_{i \in S} y_i$.

To ensure that the variables y_i have the intended semantics, we introduce the following constraints:

- For every $i \in S$, add the constraint $y_i \leq \frac{1}{10} \sum_{i=1}^{m} x_{ij}$, and the nonlinear constraint $y_i \in \{0, 1\}$.

Notice that, by themselves, these constraints do not ensure the intended semantics for y_i. For example, we could satisfy them by setting all y_is to 0. However, since the objective function is to ***maximize*** the sum of y_is, an optimal solution will assign to each y_i the maximum possible value consistent with the constraint that y_i is at most one-tenth of the number of borrowed by i. Since all variables are restricted to take on values 0 or 1, y_i can be set to 1 only if subscriber i has borrowed (at least) ten books.

Answer to Question 2.

a. From the input $A = \{a_1, \ldots, a_n\}$, b of the problem we construct the following 0-1 linear program:

Variables. x_i for every i such that $1 \leq i \leq n$. The intention is that $x_i = 1$ of element a_i is selected for the subset S, and $x_i = 0$ otherwise.

Objective function. maximize $\sum_{i=1}^{n} a_i x_i$.

Constraints.

- $\sum_{i=1}^{n} a_i x_i \leq b$.
- $x_i \in \{0, 1\}$ (nonlinear constraint).

b. For any $c > 1$, consider the input with $a_1 = 1, a_2 = c + 1$, and $b = c + 1$. Then the greedy algorithm will return the set $\{a_1\}$ with sum 1, while the optimal subset is $\{a_2\}$ with sum $c + 1$. So, for this input the sum of the elements of the set returned by the greedy algorithm is less than c times the sum of the elements of the optimal set.

c. The algorithm is the same as the bad GreedySum algorithm of part (b), with the proviso that we sort the elements of A in decreasing order (largest first).

```
GreedySumSorted(A, b)
sort A in decreasing order so a_1 > a_2 > ... a_n
X := ∅
total := 0
for i := 1..n do
    if total + a_i ≤ b then
        X := X ∪ {a_i}
        total := total + a_i
return S
```

By the conditional in line 5, it is obvious that the algorithm outputs a set X whose sum of elements is at most b. Let $X = \{x_1, \ldots, x_\ell\}$, where $x_1 > \ldots > x_\ell$, and $Y = \{y_1, \ldots, y_m\}$ be an optimal set, where $y_1 > \ldots > y_m$.

Note that the first element that our greedy algorithm adds to X is the largest integer in A that is at most b; therefore $x_1 \geq y_1$. Continue down the list of elements in X and Y as long as the element of X is greater than or equal to the corresponding element of Y, until we either run out of elements in X or the next element x_{k+1} of X is less than the corresponding element y_{k+1} of Y. (Note that we cannot run out of elements in Y before this happens because otherwise the sum of the elements in X is greater than the sum of the elements in Y, which is not possible since Y is optimal.)

To simplify notation, let $X^t = \sum_{i=1}^{t} x_i$ (i.e., the sum of the first t elements of X) and $Y^t = \sum_{i=1}^{t} y_i$. So the sum of all elements of X is X^ℓ and the sum of all elements of Y is Y^m. We want to prove that $X^\ell > \frac{1}{2}Y^m$. We have:

(A) $X^\ell \geq X^k$, since $\ell \geq k$.

(B) $X^k + y_{k+1} > b$. This is because, by definition of k, $y_{k+1} > x_{k+1}$ (or x_{k+1} does not exist because $k = \ell$). Thus our greedy algorithm considers y_{k+1} before x_{k+1} (or before terminating, if x_{k+1} does not exist). If, contrary to our claim, $X_k + y_{k+1} \leq b$, the algorithm would have added y_{k+1} and not x_{k+1} (or nothing) as the $(k+1)$-th element of X.

(C) $Y^m \leq b$, since the sum of all elements of Y does not exceed b.

(D) $Y^m < X^k + y_{k+1}$, by (B) and (C).

(E) $y_{k+1} < X^k$, since $y_{k+1} < y_k \leq x_k \leq X^k$, where the first inequality is by the ordering of the elements of Y, the second inequality is by definition of k, and the third inequality is by definition of X^k.

(F) $Y^m < 2X^k$, by (D) and (E).

By (A) and (F), we have $X^\ell / Y^m > X^k / 2X^k$ and so $X^\ell > \frac{1}{2}Y^m$, as wanted.

THAT'S IT WITH HOMEWORK SOLUTIONS, FOLKS!