Combinatorial optimization
Exercise sheet 3

Solutions by: Anjana E Jeevanand and David Čadež

1. November 2023

Exercise 3.2.

i) On other pages

ii) On other pages

iii) On other pages

iv) From part (iii) we know that lengths of P_i will not decrease. Suppose $j = i + 1$, so they are consecutive. If they were not disjoint then, again using part (iii), $|E(P_j)| > |E(P_i)|$.

Let now $j > i$ and $|E(P_i)| = |E(P_j)|$. Then, since lengths in the sequence do not decrease,

$$|E(P_i)| = |E(P_{i+1})| = \cdots = |E(P_j)|.$$

Suppose there exists a path P_k for some $k \in \{i + 1, \ldots, j\}$ such that $E(P_i), E(P_{i+1}), \ldots, E(P_k)$ are not pairwise disjoint. Take such k to be minimal. Since $E(P_i), E(P_{i+1}), \ldots, E(P_{k-1})$ are pairwise disjoint, there exists $l \in \{i, \ldots, k-1\}$ such that P_l and P_k are not disjoint. Since we can switch the order of augmentation if the M-augmenting paths are vertex-disjoint, we can simply augment over all paths $P_i, \ldots, P_{k-1}$ except on P_l. Then we are again in the same situation as in part (iii) of this exercise and we get $|E(P_k)| > |E(P_l)|$. This is contradiction with assumption that such a path P_k exists.

v) From part (ii) we know that at each step of the sequence $P_1, \ldots, P_{\nu(G)}$ we will have a path that is at most of length $\frac{\nu(G)+|M|}{\nu(G)-|M|}$ where $|M|$ is the size of the matching before augmenting. So if we have done i augmentations up to now, the set $|M|$ has size i. And thus there exists an augmentation path of length at most $\frac{\nu(G)+i}{\nu(G)-i}$. Since the lengths of these paths strictly increase, all of the paths up to this point must have been at most that length. And assume "worst case" where all paths after i-th will be of pairwise different lengths. Also keep in mind that all M-augmenting paths have odd lengths. The following is then the upper limit for number of possible different numbers

$$\left\lfloor \frac{1}{2} \left\lfloor \frac{\nu(G)+i}{\nu(G)-i} \right\rfloor \right\rfloor + \nu(G) - i$$

Since this holds for any $i \in \{1, \ldots, \nu(G)\}$, we should look at which i this bound is tightest. Write $w := \nu(G)$. To find that value it is enough to try to minimize

$$\frac{w+i}{w-i} - 2i + 2 = \frac{w+i-(2i-2)(w-i)}{w-i} = \frac{w+i-2wi+2i^2-2i+2w}{w-i}$$

We can pretend $i \in \mathbb{R}$ and expression continuous, so we can take the derivative. We get

$$\frac{(4i-1-2w)(w-i)+3w-i-2wi+2i^2}{\ldots} = \frac{-2i^2+4iw+2w-2w^2}{\ldots}$$

Look at the zeros of the numerator. It is a polynomial of 2nd degree, so they are

$$i_{1,2} = \frac{2w \pm \sqrt{4w^2-4(w^2-w)}}{2} = w \pm \sqrt{w}$$

Since $i \leq w$ and $i \in \mathbb{N}$, we take an integer close to $w - \sqrt{w}$. Lets try $i = \lceil w - \sqrt{w} \rceil$. So the number of different integers in the sequence $|E(P_i)|_{i=1,\ldots,\nu(G)}$ is at most

$$\left\lfloor \frac{1}{2} \left\lfloor \frac{2w + \lceil -\sqrt{w} \rceil}{\lceil \sqrt{w} \rceil} \right\rfloor \right\rfloor + w - \lceil w - \sqrt{w} \rceil$$

Write $\sqrt{w} = u + d$, where $u = \lfloor \sqrt{w} \rfloor$. Then this simplifies to

$$\begin{aligned}
\left\lfloor \frac{1}{2} \left\lfloor \frac{2(u^2+2du+d^2)-u}{u} \right\rfloor \right\rfloor + u = \\
&= \left\lfloor \frac{1}{2} \left\lfloor 2u + 4d + \frac{2d^2}{u} - 1 \right\rfloor \right\rfloor + u \\
&= 2u + \left\lfloor \frac{1}{2} \left\lfloor 4d + \frac{2d^2}{u} \right\rfloor - \frac{1}{2} \right\rfloor \\
&= 2\sqrt{w} - 2d + \left\lfloor \frac{1}{2} \left\lfloor 4d + \frac{2d^2}{u} \right\rfloor - \frac{1}{2} \right\rfloor \\
&\leq 2\sqrt{w} - 2d + \left\lfloor \frac{1}{2} \left\lfloor 4 + \frac{2}{u} \right\rfloor - \frac{1}{2} \right\rfloor \\
&\leq 2\sqrt{w} - 2d + 1 \\
&< 2\sqrt{w} + 1
\end{aligned}$$

Which proves there is less than $2\sqrt{\nu(G)} + 1$ different numbers in the sequence $|E(P_i)|_{i=1,\ldots,\nu(G)}$.

vi) Denote $G = (A \dot\cup B, E)$.

First note that every M-augmenting path is odd, so it has one ending in A and other in B. Using that remark the algorithm will start at unmatched vertices in A and construct a directed graph from which we will be able to extract shortest M-augmenting paths.

The algorithm is out of two parts:

(a) BFS starting from unmatched vertices in A and searching for unmatched vertices in B, and thus creating some directed graph

(b) Many DFS-es starting at the leaves of a directed graph mentioned above and searching for unmatches vertices in A. We will keep track of visited vertices to obtain a family of disjoint shortest M-augmenting paths.

Let τ be a function on vertices $V(G)$ such that $\tau(v) = v$ for $v \in V(G)$ unmatched and $\tau(v) = u$ if $\{u, v\} \in M$.

We define the following algorithm:

```
I = unmatched vertices
P = ∅                                  ▷ used to store beginnings of paths
μ ≡ ∅                                  ▷ edges of a directed graph, as a function
π ≡ ∞                                  ▷ function on vertices
for v ∈ A ∩ I do
    π(v) = 0
end for
queue q = A ∩ I                        ▷ FIFO queue
l_min = ∞
while q not empty & π(q.top()) < l_min do
    v := q.pop()
    if v ∈ A then
        for u ∈ N(v) \ {τ(v)} do
            if π(u) > i + 1 then
                μ(u) := {v}
                π(u) := i + 1
                q.add(u)
            else if π(u) = i + 1 then
                μ(u) := μ(u) ∪ {v}
            end if
        end for
    else if τ(v) ≠ v then
        μ(τ(v)) := v
        π(τ(v)) := i + 1
        q.add(τ(v))
    else
        P.add(v)
        l_min := π(v)
```

end if
end while
return P, μ

Description: The algorithm is a modified BFS. It keeps stored:

- distance (length of shortest M-alternating path) of each visited element to the nearest element of $I \cap A$
- directed edges going along shortest M-alternating paths towards elements of $I \cap A$

At the end it returns P, which are the elements of $I \cap B$ that are in the union of shortest M-augmenting paths. If P is empty, then $I \cap B$ is not reachable from any vertex in $I \cap A$ with M-augmenting paths. This means matching M is maximum matching (using Berge theorem).

The time complexity of this algorithm is clearly $\mathcal{O}(m+n)$, because BFS is $\mathcal{O}(m+n)$ and we do only a few constant operations at each vertex.

Denote the directed graph $\overline{G} = (V(G), \{v \to u \mid v \in V(G), u \in \mu(v)\})$. This directed graph is clearly acyclic since for each edge $v \to u$ we have

$$\pi(v) = \pi(u) + 1. \tag{1}$$

All paths in $\overline{G}$ starting in an edge $v \in P$ and ending in an edge $u \in I \cap A$ have the same length, which follows from 1.

Note that two paths in $\overline{G}$ are vertex-disjoint exactly when their endpoints are different and they are edge-disjoint. This follows from the fact that a path is M-alternating and if they share a vertex, they must also share the edge matching this vertex (except endpoints, which we assumed separately). Using this remark it is enough to remove edge on that paths in the algorithm.

On graph $\overline{G}$ we run DFS multiple times. In each iteration we find an (shortest) M-augmenting path and remove all vertices from the graph before next iteration. That way we find some maximal family of M-augmenting paths from $I \cap B$ to $I \cap A$.

Sketch of an algorithm:

$\mathcal{P} = \emptyset$
for $v \in P$ **do**
  Run DFS in $\overline{G}$ starting in v and searching for any element in $I \cap A$.
  if path was found from P to $I \cap A$ **then**
    Add path to $\mathcal{P}$
    Remove all vertices in $\overline{G}$ that lie on that path
  end if
end for

Time complexity of this algorithm is also clearly $\mathcal{O}(m+n)$ because DFS is. If we run both parts one after the other the time complexity is also $\mathcal{O}(m+n)$.

Lets prove this family $\mathcal{P}$ satisfies the condition in the exercise. Suppose we augmented M using all paths in $\mathcal{P}$. Denote the augmented matching M'. Since all disjoint paths are disjoint, it does not matter in which order we do the augmentating. Suppose there exists an M'-augmenting path P' for which $|E(P')| \leq l_{min}$. Separate cases

- If it shared a vertex with some path $P \in If\mathcal{P}$, we can use part 3 of this exercise to prove on matching $M \triangle (\cup\mathcal{P} \setminus \{P\})$ that $|E(P')| > |E(P)| = l_{min}$.
- If it is disjoint with all paths in the family $\mathcal{P}$, then P' would be contained in the graph $\overline{G}$. Let $v' \in P' \cap I \cap B$. Then in the for loop when we pick v' we would have found an M-augmenting path that starts in v'. Therefore $\mathcal{P}$ would contain a path that is not disjoint with P'.

Therefore such a path cannot exist and thus $\mathcal{P}$ satisfies inequality in the exercise.

vii) Using previous parts of the exercise we can construct an algorithm that finds a maximum matching of a bipartite graph in time complexity $\mathcal{O}(\sqrt{n}(m+n))$.

We actually just use the algorithm described above. Specificaly:

- Run algorithm above and get disjoint shortest M-augmenting paths.
- Augment over those paths.
- Go back to first step.

From part (v) of this exercise we know we will have to iterate at most $2\sqrt{\nu(G)}+1$ times. Using $2\nu(G) \leq n$ we quickly get that all together would be in the class $\mathcal{O}(\sqrt{n}(n+m))$.