Homework 4 Exercise 1

Oscar Garnier

9 janvier 2023

For simplicity, we will assimilate the notations $C, E(C)$ and $V(C)$ when no ambiguity comes of it.

Question 1

C contains an even number of edges, since half of them are supposed to be in M.
Let $u \in V$. Since M is a perfect matching in G, $\exists! e \in E$ s.t. $e \in M$ and $u \in e$.

1. If $e \notin C$ then $e \in M \Delta E(C)$ and u is still uniquly matched in M' in M'
2. If $e \in C$, then $\exists! e' \in C$ s.t. $u \in e'$, so e and e' are adjacent, which means $e' \notin M$. Hence $e \notin M'$ and $e' \in M'$. So u is still uniquly matched in M'.

Question 2

Let us denote $c(M) = \sum_{e \in M} c(e)$. Notice that, given M a matching and C an alternating cycle, $c(M' = M\Delta C) = c(M) - c(C \cap M) + c(C/M)$. Let M be a minimum cost perfect matching, and C be an M-alternating cycle in G. Since M is of minimal weight, $c(M' = M\Delta C) \geq c(M)$. Hence

$$c(M) - c(C \cap M) + c(C/M) \geq c(M) \Rightarrow c(C \cap M) \leq c(C/M)$$

So C is not negative. G has no negative M-alternating cycles.

Let M be a perfect matching, and M' be some other perfect matching. Let us show that $C = M\Delta M'$ is a set of M-alternating cycles.
Let $u \in E$. $\exists! a \in M s.t. u \in a$ and $\exists! b \in M s.t. u \in b$. If $u \in V(C)$ (where $V(C)$ is defined as the set of endpoints of edges in C), this means $a \neq b$. So u is the endpoint of exactly one edge from M and one other edge from M'. By iterating this proprety, u is part of an M-alternating cycle. There are then as many of these as there are connexe components in C.

Now suppose G has no negative weight M-alternating cycles. Since $C = M\Delta M', M' = M\Delta C$, hence $c(M') = c(M) + c(C/M) - c(C \cap M)$. And since C is composed of M-alternating cycles, each non-negative, $\forall C_i$ connected component of C, $c(C_i/M) - c(C_i \cap M) \geq 0$, by summing these inequalities we get $c(C/M) - c(C \cap M) \geq 0$, hence $c(M') \geq c(M)$. M is in fact a minimal cost perfect matching.

Question 3

If M is not a minimum cost perfect matching, according to the previous question, there exists a negative M-alternating cycle C in G. But

$$c(C \cap M) = \sum (x, y) \in C \cap M c((x, y)) = \sum_{(x,y) \in C \cap M} p(x) + p(y) = \sum_{x \in V(C)} p(x)$$

. On the other hand

$$c(C/M) = \sum_{(x,y) \in C/M} c((x, y)) \geq \sum_{x \in V(C)} p(x)$$

This means the cycle is non-negative, which is absurd.
So M is a minimal cost matching.

Question 4

P starts and ends with an unmatched edge, since u and v are not covered by M, so there are k matched edges and $k+1$ unmatched edges in P. $M \Delta P$ removes the k edges from M and adds the $k+1$ unmatched ones, so it does increase the size of M by 1. For every edge $e \in P/M$, the endpoints are either

1. The endpoints of P, so unmatched
2. The endpoints of the adjacent edges in P , which are matched since P is alternating. This means that both these vertices are endpoints of no other edge in M (they can only be matched once), and the matched edges they were endpoints of is removed from M, so e is a valid addition to the matching.

As far as the price function p, it has not changed, so clause $(P1)$ is still verified, and $\forall e = (x, y) \in M'/M, e \in P$, so $c(e) = p(x) + p(y)$, as P is a perfect path. So clause $(P2)$ is still verified. p is a price function for M.

Question 5

As long as the current matching is not perfect, there exists an uncovered vertex u.

1. If $u \in L$ we have found r.
2. If $u \in R$, since there are only edges from R to L, every matched vertex in L is matched to a vertex in R. Hence, denoting L_m the matched vertices in L, $|L_m| = |R_m| < \frac{n}{2}$. So there exists an unmatched vertex $r \in L$.

Question 6

I will prove the following invariant : at the end of while loops with i odd, there is an alternating path between r and every element in L_i (which is refered to as L_{i+1} during the loop).
Initialisation :

i = 1. Every element $u \in L_1$ is in it by definition because $(r, u) \in E$ (and $(r, u) \in E/M$ or r would be matched. So there in in fact an alternating path.
Heredity :
Let $i = 2k + 1$, Let $u \in L_i$, by definition of L_i, $\exists v \in L_{i-1}$ s.t. $(u, v) \in E$. By definition of $v \in L_{i-1}, \exists w \in L_{i-2}$ s.t. $(w, v) \in M$. This means v is matched and $(u, v) \notin M$. We know from the invariant that there is an alternating path P between r and every vertex in L_{2k-1}, particularly w. Appending these two edges to P we get P' alternating path between r and u, which concludes the proof.

We can only discover an unmatched vertex on a step ending with an odd i, since the other ones only discover vertices throught edges in M, so matched vertices. When such a vertex is discovered, there exists an alternating path from r to it, and they are both unmatched. Since we considered only tight edges, P is a good path.

Question 7

Every layer is defined by adding only vertices that are not in any of the other layers ; this means that they are all disjoint.

Let A be the union of the odd layers. If $x \in A$, the layers are disjoint so $x \notin S$. x is in some odd layer i, which means the layer $i - 1$ is in the tree, and by definition of the layer i, there is a tight edge between some vertex in $i - 1$ and x. SO $x \in N_{tight}(S)$. Hence $A \subset N_{tight}(S)$.

Reciproqually, let $x \in N_{tight}(S)$, this means $x \notin S$ and $\exists y \in S$ s.t. xy is tight. $y \in S$ so there exists a layer L_i s.t. $y \in Li$ with even i.
If $x \notin A$, this means that, when defining L_{i+1}, $x \notin \bigcup k \leq iL_k$ and $\exists x \in L_i$ s.t. $xy \in E$ and is tight, so $x \in L_{i+1}$, which is absurd. $x \in A$, so $N_{tight}(S) \subset A$

$A = N_{tight}(S)$

Question 8

— It is easy to proove by induction that all the even layers are in L and all the odd layers are in R, so $S \subset L$.
— The initial vertex r is uncovered, and part of the 0 layer, so S has at least one uncovered vertex.
— Let $x \in N_{tight}(S) = A$ and suppose it is not matched to a vertex in S. We know it is not unmatched overall, since the algorithm returned the tree rather than a good path. So it is matched to some vertex $y \notin S$, then $y \in L$ so $y \notin A$. Hence, since $x \in A$, $x \in L_i$ for some odd i, and all the conditions are met for $y \in L_{i+1}$, so $y \in S$, which is absurd.

This proves the 3 clauses for being a good set, and concludes the question.

Question 9

Every element in $N_{tight}(S)$ is matched to an element in S, and by propreties of matchings, we know that these elements are all distinct. Hence $|N_{tight}(S)| \leq |S \cap M|$. We also know that S contains at least one unmatched vertex, so $|N_{tight}(S)| < |S|$.

Question 10

By construction of the search tree, which looks throught all the tight edges in the graph, every tight edge connected to S is in A (this is the same as before, if it is not, it is in the following layer, which is absurd). Hence, all the edges considered are NOT tight ; $\forall xy \in E, x \in S, y \notin N_{tight}(S), c(xy) > p(x) - p(y)$. This proves $\alpha > 0$.

Question 11

Let $xy \in E$. If $x \in S$ or $y \in S$, we will consider $x \in S$ w.l.o.g. In this case, either :

— $y \in N_{tight}(S)$ and $c(xy) = p(x)+p(y)$. This means $p'(x)+p'(y) = p(x)+\alpha+p(y)-\alpha = c(xy)$. The edge stays tight, so if satisfies (P1), and if it was matched, (P2).

— $y \notin N_{tight}(S)$, which means $\alpha \leq c(xy) - p(x) - p(y)$. Hence $p'(x) + p'(y) = p(x) + \alpha + p(y) \leq c(xy)$. So the edge satisfies (P1).

If $x, y \notin S$ then either

— xy is not matched, then $p'(x) + p'(y) \leq p(x) + p(y) \leq c(xy)$ and (P1) is satisfied.

— xy is matched, in which case $x, y \notin N_{tight}(S)$ either (since they are all matched to S), so $p'(x) + p'(y) = p(x) + p(y) = c(xy)$ and both (P1) and (P2) are satisfied.

p' is in fact a price function.

Question 12

Every time the algorithm returns a search-tree path, the matching augments by 1. The matching is bounded in size by $|L|$, so, if this step is called enought, the algorithm will find a maximal matching and terminate. For this step not to be called a finite amout of times, the search for a good path algorithm would need to run infinite times, and only return a search tree (every time it returns a search tree-path, the matching increases).

But returning the search tree leads to a change in price function, which we have proven to be valid, and if we look at the quantity $\sum_{x \in V} p(x)$, it is strictly increasing at each call of *update the price function*

Note the following, with regard to the new price function :

$$\begin{aligned}\sum_{x \in V} p'(x) &= \sum_{x \in S} (p(x) + \alpha) + \sum_{x \in N_{tight}(S)} (p(x) - \alpha) + \sum_{x \in V/S/N_{tight}(S)} p(x) \\ &= \sum_{x \in V} p(x) + \alpha(|S| - |N_{tight}(S)|)\end{aligned}$$

But we showed that $|S| > |N_{tight}(S)|$, so $\Delta(p) = \sum_{x \in V} p'(x) - \sum_{x \in V} p(x) \geq \alpha$. Every weight in the graph is an integer, so, by induction, $\alpha \in \mathbb{N}^*$, which means $\Delta(p) \geq 1$

For a given graph G, matching M and valid price function p, define $A = \sum_{x \in V} p(x) + |M|$. Each run of the algorithm increases this quantity by at least 1.

During a run of the main while loop, the algoithm returns either :

— A good path P, we have shown that this path can be used to increase the matching by one edge, with p still begin a valid price function. Hence A increases by 1.

— A search tree. We have shown we can use this tree to define a new price function, which is still valid for the matching, we also showed that $\Delta(p) \geq 1$, so A increases by at least 1.

Lastly, $sum_{xy \in E} c(xy) \geq sum_{xy \in E} p(x) + p(y) \geq sum_{xy \in E} p(x) + sum_{xy \in E} p(y) \geq \sum_{z \in V} p(z)$ because there exists a perfect mathching, so every vertex has at least one edge connected to it. $|M|$ is bounded by $|L|$. This means A is bounded and increases by at least one on every loop of the algorithm, so the algorithm terminates. Since the end conditon is for M to be a perfect matching, it is correct.

Question 13

Let $n = |E|$ and $m = |V|$. Since α is defined so that p' is still a price function, but as a minimum, there exists $xy \in E, x \in S, y \notin N_{tight}(S)$ s.t. $c(xy) - p(x) - p(y) = \alpha$. This means, when updating the price function, an extra edge becomes tight. As we showed int he previous questions, tight edges of the graph stay tight. This means the *update the price function* step can only happen $O(m)$ times, every time, it will need to look throught $O(m)$ edges to find α.
Updating the matching will happen exactly once for every edge of the matching, or $O(n)$ times. Finding the search tree is, in the worst case, over all the vertices of the graph, so it's complexity is $O(n)$. Checking wether M is maximal is simply a cardinality question, so $O(1)$.

So the overall complexity of the algoritm is $O((n+m)^2)$. Since the graph has a perfect matching, $|E| \geq \frac{|V|}{2}$ so the complexity is $O(m^2)$.