HW1 Can't get enough Algorithms

MPRI 2.11.1

14.09.2016 - **Due on Wed. 21/09 before 16:15**

You are asked to complete the exercise marked with a $[\bigstar]$ and to send me your solutions at:
nicolas.schabanel@cnrs.fr
(or drop it in my mail box on the 4th floor) on **Wed. 21/09 before 16:15**.

■ **Exercise 1 (Christofides' Algorithm for Metric Travelling Salesman Problem (1976)).** Recall the 2-approximation for Metric TSP: 1) Compute a minimum spanning tree T, 2) root T in some node and output the tour corresponding to the nodes in depth-first search order. One can view the tour computed by this algorithm equivalently as follows: 2.a) double the edges of T, every node has now an even degree in T which is thus eulerian, then 2.b) output an eulerian tour of T while skipping the nodes already visited. Since the distances are metric, short-cutting the tour only decreases its length and the tour output has length at most twice the weight of the minimum spanning tree.

Now, instead of doubling the edges of T, we could try to make T eulerian by adding a *minimum weight* set of edges.

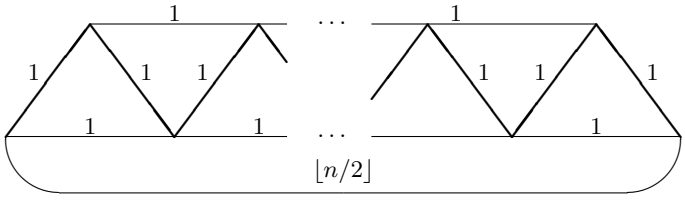
► ***Question* 1.1)** *Show that every graph has an even number of nodes of odd degree.*
▷ Hint. *What is the value of the sum of the degrees of its nodes?*

Now, let us consider the weighted subgraph H induced by the set of the nodes of odd degree in T and compute a minimum weight perfect matching M of H (using a classic polynomial-time algorithm). Then $T \cup M$ is eulerian and output an Eulerian tour of $T \cup M$ while skipping nodes that are visited twice or more.

► ***Question* 1.2)** *Show that the weight of M is at most* $\text{OPT}_{TSP}/2$.
▷ Hint. *Consider the projection of a TSP in H.*

► ***Question* 1.3)** *Conclude that this algorithm is a $3/2$-approximation for metric TSP.*

► ***Question* 1.4)** *Exhibit a critical family for this algorithm.*
▷ Hint. *Consider the metric completion of the following graph:*



■ **Exercise 2 (Cost-based approximation for Vertex Cover).** $[\bigstar]$ Recall the Weighted Vertex Cover problem: Given a undirected graph $G = (V, E, w)$ with weights $w : V \to \mathbb{R}_+$ on its vertices, find a set $C \subset V$ of minimum weight $w(C) = \sum_{v \in C} w_v$ that *covers* every edge in G, i.e. such that $(\forall uv \in E)\ u \in C$ or $v \in C$.

We consider the following algorithm. For each vertex v, a variable t_v is initialized to its weight, and when t_v drops to 0, v is picked in the cover. c_e is the *amount charged* to edge e.

Initialization:
• $C \leftarrow \emptyset$
• $\forall v \in V, t_v := w_v$
• $\forall e \in E, c_e := 0$
while C is not a vertex cover **do**
Pick an uncovered edge, say uv. Let $m = \min(t_u, t_v)$.
$t_u := t_u - m$
$t_v := t_v - m$
$c_{uv} := m$
Include in C all vertices having $t_v = 0$.
Output C.

► ***Question* 2.1)** *Show that this is a factor 2 approximation algorithm.*
▷ Hint. *Show that the total amount charged to edges is a lower bound on* OPT *and that the weight of cover C is at most twice the total amount charged to edges.*

► ***Question* 2.2)** *Exhibit a family of tight instances.*

■ **Exercise 3 (Maximum arc-disjoint paths problem).** Given a directed graph $G = (V, A)$ and k pairs of nodes $(s_1, t_1), \ldots, (s_k, t_k) \in V^2$, find a maximum subset $I \in \{1, \ldots, k\}$ that can satisfy simultaneously all the path requests in I, i.e., such that there exists a simple path P_i from s_i to t_i for each $i \in I$, such that the paths $(P_i)_{i \in I}$ are all arc-disjoint. This problem is not only NP-complete but hard to approximate as we will show below.

Let $n = |V|$ and $m = |A|$. We will first design a greedy $\frac{1}{2\sqrt{m}+1}$-approximation for this problem:

Algorithm 1 Greedy algorithm for the maximum arc-disjoint paths

$I := \varnothing$
while there is a path in G between some unserved pair in $\{1, \ldots, k\} \setminus I$ **do**
Let (s_i, t_i) be the unserved pair with the shortest path P_i from s_i to t_i.
Add i to I, select P_i to connect s_i to t_i, and delete from G the arcs used by P_i.
return $I, (P_i)_{i \in I}$.

Let I be the solution output by the greedy algorithm and I^* an optimal solution. We say that a path is *short* if it uses at most $\sqrt{m}$ arcs, and *long* otherwise. We denote by I_s and I_s^* the set of indices of pairs connected by short paths in I and I^* respectively.

► ***Question* 3.1)** *Show that the number of pairs connected by a long path in I^* is at most $\sqrt{m}$.*

► ***Question* 3.2)** *Show that $|I_s^* \setminus I_s| \leqslant |I_s|\sqrt{m}$.*

► ***Question* 3.3)** *Conclude that the greedy algorithm is a $\frac{1}{2\sqrt{m}+1}$-approximation.*

It has been proved that it is NP-complete for $k = 2$ pairs to decide whether it is possible to find two disjoint paths to connect both pairs or only one. Let us show that it implies that there is no $\Omega(m^{-\frac{1}{2}+\varepsilon})$-approximation for the general case for all $\varepsilon > 0$, unless $P = NP$. First note that since the greedy algorithm output at least one path (unless if the instance is degenerated and no pair is connected in G), the approximation ratio is always at least $1/k$. In order to find a bad instance to show that no $\Omega(m^{-\frac{1}{2}+\varepsilon})$-approximation exists, we need to consider $k = \Omega(m^{\frac{1}{2}-\varepsilon})$.

Here is the reduction: for every directed graph $G = (V, A)$, instance of the NP-complete two pairs problem, construct the graph G' as shown in Fig. 1 with $k = \lceil |A|^{1/\varepsilon} \rceil$ pairs $(a_1, b_1), \ldots, (a_k, b_k)$.

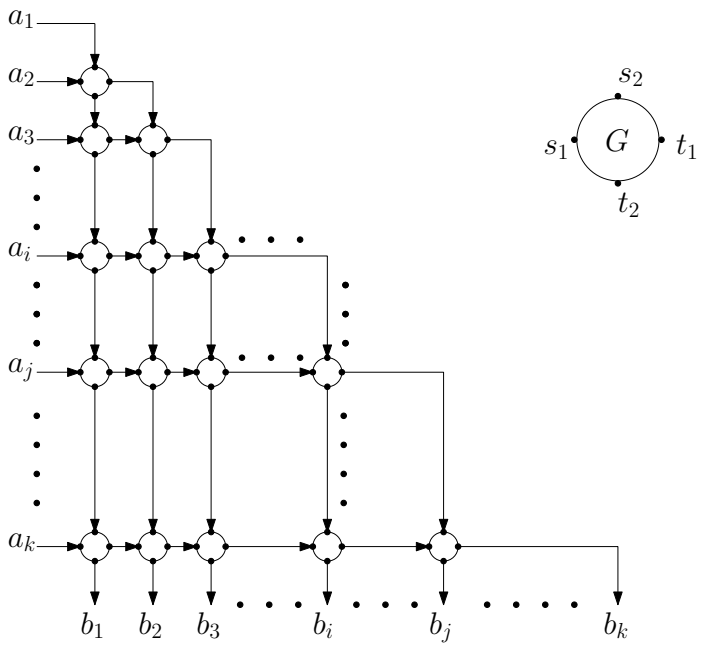


Figure 1: The graph G'.

► ***Question* 3.4)** *How many arcs has G' as a function of k? Prove that this reduction implies that if there is a $\Omega(m^{-\frac{1}{2}+\varepsilon})$-approximation for the maximum arc-disjoint paths problem with k pairs, then $P = NP$.*