University of Padova
Master's degree in Computer Science

Advanced Algorithms

Michele Scquizzato
scquizza@math.unipd.it

Last update: March 24, 2023

This document contains some examples of exam exercises (with solutions) for the *Advanced Algorithms* course. Exercises are organized in two parts: theory questions and problem solving.

1 Theory Questions

Question 1

Consider the following weighted graph, represented by an adjacency matrix where each numerical value represents the weight of the corresponding edge, and where the symbol '−' indicates the absence of the edge between the corresponding nodes.

	a	b	c	d	e	f
a	-	3	7	-	1	5
b		-	6	8	-	-
c			-	2	9	-
d				-	-	-
e					-	4
f						-

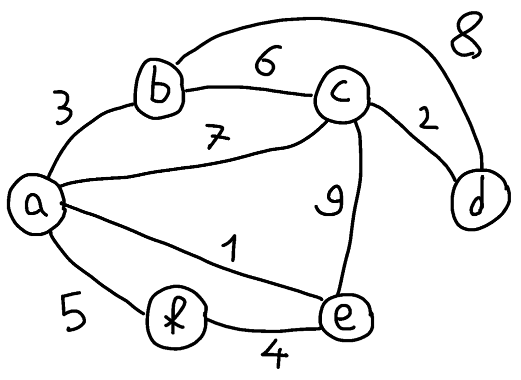
(a) Draw the graph.

(b) List the edges of the minimum spanning tree in the order they are selected by Kruskal's algorithm.

(c) List the edges of the minimum spanning tree in the order they are selected by Prim's algorithm starting at node a.

Solution:

(a)



(b) $(a, e), (c, d), (a, b), (e, f), (b, c)$.

(c) $(a, e), (a, b), (e, f), (b, c), (c, d)$.

Question 2

Does Dijkstra's algorithm work on graphs with negative weights? Motivate your answer.

Solution: No. Consider the following graph with 3 vertices a, b, c and with 3 arcs: (a, b) with weight 1, (a, c) with weight -1, and (b, c) with weight -3. If a is the source vertex, then Dijkstra's algorithm returns -1 as the length of a shortest path from a to c, which instead is -2.

Question 3

With reference to the problem of the minimum vertex cover, which in class we have 2-approximated by computing a maximal matching:

(a) Give the definition of vertex cover of a graph.

(b) Give the definition of matching in a graph.

(c) Find a maximal matching in the graph of Question 1.

Solution:

(a) A vertex cover of a graph is a set of vertices that includes at least one endpoint of every edge of the graph.

(b) A matching in a graph is a set of edges without common vertices.

(c) For example $\{(a, b), (c, d), (e, f)\}$ or $\{(a, c), (b, d), (e, f)\}$.

Question 4

For each of the following problems, say whether it is NP-hard or not and, if not, specify the complexity of the best algorithm seen in class.

(a) All-pairs shortest paths

(b) Connected components

(c) 3SAT

(d) Minimum spanning trees

(e) Metric TSP

Solution:

(a) $O(n^3)$

(b) $O(m + n)$

(c) NP-hard

(d) $O(m \log n)$

(e) NP-hard

Question 5

Show that a graph with n vertices and with a minimum cut of size t has at least $tn/2$ edges.

Solution: By definition of minimum cut we have $d(v) \geq t$ for any vertex $v \in V$, where $d(v)$ denotes the degree of v. Then, summing up over all the n vertices we obtain $\sum_{v \in V} d(v) \geq tn$. The claim follows by observing that $\sum_{v \in V} d(v) = 2m$.

2 Problem Solving

Problem 1

A *minimum bottleneck spanning tree* of a connected graph G is a spanning tree of G whose largest edge weight is minimum over all spanning trees of G.

(a) Prove that a minimum bottleneck spanning tree is not necessarily a minimum spanning tree.

(b) Prove that a spanning tree T which is *not* a minimum bottleneck spanning tree cannot be a minimum spanning tree. (Hint: focus on the edge of T of largest weight and try to replace it with an edge from some other suitable spanning tree...)

Solution:

(a) Consider a triangle with two edges of weight 2 and one edge of weight 1. There are three distinct spanning trees, each with largest edge weight 2, hence all three spanning trees are also a minimum bottleneck spanning tree; however, the tree with both edges of weight 2 is not a minimum spanning tree.

(b) Let e be the edge of largest weight in T, and let T_1 and T_2 be the two subtrees of T that remain should e be removed. Now consider a minimum bottleneck spanning tree T', and let e' be an edge of T' that connects T_1 and T_2; since T' is a minimum bottleneck spanning tree and T isn't, the weight of e' is smaller than the weight of e. Then, by replacing e with e' in T we obtain a new spanning tree with total weight smaller than the total weight of T, meaning that T cannot be a minimum spanning tree.

Problem 2

Given a set S of n integers and an additional integer t, assume that $\forall s \in S, 0 \leq s \leq t$. Consider the optimization problem where the set of feasible solutions is

$$\{S' \subseteq S \,\text{such that} \sum_{s \in S'} s \leq t\},$$

the cost of a feasible solution S' is $c(S') = \sum_{s \in S'} s$, and the goal is to compute the maximum cost among all the costs of the feasible solutions.

(a) Design a simple 2-approximation polynomial-time algorithm for this problem. (Hint: consider a *descending* ordering of the values in S, and then do a single pass over such values.)

(b) Prove that such algorithm is a 2-approximation algorithm.

Solution:

(a)
```
APPROX_SS(S,t)
{s_1,s_2,...,s_n} <- SORT-DECREASING(S)
sum = s_1
for i = 2 to n do
   if sum + s_i <= t then
      sum = sum + s_i
   else
      return sum
return sum
```

(b) First of all we observe that, since *sum* is initialized to $s_1 \leq t$ and a value s_i is added to *sum* only if $sum + s_i \leq t$, the returned value is always the cost of a feasible solution. If $s^\star$ denotes the maximum cost, we now need to prove that $s^\star/sum \leq 2$.

case 1 The algorithm returns out of the for loop: hence $sum = \sum_{s \in S} s \leq t$, that is $sum = s^\star$, and thus $s^\star/sum = 1 \leq 2$.

case 2 The algorithm returns from inside the for loop: hence there exists and index i' such that $sum + s_{i'} > t$. Observe that

$$s_{i'} < s_1 \leq sum,$$

and hence

$$2 \cdot sum > sum + s_{i'} > t,$$

that is

$$sum > \frac{t}{2} \geq \frac{s^\star}{2}.$$

Problem 3

Let $X_1, X_2, \ldots, X_n$ be independent indicator random variables such that $\Pr(X_i = 1) = 1/(4e)$. Let $X = \sum_{i=1}^n X_i$ and $\mu = E[X]$. By applying the following Chernoff bound, which holds for every $\delta > 0$,

$$\Pr(X > (1+\delta)\mu) < \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^\mu$$

prove that

$$\Pr(X > n/2) < \frac{1}{(\sqrt{2})^n}.$$

Solution: To apply the Chernoff bound we set $n/2$ equal to $(1+\delta)\mu$; since $\mu = E[X] = \sum_{i=1}^n E[X_i] = n/(4e)$, we get $\delta = 2e - 1$. Therefore

$$\begin{aligned}
\Pr(X > n/2) &= \Pr(X > (1 + 2e - 1)\mu) \\
&< \left(\frac{e^{2e-1}}{(2e)^{2e}}\right)^{n/(4e)} \\
&< 1/(2^{2e/4e})^n \\
&= \left(1/\sqrt{2}\right)^n.
\end{aligned}$$