CMPT404/705: Design and Analysis of Computing Algorithms

Take-Home Exam

Instructor: Igor Shinkar *Due date: April 21, 6:30PM*

- Write your name and SFU ID (numerical and alphabetical) at the top of your solution.
- Submit your solution as a pdf to Coursys before April 21, 6:30PM.
- You are encouraged to type your solutions in word/tex. If you write and scan your solutions, please make sure handwriting and the quality of the scans are readable.
- The exam consists of six problems, each worth 10 points. Choose any 4 problems out of 6, and to submit solutions to only these 4 problems. If you solve more than 4 problems, only the first four problems in your solution will be graded. No extra points for solving more problems.

- **This is an open book exam. You may use books, lecture notes, the internet**.
- You may use all algorithms and theorems we saw in class without proof.
- For solutions found on the internet, you should add the reference (no penalty).
  No matter how you find the solution, you need to explain all your answers.
  Provide enough details to convince me that you understand the solution.

- **You may not ask chegg.com, bartleby.com or similar websites for solutions.**
- **You may not discuss your solutions with other students.**

- Explain all your answers. Provide enough details to convince me that you understand the solution.

Good luck!

Question 1 (10 points)

(a) **[4 points]** *Design an algorithm with the following guarantees:*

Input: *An undirected connected graph $G = (V, E)$ such that the edges of G are colored either RED or BLUE.*

Goal: *Find a spanning tree of G that minimizes the number of RED edges in the tree.*

Running time : $\text{poly}(|V| + |E|)$.

(b) **[6 points]** *Design an algorithm with the following guarantees:*

Input: *an undirected connected graph $G = (V, E)$, where the edges of G are colored with either BLUE/RED/GREEN. It is guaranteed that*

- *If we take only the BLUE and the GREEN edges, then the graph G with these edges is connected.*
- *If we take only the RED and the GREEN edges, then the graph G with these edges is connected.*

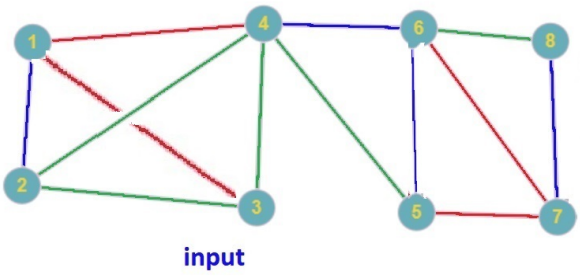
Goal: *Find the minimal possible number of edges from G such that in the remaining subgraph two conditions hold:*

(a) In the remaining subgraph the BLUE and GREEN edges form a connected subgraph spanning all vertices of G.

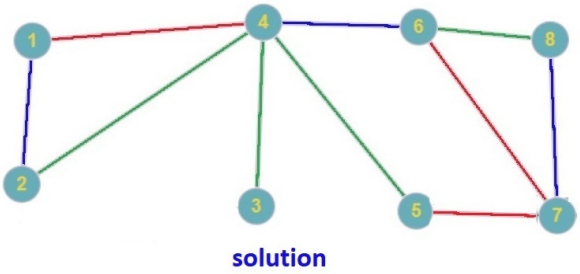
(b) In the remaining subgraph the RED and GREEN edges form a connected subgraph spanning all vertices of G.

Running time : $O(|V| + |E|)$.

The example below shows an input and a possible solution.



input



solution

Question 2 (10 points) *Design an algorithm with the following guarantees:*

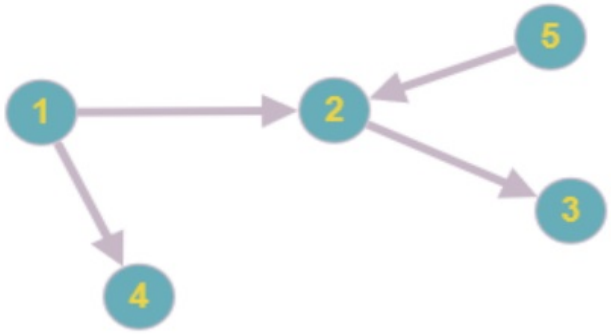
Input: *A directed acyclic graph* $G = (V, E)$.

Goal: *Find a subset* $S \subseteq V$ *of maximum size such that for any two vertices* $u, v \in S$ *the following holds: either there is a path from* u *to* v *or there is a path from* v *to* u.

**If more than one such set* S *exists, it suffices to return one of them.*

Running time : $O(|V| + |E|)$,

For example, in the example below you can output the set $\{1, 2, 3\}$ *or the set* $\{2, 5, 3\}$. *You cannot output* $\{1, 2, 5\}$ *because you cannot get from 1 to 5 or from 5 to 1.*



Question 3 (10 points) *There are n stones in a row, numbered $1, 2, \ldots, n$. For each $i \in \{1, \ldots, n\}$ the height of the i'th stone is h_i. A frog starts from stone 1, and repeatedly jumps forward until it reaches stone n. From stone i the frog may jump either to $i+1$ or $i+2$.*

1. *If the frog jumps from i to $i+1$, then the cost is $|h_i - h_{i+1}|$.*
2. *If the frog jumps from i to $i+2$, then the cost is $(h_i - h_{i+2})^2$.*

Design an algorithm with the following guarantees:

Input: *An array $h[1, \ldots, n]$ of costs.*

Goal: *Find the sequence of jumps that allows the frog to reach stone n at minimal total cost.*

Running time: $O(n)$.

Example: *Let $n = 7$ and $h = [3, 0, 2, 4, 5, 7, 3]$. The frog may reach the last stone using the following sequence of steps*

1. *Jump from 1 to 3 at cost $(3-2)^2 = 1$.*
2. *Jump from 3 to 4 at cost $|2-4| = 2$.*
3. *Jump from 4 to 5 at cost $|4-5| = 1$.*
4. *Jump from 5 to 7 at cost $(5-3)^2 = 4$.*

The total cost here is $1 + 2 + 1 + 4 = 8$.

Question 4 (10 points)

1. **[7 points]** *Design a polynomial time algorithm that gets as input a graph G on n vertices such that G contains an independent set of size at least $3n/5$, and returns an independent set of size at least $n/5$.*

2. **[3 points]** *Does the algorithm above imply that the Max-Independent-Set problem has a poly-time 1/3-approximation algorithm? Explain your answer.*

Question 5 (10 points) *Given an undirected graph $G = (V, E)$, denote by $Rem_\Delta(G)$ the minimum number of edges required to remove from G so that the remaining graph contains no triangle.*

1. **[5 points]** *Prove that $Rem_\Delta(G) \leq |E|/2$ for all graphs $G = (V, E)$.*

   (Hint: a bipartite graph has no triangles.)

2. **[5 points]** *Design a deterministic polynomial time 3-approximation algorithm for $Rem_\Delta(G)$. That is, the algorithm outputs a subset of edges $E' \subseteq E$ such that $G = (V, E \setminus E')$ contains no triangles, and $|E'| \leq 3 \cdot Rem_\Delta(G)$.*

Question 6 (10 points)

In the weighted MAX-CUT problem the input is an undirected graph $G = (V, E)$, and weights $w_e \geq 0$ for each $e \in E$.

The goal is to partition V into two disjoint sets $V = A \cup B$ so as to maximize the total weight of the edges between A and B (that is, $\sum_{\substack{(u,v)\in E \\ u\in A, v\in B}} w_{u,v}$). Let MAX-CUT(G) be the total weight of the arcs going from A to B in the optimal partition.

Given an undirected graph $G = (V, E)$ with weights $\{w_e > 0 : e \in E\}$, consider the following LP:

$$\text{maximize} \sum_{(u,v)\in E} \frac{w_{(u,v)}}{2} \cdot (1 - x_{(u,v)})$$

$$\text{subject to} \begin{cases} -1 \leq x_{(u,v)} \leq 1 & \forall (u,v) \in E \\ x_{(u,v)} + x_{(v,w)} + x_{(u,w)} \geq -1 & \forall (u,v),(v,w),(u,w) \in E \end{cases}$$

(a) **[3 points]** *Prove that LP-value $\geq$ MAX-CUT(G) for any graph G.*

(b) **[3 points]** *Prove that LP-value $\leq 2 \cdot$ MAX-CUT(G) for any graph G.*

(c) **[4 points]** *Show a graph G for which the LP value is strictly larger than MAX-CUT(G). Show a multiplicative gap as large as you can.*