CMSC 351 Summer 2025 Homework 11

Due Thursday 21 August 2025 by 23:59 EST on Gradescope.

Directions:

- Homework must be done on printouts of these sheets and then scanned properly, or via Latex, or by downloading, writing on the PDF, and uploading. If you use Latex please do not change the Latex formatting.
- Do not use your own blank paper!
- The reason for this is that Gradescope will be following this template to locate the answers to the problems so if your answers are organized differently they will not be recognized.
- Tagging is automatic, you will not be able to manually tag.

---

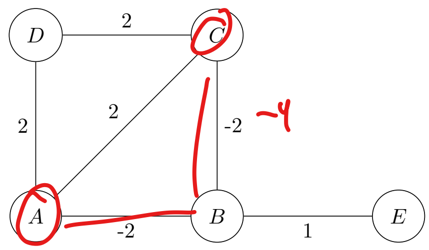
1. Complete the following Sudoku puzzle! [12 pts]

8	9	1	3	7	4	6	5	2
7	2	6	1	5	8	9	4	3
3	4	5	6	2	9	1	7	8
4	7	9	5	8	6	2	3	1
1	6	2	4	3	7	8	9	5
5	3	8	9	1	2	4	6	7
9	1	3	8	6	5	7	2	4
6	8	7	2	4	3	5	1	9
2	5	4	7	9	1	3	8	6

2. For each of the following decision problems and instances write YES or NO in each box stating whether the witness is valid or not.

   (a) Q: Given a weighted graph, are there two vertices which have the property that there is a walk between them whose total weight is 0 ? [9 pts]

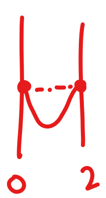
   I: This graph:



Witness	Valid?
A and C	NO
A and B	YES
A and E	NO

   (b) Q: Given a function, is there an interval on which it is increasing? [9 pts]

   I: The function $f(x) = x^3 - 4x$



Witness	Valid?
$[-2, -1.9]$	YES
$[0, 2]$	No
$[2, \infty)$	YES

$(-2)^3 - 4(-2)$

$= -8 + 8 = 0$

$(-1.9)^3 - 4(-1.9)$

3. Consider this decision problem: [20 pts]

> Q: Given a simple, connected, unweighted, undirected graph with $n \geq 3$ vertices represented by an adjacency matrix A, is there a cycle of length exactly 3?

Explain why $Q \in P$.

Your explanation does not need to be a formal proof but should be clear enough to explain how the DTM runs in polynomial time and under what conditions YES and NO are output.

Solution:

To check for 3-pairs (cycle) by brute force in an adjacency matrix we can just brute force checking every 3-pair. This would be n^3 checks and would therefore run in $O(n^3) = O(n^k)$

so $Q \in P$

YES will output when there are 3 distinct vertices and there exists an edge between all 3 of them (all edges are non-infinity)
NO will output if there is not, or all other cases I suppose

Even if you optimize this to only check for 3-distinct vertex pairs AND to disregard duplicates like 1-2-3 == 2-3-1
it would still run in polytime i think

4. Suppose k is a positive integer. The Weight-Constrained Spanning Tree Problem asks:

Q: Given a simple, connected, weighted, and undirected graph (represented as an adjacency matrix) and an integer $k \geq 0$, is it possible to find a spanning tree with total weight less than or equal to k?

For a graph I and a spanning tree x we wish to define a verifier $V(I, x)$ which satisfies the verifier definition of NP.

assume x is a valid spanning tree?

(a) What definition of $V(I, x)$ works? [5 pts]

Solution:

V(I,x) w/ I=graph and x=spanning tree
V(I,x) adds the weights from all of the edges in x.
Returns YES if sum<=k, NO otherwise.

(b) Justify why $V(I, x)$ runs in polynomial time as a function of the number of vertices. [5 pts]

Solution:

Because summing a list of edges n runs in O(n) which is polytime

$O(n) = O(n^1) = O(n^k)$

(c) Show that $V(I, x)$ satisfies the verifier definition. [10 pts]

Solution:

(a) $\forall I, \forall x \, (W(I,x) = YES \rightarrow Q(I) = YES)$

(b) $\forall I \, (Q(I) = YES \longrightarrow \exists_x, W(I,x) = YES)$

(a) given (I,x), if W(I,x) = YES, that means the total weight of spanning tree x<=k,
and so Q(I) = YES
(b) given graph I, if Q(I) = YES, there exists a spanning tree x such that its total weight <=k
let x be that spanning tree, then W(I,x) = YES

5. The Nonempty Subset Target Sum Problem asks:

> Given a set S of integers and a target k, find and return a nonempty subset of S which adds up to k. If none exists, return NONE.

(a) Just for a warm-up, given $S = \{-1, 3, 7, 2\}$ and $k = 4$, what is a result of the Nonempty Subset Target Sum Problem? [5 pts]

Solution:

{-1,3,2}

(b) Just for a warm-up, given $S = \{-1, 3, 7, 2\}$ and $k = 100$, what is a result of the Nonempty Subset Target Sum Problem? [5 pts]

Solution:

NONE

(c) Suppose we have an oracle which works as follows: We give it a set of integers, an element x in the set, and a target k, it tells us whether or not there is a nonempty subset of the set which includes x and adds up to k. [20 pts]

Given a set S of integers, explain why the Nonempty Subset Target Sum Problem is polynomially reducible to the oracle.

You can treat the set as a list for anything algorithmic.

Solution:

The oracle checks if there is a nonempty subset which includes x and adds to k.
Here we iterate through all x in the set, checking the oracle at each x:
- If the oracle returns true at some x, a valid subset exists, Q returns YES and we have an instance w/ current x and the target k, set S from the original problem Q.
- If no oracle ever returns true, no valid subset exists, so the Q will return NO.

This is valid because the oracle checks the subset sum validity at each x, essentially just representing one iteration of a full brute force check. Set size = n: We run at most n iterations w/ oracle checking being polytime.

Example: S = { -4, 2, 7, 1 }, k = 8
Try each x in the set:
x= -4: does there exist a subset w/ -4 that sums to 8?
-> NONE
x = 2: does there exist a subset w/ 2 that sums to 8?
-> NONE
x=7: does there exist a subset w/ 7 that sums to 8?
-> {7,1} works so YES returns with instance {7,1}
Stop after this iteration, return YES.
Here we ran n=4 oracle calls -> polynomial