CMPE/CISC-365*
Quiz #4
November 29, 2019

Student Number (Required) ______________________

Name (Optional)________________________________

This is a closed book test. You may refer to one 8.5 x 11 data sheet.

This is a 50 minute test.

Please write your answers in ink. Pencil answers will be marked, **but will not be re-marked under any circumstances.**

The test will be marked out of 50.

Question 1	/30
Question 2	/5
Question 3	/15
TOTAL	/50

QUESTION 1 (30 Marks)

The year is 2028. Kanye West has just been elected Lifetime Secretary General of the United Nations. You are in charge of evacuating the few remaining members of the League of Rational People to their new home on Titan. You have a large fleet of one-way ships at your disposal, and your task is to assign **all the people** to ships in such a way as to **minimize the total launch cost**.

There are ***n ships*** and ***m people***. You may assume that $n \geq m$. You are not required to launch all the ships (only the ones you assign people to).

Each ship S_i has a maximum load value m_i that specifies the maximum number of kilograms the ship can carry.

Each ship S_i has a launch cost value c_i, but we only launch ships if they have passengers. We can define R_i as

$$R_i = \begin{cases} 1 \text{ if ship } S_i \text{ is occupied} \\ 0 \text{ if ship } S_i \text{ is empty} \end{cases}$$

Then the total cost of launching is C, where $C = \sum_{i=1}^{n} R_i * c_i$

Each person P_i has mass g_i. You may assume that no person has a mass that is greater than the capacity of any ship.

So the goal is to minimize the total cost C, with the constraints :

- each person is assigned to a ship, and
- no ship is overloaded (ie the sum of the masses of the people in ship S_i does not exceed m_i)

Here is a very small example

Person	P_1	P_2	P_3	P_4
Mass	80	75	100	90

Ship	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
Capacity	200	300	175	150	500	275	310	340
Launch Cost	100	150	25	90	600	120	160	140

One solution is to assign P_1 and P_2 to S_1, P_3 to S_7, P_4 to S_3.

The total launch cost for this solution is 100 + 160 + 25 = 285

You have been instructed to create a Branch & Bound algorithm to find the least expensive combination of ships that will carry all the people to Titan.

In this question, correct answers that demonstrate careful thought will receive more marks than answers that are correct but trivial.

(question continues on next page)

(a) **[5 marks]** Describe the sequence of decisions your algorithm will make. Will you iterate over the list of people or over the list of ships, or some combination? Explain your reasoning.

Solution:

Iterate through the list of passengers, assigning each one to a ship. ***5 marks***

OR

Iterate through the list of ships, choosing a set of passengers for each ship. ***3 marks***

This solution is impractical because it requires considering O(2^m***) possible expansions for each partial solution***

Marking: Give part marks for anything that is plausible. The essential requirement is that there must be a defined sequence of decisions that can eventually construct the optimal solution no matter what it is. For example, a decision sequence that forces some passenger to be assigned to a particular ship (or never considers some possible ship/passenger combinations would not be acceptable.

(b) **[8 marks]** Describe how you will compute the initial value of the Global Upper Bound. (If you decide to sort the data, explain your reasons for doing so.)

Solution: Sort the passengers into descending order by mass, and sort the ships into ascending order by cost value. My reason for doing this is to try to pay as little as possible to launch the passengers with the greatest mass. Construct a feasible solution by iterating through the sorted list of passengers, assigning each passenger to the first ship that has room for the passenger.

OR

Sort the passengers into ascending order by mass, and sort the ships into ascending order by cost value. The rationale here is to try to get as many passengers as possible into the inexpensive ships.

Marking:

Sorting makes these Greedy heuristics possible. They don't have to sort on the same criteria as I suggest, but they should sort ships, passengers or both, and give a reason.

"Assign people to ships (or ships to people) without sorting anything": 3 marks

"Sort the values then assign people to ships (or ships to people)" without explanation: 5 marks

"Sort the values then assign people to ships (or ships to people)" with explanation: 8 marks

(c) **[3 marks]** Describe how you will compute Cost So Far for each partial solution

Solution:

Compute the total launch cost for all ships which have passengers assigned to them.

Marking:

For the correct answer: 3 marks

For an incorrect answer that shows understanding of the "Cost So Far" concept": 1 mark

(d) **[8 marks]** Describe how you will compute Guaranteed Future Cost for each partial solution.

Solution:

Method 1: For each remaining passenger, determine if any of the ships already in use have room for this passenger. If there is any passenger that cannot fit in one of the active ships, add the launch cost of the least expensive currently empty ship. This is effectively saying "We need at least one more ship". This is a weak answer, but it is valid.

Method 2: Focus on the remaining passengers who cannot be assigned to any ship that is currently in use. Sum their masses and divide by the capacity of the largest unused ship. This gives a lower bound LS on the number of ships needed to accommodate all remaining passengers. Sum the smallest LS costs of the unused ships. This sum is a guaranteed future cost.

Method 3: Sum the masses of all unassigned passengers. Subtract from this the remaining capacity in all active ships. This gives a total mass TM that must be accommodated in more ships. Taking the empty ships in descending order of capacity, reduce TM by each ship's capacity until TM = 0. This gives a lower bound LS on the number of ships needed to accommodate all remaining passengers. Sum the smallest LS costs of the unused ships. This sum is a guaranteed future cost.

Other methods are certainly possible!

Marking: the answer should incorporate some analysis of the remaining passengers and ships. This is the hardest part of this question, and some leniency can be applied in grading it. If they show or state an understanding of Guaranteed Future Cost (ie, a value x such that all expansions of the partial solution cost $\geq$ x), they should get at least 4 marks.

For something like Method 1:	**6 marks**
For something like Method 2 or 3:	**8 marks**
For something that is unlike any of my solutions but which works:	**8 marks**

For a method that tries to do something like one of these but contains errors, deduct 1 or 2 marks

(e) **[6 marks]** Describe how you will compute Feasible Future Cost for each partial solution.

Solution: Use exactly the same technique as was described for computing the initial value of the Global Upper Bound, applied to the remaining passengers.

Marking: This question is straightforward, but it is based on their answer for computing the Global Upper Bound. If their answer for that part doesn't work and they use it again here, they should not lose marks for it twice.

If they show that they know what Feasible Future Cost means but they cannot compute it, give at least 3 marks.

QUESTION 2 (5 marks)

Professor Snope believes that he has found a proof that **Max_Clique** can be solved in polynomial time. (Remember, **Max_Clique** is "Given a graph G, find the size of the **largest** set of vertices in G that are all neighbours of each other.") His proof goes like this: Let G be a graph with n vertices. Solve the Max_Clique problem on G using this algorithm:

1. temp = 1
2. check every pair of vertices in G to see if any of them are neighbours … this takes $O(n^2)$ time. **If a pair of neighbours is found**, temp = 2
3. check every group of 3 vertices in G to see if they are all neighbours … this takes $O(n^3)$ time. **If such a group is found**, temp = 3
4. check every group of 4 vertices in G to see if they are all neighbours … this takes $O(n^4)$ time. **If such a group is found**, temp = 4
5. check every group of 5 vertices in G to see if they are all neighbours … this takes $O(n^5)$ time. **If such a group is found**, temp = 5

etc.
…
return temp

Since each step runs in $O(n^t)$ time for some integer t, Snope claims that his algorithm runs in polynomial time. Is he right? Explain your answer.

(Use the next page – even though your answer may be quite short I'll give you lots of space for it.)

Answer page for Question 2

Solution:

Snope is wrong. For a problem to be solvable in polynomial time, it must be true that there is an algorithm that solves all instances of size n that runs in $O(n^k)$ ***time for some constant k. Snope's algorithm executes a sequence of*** $O(n^t)$ ***stages, but t is is not constant. For example, at some point in the algorithm we will need to examine all groups of n/2 vertices, which by his own analysis will take*** $O(n^{n/2})$ ***time.***

Marking

Students who say that Snope is wrong but give no explanation should get 3 marks.

Students who say he is wrong but give a poor explanation should get 4 marks.

Students who state that Snope is correct and give a justification such as "the sum of polynomials is polynomial" should get a mark of about 2

Students who say he is correct and give no explanation should get 1

QUESTION 3 (15 Marks)

Consider the following restricted form of the Subset Sum Problem:

Multiples-of-5-Subset-Sum (M5SS): Let S be a set of numbers all of which are multiples of 5, and let k be an integer that is a multiple of 5. Does S contain a subset that sums to k?

Example: S = {15, 10, 1375, 840, 200} k = 225 … in this case the answer is "Yes"

(a) (5 marks) Show that M5SS $\in NP$ (ie. show that M5SS has the properties required to be a member of the class NP)

SOLUTION:

M5SS is a decision problem for which the answer is completely determined by the instance – there is no randomness or nondeterminacy.

If the answer to an instance of M5SS is Yes and we know the details of the solution, we can verify it in O(n) time by adding the selected elements of S and confirming that they sum to k.

Therefore M5SS $\in NP$

ALTERNATIVE SOLUTION: M5SS is a special case of Subset-Sum, which we know is $\in$ ***NP. Therefore M5SS*** $\in NP$.

Marking

for the alternative solution – it is short but complete!	5
if they mention the "verify Yes in polynomial time" but leave out the "decision problem" requirement	4
if they mention the "decision problem" requirement but leave out the "verify Yes in polynomial time" requirement	3
if they show they know the requirements but cannot show that this problem meets them	2
for showing partial knowledge of the requirements with no connection to this problem	1

(b) (10 marks) Prove that M5SS is NP-Complete by reducing Subset Sum to M5SS. Remember to show that your transformation of an instance of Subset Sum to an instance of M5SS takes polynomial time, and that your transformation is answer-preserving.

SOLUTION: Let S,k be an instance of Subset Sum, and let |S| = n.

Construct an instance S′, k′ of M5SS by multiplying each element of S, and k, by 5. This transformation takes O(n) time.

Proof that the transformation is answer-preserving:

Suppose the answer to Subset Sum on S, k is Yes. Then the corresponding subset in S′ will sum to 5*k, which is the value of k′ … so the answer to M5SS on S′,k′ is Yes.

***** Suppose the answer to M5SS on S′,k′ is Yes. Then the corresponding subset in S will sum to k′/5, which is the value of k … so the answer to Subset Sum on S, k is also Yes.***

ALTERNATIVE ARGUMENT for paragraph ** : Suppose the answer to Subset Sum on S,k is No. That means every subset of S sums to something other than k. That means every subset of S′ sums to something other than k′ since the sum of a subset in S′ is just 5 times its sum in S. So the answer to M5SS in S′,k′ is No.

Marking: for the transformation: 3 marks

- other valid transformations are possible. For example we can multiply all values by 10
- give part marks for partial answers
- give 2 for a transformation that doesn't work

for demonstrating that it is answer-preserving: 7 marks

- part marks for incomplete or partially correct answers

- for describing what needs to be done, but not being able to do it for this problem : 4 marks