National University of Singapore

School of Computing

CS4234 - Optimisation Algorithms

(Semester 1: AY2017/18)

Date and Time: Tuesday, 10 October 2017, 12.05-13.35 (90m)

INSTRUCTIONS TO CANDIDATES:

1. Do **NOT** open this assessment paper until you are told to do so.
2. This assessment paper contains THREE (3) sections.
   It comprises TEN (10) printed pages, including this page.
3. This is an **Open Book Assessment**.
4. Answer **ALL** questions within the **boxed space** in this booklet.
   Only if you need more space, then you can use the empty page 10.
   You can use either pen or pencil. Just make sure that you write **legibly**!
5. Important tips: Pace yourself! Do **not** spend too much time on one (hard) question.
   Read all the questions first! Some questions might be easier than they appear.
6. You can use **pseudo-code** in your answer but beware of penalty marks for **ambiguous answer**.
   You can use **standard, non-modified** classic algorithm in your answer by just mentioning its name, e.g. run Dijkstra's on G, run Kruskal's on G', etc.
7. For **Section B and C**, the max number of allowed operations is 100 Million per test case.

My Student Number:

A	0							

This portion is for examiner's use only

Section	Maximum Marks	Your Marks	Remarks
A	30		
B	30		
C	40		
Total	100		

A Be the Computer (30 marks)

Part 1-4

For **Part 1-4**, you are given a 'small' instance of a disconnected graph. In **Part 1** and **Part 3**, the graph is weighted and values inside the circles (that represent the vertices) are in this format: "Vertex Number/Weight of that Vertex". In **Part 2** and **Part 4**, the graph is unweighted.

A.1 Part 1 (8 marks)

Solve the MIN-WEIGHT-VERTEX-COVER (MWVC) on Figure 1 below by **circling the vertices** that you want to put inside the minimum weight vertex cover **and write down** the value of the optimal answer in the box provided. Answer that is a vertex cover but **not** the minimum weight will be given partial marks only if it is not worse than 2-times the minimum weight.

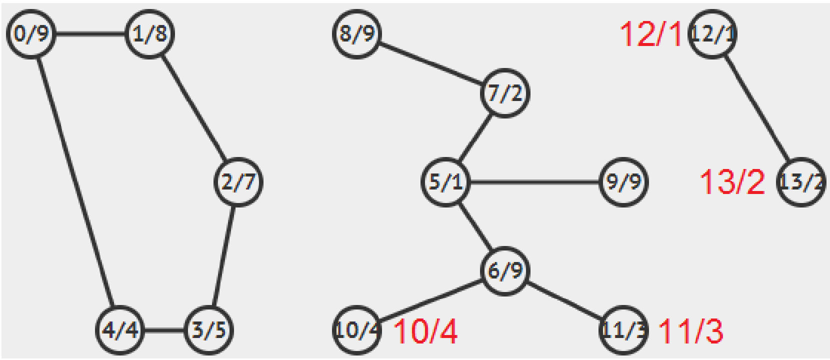

Figure 1: Instance for Part 1, circle the vertices in your MWVC and write the minimum weight below:

A.2 Part 2 (6 marks)

Please redo the question from **Part 1** above, but now the vertices in Figure 2 are unweighted.

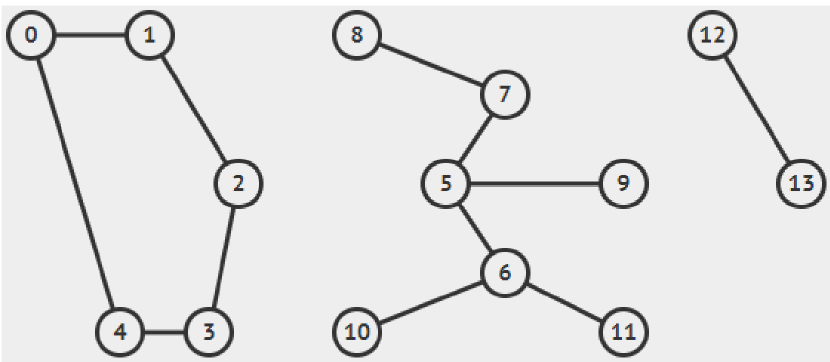

Figure 2: Instance for Part 2, circle the vertices in your MWVC and write the minimum weight below:

A.3 Part 3 (4 marks)

Now, what if we have to solve a different problem instead: The MAX-WEIGHT-INDEPENDENT-SET (MWIS). If you forget the definition, here it is: "Given a graph $G = (V, E)$, pick the maximum-weight set $I \subseteq V$ so that no two vertices in I share an edge.". Solve the problem on the graph in Figure 3 (which is exactly the same as in **Part 1** above).

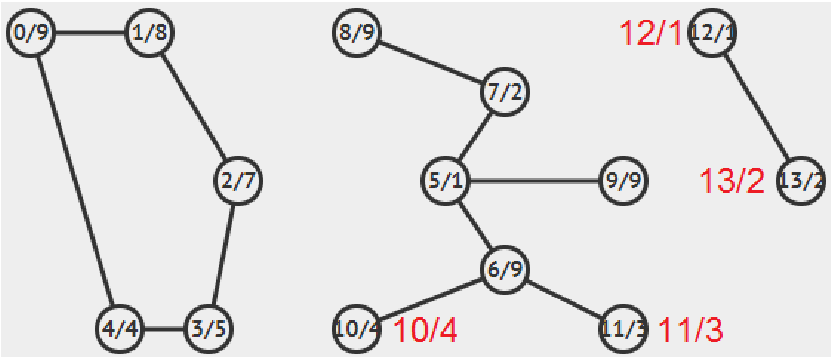

Figure 3: Instance for Part 3, circle the vertices in your MWIS and write the maximum weight below:

A.4 Part 4 (2 marks)

Now solve the MWIS problem on the unweighted graph in Figure 4 (which is exactly the same as in **Part 2** above).

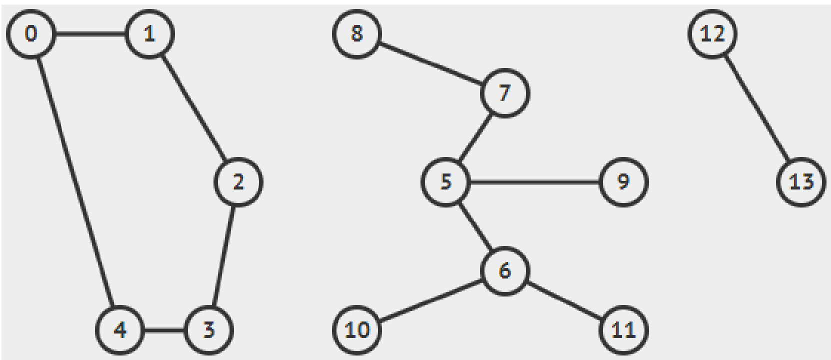

Figure 4: Instance for Part 4, circle the vertices in your MWIS and write the maximum weight below:

Part 5-6

In **Part 5-6**, you are given the same question: Given a list L of N $(2 \leq N \leq 10)$ integers, we draw a directed edge (u, v) with weight (capacity) $gcd(u, v)$ if $u \in L, v \in L, v > u, gcd(u, v) > 1$. Note that $gcd(u, v)$ means the greatest common divisor between the two integers u and v, i.e. $gcd(2, 4) = 2$, $gcd(2, 8) = 2$, $gcd(4, 8) = 4$, $gcd(4, 7) = 1$, etc. We set source vertex s as the smallest integer in L and sink vertex t as the largest integer in L. Your task is simple: Compute the maximum flow on this specially designed flow network.

For example: If $N = 3$ and $L = \{2, 4, 8\}$, then the answer is 4 (see Figure 5).

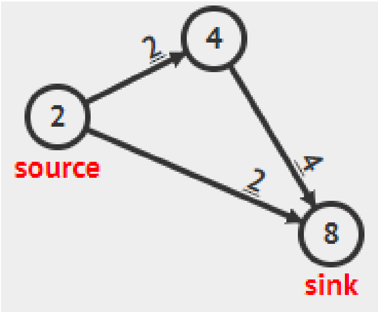

Figure 5: Flow Graph Example

A.5 Part 5 (4 marks)

If $N = 4$, $L = \{4, 6, 8, 9\}$, what is the answer?

A.6 Part 6 (6 marks)

If $N = 10$, $L = \{5, 3, 7, 27, 11, 18, 13, 19, 9, 17\}$, what is the answer?

Section A Marks = __ + __ + __ + __ + __ + __ = ___

B Decision Problems (30 Marks)

B.1 Unique Ranking? (15 Marks)

Steven has N ($1 \leq N \leq 100$) competitive students[1] aiming to be part of his ACM ICPC teams.

He currently has M ($0 \leq M \leq N \times (N-1)/2$) *confirmed observations* of the relative strengths of 2 students in form of pair (A, B) where student A is objectively judged to be 'more competitive' than student 'B' for NUS ACM ICPC programming team. You can assume that Steven is a veteran coach, e.g. if he finds that A is 'more competitive' than B, and B is 'more competitive' than C, then A is 'more competitive' than C too.

He wants to know if he can uniquely order his N students in descending order from the most competitive to least competitive so that he can decide who to pick for NUS ACM ICPC teams 2017/2018?, i.e. decide YES/NO.

For example, if $N = 3$ (students 0, 1, 2) and $M = 2$ (e.g. Steven currently knows that student 0 is more competitive than student 1 and student 0 is also more competitive than student 2). Then Steven knows that he will rank student 0 higher than both student 1 and 2. But with just those $M = 2$ confirmed observations, he has no idea where student 1 stands compared to student 2 as he does not have this information yet. If he then do one more individual contest between this pair and found that student 2 is more competitive than 1, then his final order is confirmed, i.e. student 0, 2, then 1.

Is this an NP-complete decision problem? If yes, prove it by showing a polynomial-time reduction from a known NP-hard problem into this problem and show how you are going to attack this problem given that N can be up to 100. However, if you are sure that this is not an NP-complete decision problem, show a polynomial time algorithm to solve it and what is the time complexity?

[1] Yes, assume that he had lots of students who applied; actually he only have 36 applicants and have picked his top 18 NUS students for this AY 2017/2018 during last month NUS ACM ICPC Selection Contest on Sat, 02 Sep 2017.

B.2 Nonogram? (15 Marks)

Given an empty $R \times C$ bitmap grid and R row sums and C column sums ($1 \leq R, C \leq 50$), your job is to decide if you can fill in that empty grid with either a 0/white or a 1/black such that all row and column sums are satisfied? For example 1 below, the answer is a YES, e.g. by filling the first row with '1'+'0' and the second row with '0'+'1'. Note that this is not the only valid way.

```
-----                -----
|?|?| -> 1           |1|0| -> 1
-----                -----
|?|?| -> 1           |0|1| -> 1
-----                -----
 v v                  v v
 1 1                  1 1
```

For example 2 below, the answer is clearly a NO.

```
-----
|?|?| -> 1
-----
|?|?| -> 2
-----
 v v
 2 2
```

Design your best algorithm to decide this and analyze its time complexity.

Section B Marks = ___ + ___ = ___

C (NP?-)hard Questions (40 Marks)

C.1 The-Quick-Brown-Fox-Jumps-Over-The-Lazy-Dog (20 Marks)

Do you realize that the sentence 'THEQUICKBROWNFOXJUMPSOVERTHELAZYDOG' (all in UPPERCASE with spaces omitted) uses ALL characters ['A'..'Z']? We call such sentence: 'Pangram'.

Now, given N English words ($1 \leq N \leq 25$, each word uses only uppercase ['A'..'Z'] and is not more than 100 characters), write an algorithm to pick the smallest number of word(s) to form a Pangram. We guarantee that the answer is ≥ 1. Also analyze the time complexity of your algorithm!

Example 1: If $N = 1$ and the only word is 'ABCDEFGHIJKLMNOPQRSTUVWXYZ', then the answer is just 1 word. Example 2: If $N = 3$ and the words are 'THEQUICKBROWNFOXJUMPSOVERTHELAZY', 'DOG', and 'CAT', then it the answer is 2 (if we pick 'THEQUICKBROWNFOXJUMPSOVERTHELAZY' and 'CAT', we still haven't use character 'D' and 'G').

C.2 Prison-Break (20 Marks)

Given a map of a prison in a 2D grid of size $R \times C$ ($0 \leq R, C \leq 1\,000$) where a 'G' represents the garden (outside world), a '#' represents a very strong wall, a '.' represents an open space, a 'D' represents a jail door, ~~and a 'P' (there are exactly two of them) represents the two prisoners (~~'L' and 'M'~~)~~ who are trying to escape. For the purpose of this problem, digging tunnels or breaking walls are impossible so 'L' and 'M' will steal jail door key(s). Now after months of planning and scouting, they have determined which guard(s) has which jail door key(s). Each attempt of stealing a key is risky so they want to minimize it. Tonight, they will have a last meeting during dinner to decide which subset of jail door key(s) that they have to steal so that both 'L' and 'M' can reach **any** 'G' (garden) cell (which means freedom). Then they will execute their plan at dawn tomorrow morning.

If you are 'L' or 'M', what is the best algorithm that you will use and what is its time complexity?

Example 1: For this map, 'L' (at 2, 2, we are using 0-based indexing) and 'M' (at 2, 6) have to steal door keys at (2, 3) (to free 'L'), at (2, 5) to free ('M'), and **just one** of either door at (1, 4) or (3, 4) to reach the garden, together. The final answer is 3.

```
GGGGGGGGG
G###D###G
G#LD.DM#G
G###D###G
GGGGGGGGG
```

Example 2: For this other map, The final answer is 2 ('Every man for himself': 'L' have to steal one door key below him and similarly 'M' have to steal one door key below him).

```
GGGGGGGGG
G#######G
G#LD.DM#G
G#D#.#D#G
G#.#D#.#G
GGGGGGGGG
```

If you are not confident that your answer is correct for the original problem above, please solve this variant for partial marks: What if there is no prisoner 'M'? If there is only 'L' alone in the problem above. What is the minimum number of jail door key(s) that he has to steal to gain freedom?

Section C Marks = ___ + ___ = ___

– End of this Paper, All the Best, You can use this Page 10 for extra writing space –