

National University of Singapore  
School of Computing  
**CS4234 - Optimisation Algorithms**  
(Semester 1: AY2023/24)

Thursday, 30 November 2023, AM (2 hours)

---

INSTRUCTIONS TO CANDIDATES:

1. Do **NOT** open this assessment paper until you are told to do so.
2. This assessment paper contains FOUR (4) sections.  
It comprises SIXTEEN (16) printed pages, including this page.
3. This is an **Open Book Assessment**.
4. Answer **ALL** questions within the **boxed space** of the answer sheet (page 12-16).  
For Section A, shade the option in the answer sheet (use 2B pencil).  
There are a few starred (\*) boxes: free 1 mark if left blank but 0 for wrong answer (no partial).  
The answer sheet is at page 12-16 but you will still need to hand over the entire paper.  
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 (subtask) 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 Kuhn-Munkres on graph  $G$ , Edmonds' Matching on graph  $G'$ , etc.
7. All the best :)

**A MCQs (21 × 2 = 42 marks)**

Select the **best unique** answer for each question. Each correct answer worth 2 marks.

[PAGE 2 IS NOT ARCHIVED]

[PAGE 3 IS NOT ARCHIVED]

[PAGE 4 IS NOT ARCHIVED]

[PAGE 5 IS NOT ARCHIVED]

[PAGE 6 IS NOT ARCHIVED]

[PAGE 7 IS NOT ARCHIVED]

---

[The upper side of PAGE 8 IS NOT ARCHIVED]

## B Edmonds' Matching Algorithm (8 marks)

Draw a graph  $G = (V, E)$  with the following constraints:

- It must have  $n \geq 9$  vertices and  $n \% 3 = 0$ ,
- Vertices are labeled from 0 to  $n - 1$ ,
- Vertices are unweighted,
- Edges are undirected and unweighted,
- The number of edges  $m$  is up to you,
- For each vertex  $u$ , process its neighbors in ascending vertex number.

So that if Edmonds' matching algorithm is run on  $G$ , it shrinks  $\frac{n-1}{2}$  blossoms throughout the execution of the algorithm. To simplify grading, draw a quick sketch the  $\frac{n-1}{2}$  blossom shrinking processes.

Hint: The smallest blossom is a cycle of length 3 that starts from a free vertex, alternates free-match-free, and comes back to the same free vertex.

## C Setting-Problems (25 marks)

A certain Competitive Programming course at a certain University will have its Midterm Team Contest soon! Hence, the professor of that course goes to consult his TAs, who propose a set of  $N$  (see the constraints in Section C.4) problems for the contest. The professor then looks through the problems (numbered sequentially as problem 1 to  $N$ ), and estimates the difficulty of each problem as an array  $D = \{d_1, d_2, \dots, d_N\}$ . However, he notices a major issue:  $M$  of the problems are too similar (in format of  $M$  pairs  $(a, b)$ , where  $a$  and  $b$  are the problem numbers), and it would not make sense to include two similar problems in the contest! However, the professor still needs at least  $K$  ( $1 \leq K \leq N$ ) problems



to run the contest. Please help the professor figure out if he has enough problems, or if he has to quickly come up with more (and simpler) problems for this upcoming Midterm Team Contest!

Note that if problem  $a$  and  $b$  are too similar and subsequently problem  $b$  and  $c$  are also deemed too similar, the professor is still OK to use both problems  $a$  and  $c$  (if need be – depends on  $K$  and the difficulty ratings too) as they are deemed ‘different enough’, perhaps a big fraction of his students will not notice the not-so-direct similarity anyway.

From past semesters, the professor has received feedback on the difficulty of the contests, and he has decided to be nicer this year. If there are multiple subsets of problems that fulfill the requirements, he will always pick the one with the one with the lowest combined difficulty. So, given  $N$ , array  $D$ ,  $M$  pairs of similar problems, and  $K$ , output a single integer: the lowest combined difficulty of a valid subset of problems, or -1 if no such subset exists.

For example: There are  $N = 5$  problems proposed with difficulty ratings of  $D = \{19, 7, 20, 99, 1\}$  (problem numbers are 1-based). The professor notices that  $M = 2$  of them are too similar, i.e., problem (1, 2) and problem (2, 3). If the professor needs a contest with  $K = 3$  problems, he will pick problem 5 (the easiest), 1, and 3 with combined difficulty of  $1 + 19 + 20 = 40$ . If the professor just needs  $K = 2$  problems, he will pick problem 5 and 2 with combined difficulty of  $1 + 7 = 8$ . If the professor needs  $K = 5$  problems, he is out of luck and the output is  $-1$ .

### C.1 Three Manual Test Cases (3 marks)

Output the required answer and a short explanation (1 mark each) if you are given:

1.  $N = 3$ ,  $D = \{1, 2, 3\}$ ,  $M = 1$ , problems (1, 3),  $K = 2$ .
2.  $N = 3$ ,  $D = \{1, 2, 3\}$ ,  $M = 3$ , problems (1, 3), (3, 2), and (2, 1),  $K = 2$ .
3.  $N = 11$ ,  $D = \{98, 54, 6, 34, 66, 63, 52, 39, 62, 46, 75, \underline{-}, \underline{-}, \underline{-}\}$ ,  $M = 7$ , problems (3, 9), (7, 9), (3, 7), (9, 10), (1, 10), (8, 11), (4, 9),  $K = 7$ .

### C.2 Formulate as ILP (4 marks)

Formulate `Setting-Problems` as an ILP. You can use pseudocode.

### C.3 Prove `Setting-Problems` is NP-hard (4\* marks)

Prove that `SETTING-PROBLEMS` is NP-hard.

Hint: Use a known NP-hard (decision) problem that we have learned in class.

### C.4 Write Your Best Complete Search (6 marks)

Knowing that the general form of `Setting-Problems` is NP-hard (see Section C.3), write your best complete search solution for this task and analyze your (exponential) time complexity. You will be graded as follows: 0/2/3/4/6 if blank (or wrong algorithm), fast enough<sup>1</sup> for  $1 \leq N \leq 20$ , [in-between-but-closer-to-N-20], [in-between-but-closer-to-N-26], fast enough for  $1 \leq N \leq 26$ , respectively.

<sup>1</sup>Fast enough means not more than 7 seconds assuming today’s computer can do  $10^8$  operations in 1 second.

### C.5 Solve a Special Case (4\* marks)

Assumes that the Prof simplifies his own constraint by never declaring odd-length chain of similarities, e.g., if the Prof declares  $(a, b)$  and then  $(b, c)$  as similar, he will never also declare  $(c, a)$  as similar.

If this special case changes your previous answer in Section C.4 (e.g., the solution becomes polynomial), please explain the required improvement(s) to your solution!

### C.6 Local Search Ideas (4 marks)

Design a (Stochastic) Local Search for the this problem. Describe the rough ideas of your algorithm (minimally 4 ideas, 1 mark each). Your answer will be graded based on how sound the heuristics/local search ideas that you propose, albeit we cannot test your ideas empirically in this theory paper.

## D Tuturuu (25 marks)

Okabe, the mad scientist, has accidentally created a new strain of COVID, called Tuturuu. Being responsible, he set out on a quest to find a cure. However, the task proved to be too difficult for him and he would like to ask the help of his friend, the genius scientist Kurisu.

Okabe have collected  $N$  ( $1 \leq N \leq 1500$ ) different weights  $W_1, W_2, W_3, \dots, W_N$  of inactive Tuturuus. Okabe has an infinite amount of inactive Tuturuus for each weight  $W_i$  ( $1 \leq W_i \leq 900$ ). Okabe has a special activation reagent. One drop of it can split an inactive Tuturuu of weight  $W$  into two active Tuturuus of weights  $d$  and  $\frac{W}{d}$ , where  $d$  is a positive divisor of  $W$ . Okabe can freely choose  $d$ , as the reagent is special. In order to develop the cure, he needs to produce the full set  $S$  of all the possible active weights that can be a result of dropping the reagent on the inactive Tuturuus.

As an example, if there are 2 inactive Tuturuus with weights  $\{2, 4\}$ , then the inactive Tuturuus with weight 2 may produce active Tuturuus with weights  $\{1, 2\}$  and the inactive Tuturuus with weight 4 may produce active Tuturuus with weights  $\{1, 2, 4\}$ . Hence, the full set  $S = \{1, 2, 4\}$ . Notice that active Tuturuu with weight 1 can be produced by both inactive Tuturuus with weight 2 or 4.

Since Okabe needs the help of Kurisu, he will need to give the active Tuturuus with weights in  $S$  to Kurisu as well. Hence, for each weight in the set  $S$ , he will need to produce *at least 2 (TWO) active Tuturuus*. Can you help to determine the minimum number of reagent drops that Okabe needs?

For example, if  $N = 2$  and  $W = \{2, 4\}$ , we have  $S = \{1, 2, 4\}$ . The minimum number of drops is 3. Instead of dropping the reagent twice for each inactive Tuturuu (so we have  $\{1, 2\}$ ,  $\{1, 2\}$ ,  $\{1, 4\}$ ,  $\{1, 4\}$ , and have 4/2/2 active Tuturuus of weight 1/2/4, respectively), we can just drop the reagent twice for inactive Tuturuu 4 with  $d = 1$  (so we have  $\{1, 4\}$ ,  $\{1, 4\}$ ), then we drop the reagent one more time for inactive Tuturuu 4, but this time with  $d = 2$  (remember that Okabe can freely choose  $d$ ), so we have another  $\{2, 2\}$ . This way, we have 2/2/2 active Tuturuus of weight 1/2/4, respectively, with just 3 drops.

**D.1 Three Manual Test Cases (3 marks)**

Output the required answer and a short explanation (1 mark each) if you are given:

1.  $N = 1, W = \{14\}$ .
2.  $N = 4, W = \{2, 3, 5, 7\}$ .
3.  $N = 3, W = \{2, 3, 6\}$ .

**D.2 Create S Given W in  $O(N)$  (4\* marks)**

One of the important sub-routine to solve this task is to produce the full set  $S$  given set  $W$  of  $N$  inactive Tuturuus. As  $1 \leq W_i \leq 900$ , one can just use a simple factoring algorithm. Design an algorithm to produces the full set  $S$  in  $O(N \times \sqrt{900}) \approx O(N)$ .

**D.3 Do you know the actual problem name? (1 mark)**

If you know the real name of this problem, mention it.

Hint: It is MIN-XXXXXXXXXX-CXXXXXXXXXX.

**D.4 Is this an NP-hard optimization problem? (1 mark)**

If you say it is, give a short justification, and you are allowed to use an exponential algorithm for Subsection D.5. If you say it is not, you must use a polynomial solution for Subsection D.5.

**D.5 Solve This Problem (10 marks)**

One of the route in Subsection D.4 is correct.

The correct route worth the full (10) marks.

The other one worth only 5 marks (2-opt of the optimal marks).

**D.6 At Least 3 Active Tuturuus (6\* marks)**

If the problem statement is changed into the following: “Since Okabe needs the help of Kurisu and Mayushii (Okabe’s other friend who also like Tuturuus), he will need to give the active Tuturuus with weights in  $S$  to Kurisu and Mayushii as well. Hence, for each weight in the set  $S$ , he will need to produce *at least 3 (THREE) active Tuturuus.*”

Does this ‘small’ change (from at least 2 (TWO) active Tuturuus to at least 3 (THREE) active Tuturuus) change your answer from Section D.4 and D.5?

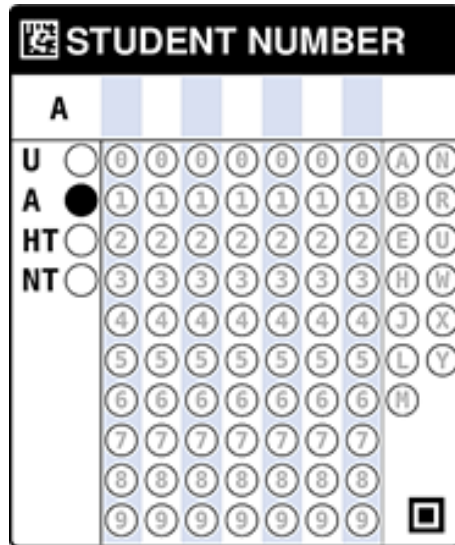
If yes, elaborate the required change(s) and re-solve the updated problem.

If no, elaborate why your previous solution for Section D.5 still works for this version.

Note that the grading scheme of this last question is very strict: 0/1/6 for wrong answer/blank/correct answer. There is no partial marks.

# The Answer Sheet

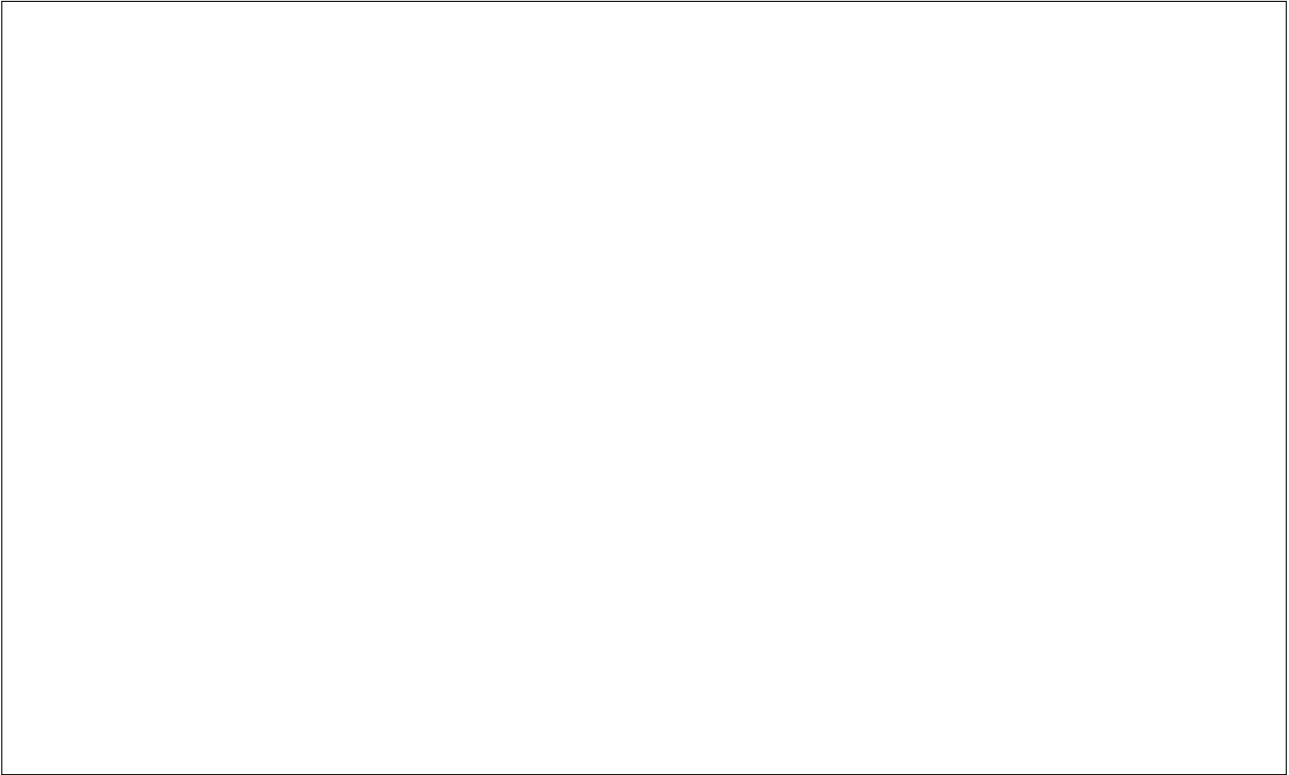
Write your Student Number and MCQ answers in the boxes below using **(2B) pencil**:



No	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

No	16	17	18	19	20	21
A	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

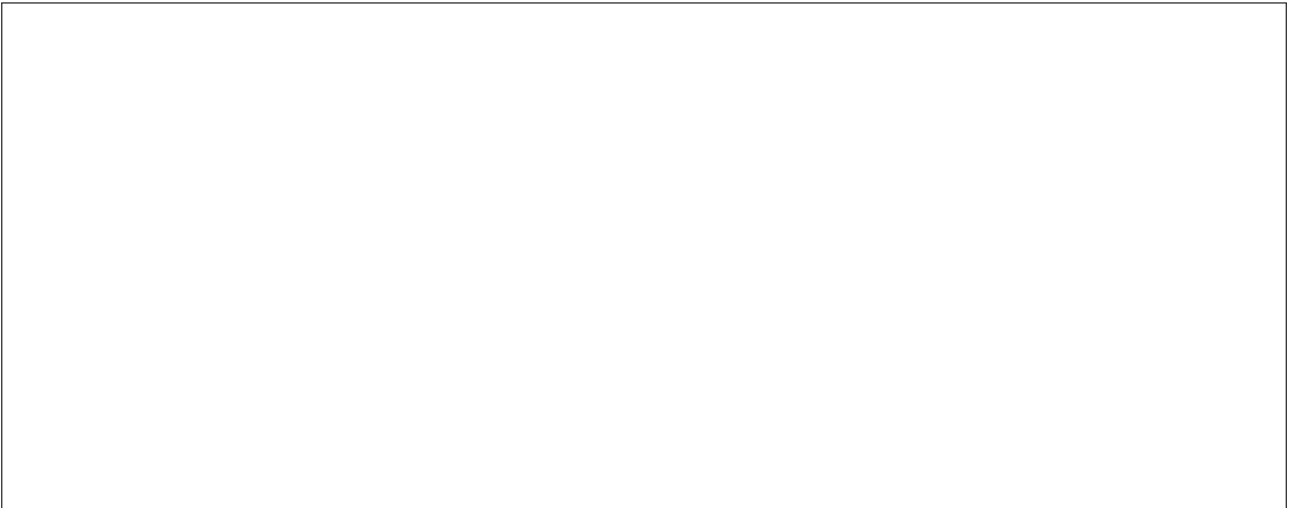
Box B. Draw the graph and the quick sketch of the  $\frac{n-1}{2}$  blossom shrinking processes.



Box C.1. 3 test cases (write 3 different integers + explanation in C.1.1, C.1.2, and C.1.3 order)



Box C.2. Formulate as ILP



---

Box C.3.\* (1 if blank, 0 if wrong) Prove SETTING-PROBLEMS is NP-hard

Box C.4. Write Your Best Complete Search

Box C.5.\* (1 if blank, 0 if wrong) Solve a Special Case

Box C.6. Local Search Ideas

Box D.1. 3 test cases (write 3 different integers + explanation in D.1.1, D.1.2, and D.1.3 order)

Box D.2.\* (1 if blank, 0 if wrong) Create S Given W in  $O(N)$

Box D.3. Say the actual problem name

Box D.4. NP-hard?

Box D.5. Solve This Problem

Box D.6.\* (1 if blank, 0 if wrong) At Least 3 Active Tutorials

– END OF PAPER; All the Best –