Algorithm Design - First Midterm

12-02-2025, *Time: 120 minutes*

Prof. Aris Anagnostopoulos, Prof. Stefano Leonardi, Dr. Simone Fioravanti

Instructions:

- You are given the question sheet and 4 sheets for your answers. The problems are 4.
- Before you start, write on the top of each sheet (1) your full name, (2) your student ID (*matricola*) and (3) the sheet number (1, 2, 3, and 4).
- If you use additional sheets as rough draft (*brutta*) for calculations, etc., you must hand it, along with the rest of the exam. You can keep the question sheet.
- Hand in the sheets in order **without attaching the sheets with each other.**
- When your pages are full, you can ask for extra pages; make sure that you also write there your name, student id, and the sheet number (5, 6, etc.). You must hand all the sheets that you receive.
- The rooms are small so it may be tempting to copy. Note that if we catch you copying in any way (copying from another student, using your phone, etc.) you will be automatically disqualified.

Problem 1

A group of employees of a company is planning to have dinner together. To increase social interaction, they would like to sit at tables such that no two employees from the same team are at the same table. For that purpose, assume there are employees from k different teams, and that there are $n_1, \ldots, n_k$ members of each team signed up for the dinner. Moreover, assume there are t available tables and that the j-th table has r_j seats, for $j = 1, \ldots, t$. Let us define the *seating arrangement* as the decision problem returning true if it is possible to distribute the employees of different teams to different tables and false otherwise.

1. Formulate this problem as a maximum flow problem and write down the condition that should hold whenever the original decision problem returns true;
2. Give the runtime it takes to solve the corresponding flow problem in terms of k, t, n_i, r_j for $i = 1, \ldots, k$, $j = 1, \ldots, t$.

Problem 2 maximize total weight

Given an undirected **line** graph $G = (V, E)$ of n vertices and and $n-1$ edges, and non negative weights $w(v), v \in V$, an independent set $S \subseteq V$ is a set of mutually non-adjacent vertices such that, for each pair $u, v \in S$, edge $(u, v) \notin E$. The total weight of an independent set S is $\sum_{v \in S} w(v)$.

1. Find an efficient algorithm that solves the problem.
2. Show a pseudo code for the algorithm.
3. Analyze its time complexity.

Problem 3

Consider an undirected graph $G = (V, E)$. The Max-Cut problem is the one of finding a partition $S, V \setminus S$ of V such that the total number of edges with one vertex in S and the other vertex in $V \setminus S$ is maximized.

1. Analyze the approximation ratio of the algorithm that assigns at random with probability $1/2$ each vertex to one of the two sides of the partition.
2. (Bonus) Give a reduction for Max-Cut that shows that it is NP-hard.
3. Analyze the running time of the algorithm.
4. (Bonus) Derive a deterministic version of the algorithm that obtains the same approximation ratio.

Problem 4

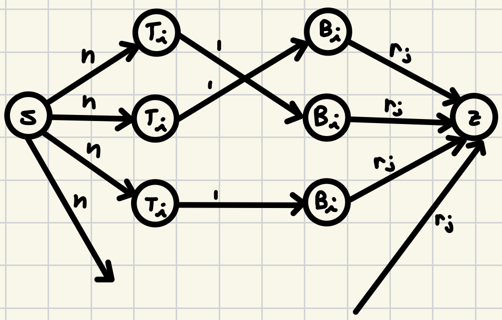
In the *subset-sum problem* we are given as input a set of n nonnegative integers $X = \{x_1, x_2, \ldots, x_n\}$, and a positive integer T. The problem is whether there exists a subset $S \subseteq X$ such that $\sum_{x \in S} x = T$. Show that the subset-sum problem is NP-complete. You can assume that you know that the problem of *3D-matching* is NP-complete.

Problem 1

A group of employees of a company is planning to have dinner together. To increase social interaction, they would like to sit at tables such that no two employees from the same team are at the same table. For that purpose, assume there are employees from k different teams, and that there are $n_1, \dots, n_k$ members of each team signed up for the dinner. Moreover, assume there are t available tables and that the j-th table has r_j seats, for $j = 1, \dots, t$. Let us define the *seating arrangement* as the decision problem returning true if it is possible to distribute the employees of different teams to different tables and false otherwise.

1. Formulate this problem as a maximum flow problem and write down the condition that should hold whenever the original decision problem returns true;
2. Give the runtime it takes to solve the corresponding flow problem in terms of k, t, n_i, r_j for $i = 1, \dots, k$, $j = 1, \dots, t$.

a) WE BUILD A GRAPH WITH A SUPER SOURCE s AND A SUPERSINK z. A NODE FOR EACH TEAM T_i, AND ONE FOR EACH TABLE B_i.



$s \to T_i$ HAS CAPACITY n (MEMBERS)
$T_i \to B_i$ HAS CAPACITY 1 (MAX ONE OF THE TEAM PER TABLE)
$B_i \to z$ HAS CAPACITY r_j (SEATS)

THE PROBLEM RETURN TRUE IFF THERE IS MAX FLOW EQUAL TO THE SUM OF ALL MEMBERS n_i

b) NUMBER OF NODES $\to$ $|V| = k + t + 2$
NUMBER OF EDGES $\to$ $|E| = k + kt + t$

MAX FLOW $F = \sum_{i=1}^{k} n_i$

IF WE USE FORD-FULKERSON $\to$ $O(|E| \cdot F)$

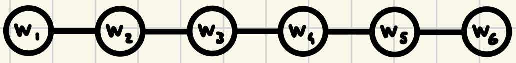
EACH AUGMENTATION ADDS AT LEAST 1 TO F

IF WE USE EDMONDS-KARP $\to$ $O(|V| \cdot |E|^2)$

Problem 2 Maximize total weight

Given an undirected **line** graph $G = (V, E)$ of n vertices and and $n-1$ edges, and non negative weights $w(v), v \in V$, an independent set $S \subseteq V$ is a set of mutually non-adjacent vertices such that, for each pair $u, v \in S$, edge $(u, v) \notin E$. The total weight of an independent set S is $\sum_{v \in S} w(v)$.

1. Find an efficient algorithm that solves the problem.
2. Show a pseudo code for the algorithm.
3. Analyze its time complexity.



a) OPT (i) = BEST SOLUTION USING $v_1 \dots v_i$

$$v_i \begin{cases} OPT(i-1) & \text{DON'T CHOOSE } v_i \\ w_i + OPT(i-2) & \text{CHOOSE } v_i \end{cases}$$

$$OPT(i) = \max(OPT(i-1),\ w_i + OPT(i-2))$$

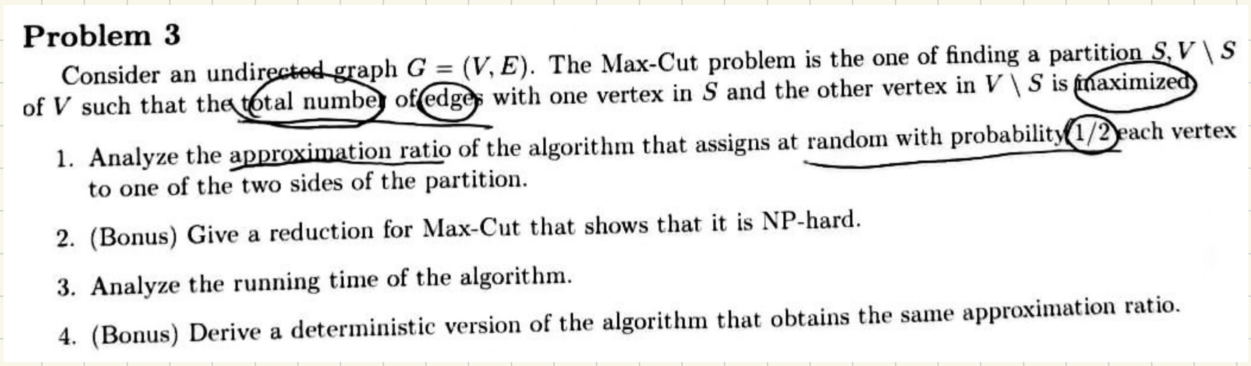
b) ALG:

```
OPT [0] = 0
OPT [1] = w [1]

FOR i = 2 .. n
    OPT [i] = max (OPT [i-1], w[i] + OPT [i-2])

RETURN OPT [n]
```

c) $O(n)$



Problem 3

Consider an undirected graph $G = (V, E)$. The Max-Cut problem is the one of finding a partition $S, V \setminus S$ of V such that the total number of edges with one vertex in S and the other vertex in $V \setminus S$ is maximized.

1. Analyze the approximation ratio of the algorithm that assigns at random with probability 1/2 each vertex to one of the two sides of the partition.
2. (Bonus) Give a reduction for Max-Cut that shows that it is NP-hard.
3. Analyze the running time of the algorithm.
4. (Bonus) Derive a deterministic version of the algorithm that obtains the same approximation ratio.

a) WE ASSIGN EACH NODE RANDOMLY IN S OR IN T WITH PROBABILITY $\frac{1}{2}$.
THE CONTRIBUTION OF AN EDGE e IS:

$$X_e \begin{cases} 1 & \text{IF } e \text{ IS IN THE CUT } (u \in S, v \in T) \\ 0 & \text{OTHERWISE} \end{cases}$$

$$Pr[u \in S, v \notin S] = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \qquad Pr[X_e = 1] = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$

$$Pr[u \notin S, v \in S] = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \qquad E[X_e] = \frac{1}{2}$$

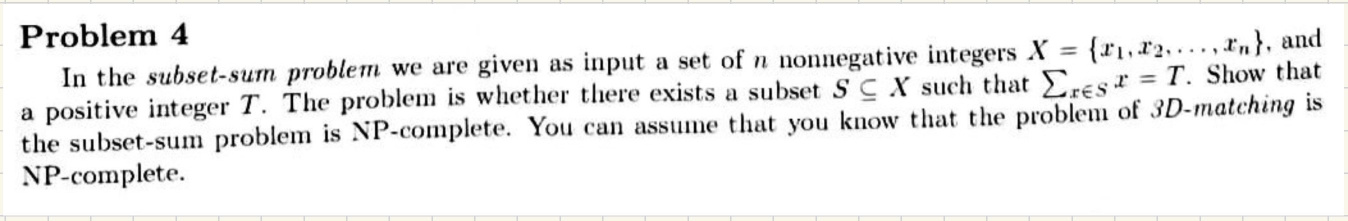
$$E[X] = \sum_{e \in E} E[X_e] = \sum_{e \in E} \frac{1}{2} = \frac{|E|}{2}$$

AN OPT CAN'T CUT MORE THAN $|E|$ EDGES $\rightarrow$ $OPT \leq |E|$

$$E[X] = \frac{|E|}{2} \leq \frac{1}{2} OPT$$

b) MAX CUT IS NP HARD SINCE IT'S DECISIONAL VERSION IS NP-COMPLETE.

c) ASSIGNING EACH VERTEX TO A SIDE TAKES $O(|V|)$, WHILE CALCULATING HOW MANY EDGES ARE CUTTED TAKES $O(|E|) \rightarrow O(|V| + |E|)$



Problem 4

In the *subset-sum problem* we are given as input a set of n nonnegative integers $X = \{x_1, x_2, \ldots, x_n\}$, and a positive integer T. The problem is whether there exists a subset $S \subseteq X$ such that $\sum_{x \in S} x = T$. Show that the subset-sum problem is NP-complete. You can assume that you know that the problem of *3D-matching* is NP-complete.

SUBSET SUM IS IN NP SINCE WE CAN CHECK, IN POL TIME, THAT A SUBSET S HAS TOTAL SUM EQUAL TO A GIVEN VALUE K.

3D MATCHING $\leq_p$ SUBSET SUM

WE FIX A BASE $B > q$. FOR EACH TRIPLE (x_i, y_j, z_k) WE BUILD A NUMBER THAT HAS 1 IN THE 3 TRIPLE'S POSITIONS, AND 0 ELSEWHERE. THE TARGET IS THE TRIPLE $(1,1,1)$. SINCE $B > q$, THERE ARE NOT CARRIES. SO THERE IS A SUBSET SUM EQUAL TO q IFF THERE EXISTS A PERFECT MATCHING.