General rules: In solving these questions you may consult books or other available notes, but you need to provide citations in that case. You can discuss high level ideas with each other, but each student must find and write his/her own solution. Copying solutions from any source, completely or partially, allowing others to copy your work, will not be tolerated, and will be reported to the disciplinary office.

You should upload the pdf file (either typed, or a clear and readable scan) of your solution to mycourses.

There are in total 22 points, but your grade will be considered out of 20.

1. (4 points) Consider the following problem: We are given a number $r$ and a maximization linear program $LP$ (say in the standard form) as follows:

$$\begin{align*}
\text{max} & \quad c^T \cdot \vec{x} \\
\text{s.t.} & \quad A \vec{x} \leq \vec{b} \\
& \quad \vec{x} \geq 0
\end{align*}$$

That is the input is $[A,c,b,r]$, where $A$ is an $m \times n$ matrix, $b$ is an $m$-dimensional vector, and $c$ is an $n$-dimensional vector, and $r$ is a number, and we want to know if $\text{Opt} (LP) \geq r$?

Without using the fact that Linear Programming can be solved efficiently using the ellipsoid method show that this problem belongs to both NP and CoNP.

Solution To show that this problem is in NP, note that a certifier can take $[A,c,b,r]$, and a vector $x$. It can check in polynomial time to see whether $x$ is a feasible solution and has cost at least $r$. If both conditions are satisfied then it outputs YES and otherwise outputs NO. Note that if $[A,c,b,r]$ is a YES input (meaning $\text{Opt} (LP) \geq r$), then there is a solution $x$ that will pass the certifiers test, and if it is a NO input (meaning $\text{Opt} (LP) < r$) then no $x$ can pass the test.

To show that this problem is in CoNP, we look at the dual. We need to find a way to certify the NO answers. How can we prove that $\text{Opt} (LP)$ is LESS than $r$? The certifier takes an $(m$-dimensional) vector $y$, and it outputs YES it $y$ is a feasible solution to the dual and has cost (in the dual) less than $r$. Note that if such a $y$ exists, then $\text{Opt} (Dual) < r$, and thus by the weak duality $\text{Opt} (LP) \leq \text{Opt} (Dual) < r$ showing that $[A,c,b,r]$ is a NO input for our problem.
On the other hand by strong duality, if \([A,c,b,r]\) is a YES input then such a \(y\) must exist (since \(\text{Opt}(\text{Dual}) = \text{Opt}(\text{LP}) \geq r\)).

2. We are given the coordinates of \(n\) points \(p_1, \ldots, p_n\) on the plane. We want to find non-overlapping disks centred on these points such that the sum of their radii is maximized.

(a) (2 points) Show that this problem can be solved as a linear program.

**Solution:**

\[
\begin{align*}
\text{max} & \quad r_1 + \ldots + r_n \\
\text{s.t.} & \quad r_i + r_j \leq d_{ij} \quad \forall i \neq j \in \{1, \ldots, n\} \\
& \quad r_i \geq 0 \quad \forall i = 1, \ldots, n
\end{align*}
\]

where \(d_{ij} = d(p_i, p_j)\).

(b) (4 points) Consider a complete directed graph on the points \(p_1, \ldots, p_n\) where for every \(i\) and \(j\), there is an edge from \(p_i\) to \(p_j\), and one edge from \(p_j\) to \(p_i\), both with a costs equal to \(d(p_i, p_j)\), the distance between the two points. We want to cover the vertices of this graph with cycles (that is every vertex belongs to at least one cycle) so that the total sum of the cost of the edges that participate in these cycles is minimized (Note that we also allow cycles with two edges).

Use the linear programming duality to prove that the solution to this problem is at least twice the solution to the non-overlapping disk problem.

**Solution:** We write the dual of the previous problem. The variables are \(y_{ij}\) for all \(i \neq j\) (the order of the indices do not matter , and \(y_{ij}\) and \(y_{ji}\) denote the same variable).

\[
\begin{align*}
\text{min} & \quad \sum_{i \neq j} d_{ij} y_{ij} \\
\text{s.t.} & \quad \sum_{j : j \neq i_0} y_{i_0 j} \geq 1 \quad \forall i_0 \in \{1, \ldots, n\} \\
& \quad y_{ij} \geq 0 \quad \forall i, j
\end{align*}
\]

Consider an optimal solution to the cycle problem, and let \(C\) be its cost. Set \(y_{ij} = 1/2\) if the edge \(ij\) belongs to any of the cycles, and set \(y_{ij} = 0\) otherwise. Then first note that \(\sum_{i \neq j} d_{ij} y_{ij} = C/2\) (we are counting \(d_{ij}/2\) for each participating edge).

Moreover because every vertex \(i_0\) has at least two edges incident to it that participate in the cycles, the constraints \(\sum_{j : j \neq i_0} y_{i_0 j} \leq 1\) are all satisfied. Thus we have a feasible solution to the dual linear program whose cost is \(C/2\).

Now weak duality tells us that the optimal solution to the non-overlapping problem (formulated as the LP) is at most the optimal to the dual which we just showed that is at most \(C/2\).

3. (4 points) We are given an \(n \times m\) table. The rows correspond to \(n\) students and the columns correspond to their favourite things (For example, food, book, sport, composer, movie, etc). For example if the entry in the food column of the \(i\)-th row is Samosa, it means that Samosa is the favourite food of the \(i\)-th student. We are also given a number \(k\), and our goal is to select \(k\) students so that every two of them share at least one favourite thing in life. Prove that this problem is NP-complete.

**Solution** The problem is clearly in NP. A certifier can take a set of students, and verify (in polynomial time) whether this set is of size \(k\), and that every two of them share at least one common element in some column. If these conditions are satisfied it outputs YES, otherwise NO.

To prove completeness we reduce the MAX-CLIQUE problem to this. Consider an input \((G, k)\) to the max clique problem, where we want to know whether a graph \(G\) has a clique of size \(k\).
We construct a matrix $M$ in the following manner. We set $n = |V|$ and $m = |E|$. The rows are indexed by the vertices of $G$, and the columns are by the edges, and we set $M[u, e] = 1$ if the $u$ is one of the endpoints of $e$. If not, we set $M[u, e] = 0$ to a unique element that does not appear anywhere else in the table (for example take $M[u, e] = (u, e)$ where $(u, e)$ is a symbol).

Now note that two rows have a common element in some column if and only if they are connected by an edge. Thus a set of $k$ students with pairwise common shared interests is exactly the same as a $k$ clique in $G$. Thus if the oracle for the $k$ student problem returns YES on $M$ we know that $G$ has a $k$-clique and if it returns No, then we know it does not have a $k$-clique.

4. (4 points) We are given $n$ numbers (each in decimal representation), and a number $k$. We want to see if it is possible to select $k$ of these numbers so that if we add them up, the digit 2 will not appear in the decimal representation of the sum. Prove that this problem is NP-complete.

Solution: The problem is clearly in NP. The certifier takes a subset of these numbers, and and verifies (in polynomial time) if there are $k$ of them and their sum does not have digit 2.

To prove completeness we reduce Independent Set Problem to this. Consider an input $(G, k)$ to the independent set problem. Suppose that $G$ has $n$ vertices and $m$ edges. Let $M$ be an $n \times m$ matrix where the rows are indexed by the vertices of $G$ and columns are indexed by the edges of $G$. We set $M[u, e] = 1$ if $u$ is an endpoint of $e$ and $M[u, e] = 0$ otherwise. Now read every row of $M$ from left to right as the digits of a number. Let these numbers be $a_1, \ldots, a_n$ (each number has at most $m$ digits). At most because some rows might start with zeros). Note that for every digit, there are only two numbers that have 1 in that digit and all the other numbers have 0’s. As a result if we sum up a subset of these numbers, we will see a digit 2 if and only if we pick both end points of an edge. This shows that $G$ has an independent set of size $k$ if and only if we can select $k$ of these numbers such that the number 2 will not appear in any of the digits.

5. (4 points) Prove that the following problem is NP-complete.

- **Input**: A CNF $\phi$.
- **Question**: Does $\phi$ have a truth assignment that assigns True to exactly half the terms in each clause?

(For example $x_1 = T, x_2 = F, x_3 = F, x_4 = F$ assigns True to half the terms in each clause of $\phi = (x_1 \lor \overline{x_2} \lor x_3 \lor x_4) \land (x_1 \lor x_2)$.

Solution: The problem is clearly in NP. The certifier takes a truth assignment, and verifies (in polynomial time) if it assigns True to half the terms in each clause, and outputs YES or NO accordingly. Note that if $\phi$ is a YES input there is a certificate that will be accepted, and if $\phi$ is a NO input, no certificate can pass the test.

To prove completeness we reduce 3SAT to this problem. Consider an input $\psi$ to 3SAT. We will construct a 6CNF $\phi$ from $\psi$ by adding 3 variables to each clause.

Some intuition: Let $(t_1 \lor t_2 \lor t_3)$ be a clause of $\psi$, and $(t_1 \lor t_2 \lor t_3 \lor y_1 \lor y_2 \lor y_3)$ be the new clause. Note that if $(t_1 \lor t_2 \lor t_3)$ is satisfied, then 1, 2, or 3 terms out of $t_1, t_2, t_3$ are True, which means that we want, respectively, 2, 1, or 0 of $y_1, y_2, y_3$ to be True. If we could manage to force say $y_3$ to False. We would be done as then $(t_1 \lor t_2 \lor t_3)$ is satisfied if and only if it is possible to get half the terms in $(t_1 \lor t_2 \lor t_3 \lor y_1 \lor y_2 \lor y_3)$ to be satisfied. Even though there is no way to force a variable to False, let us have a variable $z$, and in out mind think of it to be False. We will deal with this issue later.

The actual reduction: Introduce one special variable $z$ and then for each clause $C_j = (t_1 \lor t_2 \lor t_3)$ introduce two specific variables $y_{j1}$ and $y_{j2}$ for this clause, and replace $C_j$ with $(t_1 \lor t_2 \lor t_3 \lor y_{j1} \lor y_{j2} \lor z)$. Call this new formula $\psi$. Note that if $\psi$ is satisfiable by a truth assignment, then
we can take that truth assignment, set \( z = \text{False} \), and an appropriate number of \( y_{j1}, y_{j2} \) to \text{True}

and have exactly half the terms in each clause of \( \phi \) to be \text{True}. This shows

\[ \psi \text{ is satisfiable } \Rightarrow \phi \text{ is a YES input.} \]

However in order to be able to deduce anything about the satisfiability of \( \psi \) from the output of the oracle on \( \phi \) we also need to prove the opposite direction:

\[ \phi \text{ is a YES input } \Rightarrow \psi \text{ is satisfiable.} \]

Suppose that \( \phi \) is a YES input: There is a truth assignment that assigns \text{true} to half the terms in each clause. Note that if an assignment satisfies this property, then if we flip the value of each variable then it will still satisfy the property (all \text{true} terms become \text{false}, and all the \text{false} terms become \text{true}, so still half of each clause is \text{true}). So consider the assignment, and if \( z = \text{True} \), then flip all the variables to make \( z = \text{False} \). Now that \( z = \text{False} \), note that this assignment has to satisfy \( \psi \) as it has to assign at least one \text{true} to each \( (t_1 \lor t_2 \lor t_3) \) in order to get 3 of the 6 terms to be \text{true} because \( y_{j1}, y_{j2} \) can add at most 2 \text{true}s to each clause. We showed that

\[ \psi \text{ is satisfiable } \iff \phi \text{ is a YES input.} \]

Hence we can ask the oracle to see if our constructed \( \phi \) is a \text{YES} or a \text{NO} input and from that deduce whether \( \psi \) is satisfiable or not.