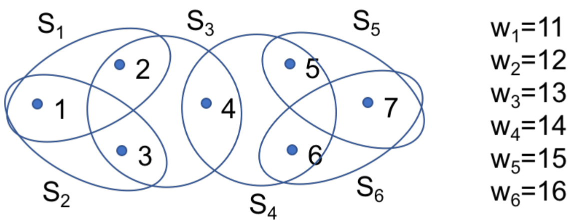
Exam Combinatorial Optimization, 25 October 2019.

- It is not allowed to use any books, notes, or calculator. Just pen and paper.
- The table shows the maximum number of points per (sub-)question.
- You may also obtain points for partial solutions.
- When you give an algorithm or proof, then don't spend too much time on writing a very detailed answer. Just describe the main idea.

1a	1b	1c	1d	2	3	4a	4b	4c	5a	5b	5c	5d	6a	6b	6c	6d	6e	Σ
1	3	3	1	8	8	4	1	4	1	3	2	2	3	1	1	2	2	50

1. The figure shows an instance of the *weighted* Set Cover problem.



(a) Give an optimal solution and its value.

(b) Give the ILP formulation for this set cover instance.

(c) Give the the dual (of the formulation in (b)) for this set cover instance.

(d) Each element is in exactly two sets. Thus, this is actually an instance of the Vertex Cover problem. Draw the corresponding graph.

The book gives the following approximation algorithm for Set Cover.

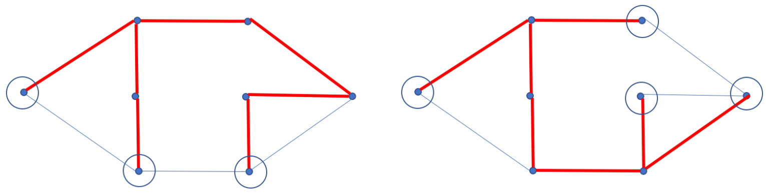
Set Cover Algorithm:
Step 1: Solve the dual.
Step 2: For each set, add it to the solution if the corresponding inequality in the dual is tight. (Remember, tight means that we have equality.)

(e) Explain why any solution produced by this algorithm is a feasible set cover. (In general, not just for the given instance).

2. In the Minimum Leaves Spanning Tree problem we need to find a spanning tree with a smallest number of leaves. (A leaf is a point of degree 1)

MINIMUM LEAVES SPANNING TREE:

Instance:	Graph $G = (V, E)$.
Solution:	A spanning tree T of G.
Value:	The number of leaves of T.
Goal:	Minimize value.



Show that there is no α-approximation algorithm for this problem for $\alpha < 3/2$, assuming that P$\neq$ NP.

3. The minimum covering problem is defined as follows:

MINIMUM COVERING PROBLEM:

Instance:	Positive numbers $a_1, a_2, \ldots, a_n$ and a number B.
Solution:	$S \subseteq \{1, 2, \ldots, n\}$ such that $\sum_{i \in S} a_i \geq B$.
Value:	$\sum_{i \in S} a_i$.
Goal:	Minimize value.

In other words, in the Minimum Covering Problem we need to find a subset of the given numbers of minimum total sum under the constraint that this sum is at least some value B.

Give a Polynomial Time Approximation Scheme (PTAS) for this problem. (Describe your algorithm, and argue that it is indeed a PTAS.)
Hint: Rounding of numbers is not needed. Almost the same problem was one of the exercises, there with $\leq$ in stead of $\geq$. Use the same idea.

4. The uncapacitated facility location problem as described in Chapter 4:

UNCAPACITATED FACILITY LOCATION (UFL):

Instance:	Set of points F (facilities) and set of points D (clients). Opening cost f_i for each $i \in F$ and connection cost (assignment cost) c_{ij} for each pair $i \in F, j \in D$. The connection cost is assumed to be metric.
Solution:	$F' \subseteq F$.
Value:	$\sum_{i \in F'} f_i + \sum_{j \in D} \min_{i \in F'} c_{ij}$
Goal:	Find a solution of minimum cost.

(a) Give an ILP-formulation of the uncapacitated facility location problem.

Assume from now that for each client j there are only three facilities to which it can possibly connect. More precisely, for $j = 1, \dots, n$ we are given a set $S_j \subset F$ with $|S_j| = 3$ and client j must be connected to one of the facilities in S_j.

(b) What change do you make to the ILP of (a) to incorporate this constraint?

(c) Give a 3-approximation algorithm for UFL with this extra restriction by LP-rounding. (Describe the algorithm. Show that the solution is feasible and show that the value is at most 3 times the LP-value.)

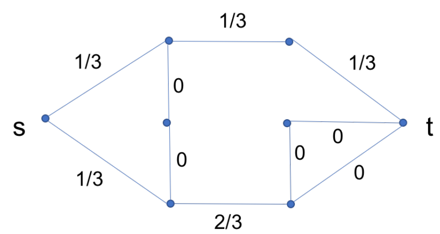
5. In the minimum s,t-cut problem we are given a graph $G = (V, E)$ and $s, t \in V$ and we need to find a smallest set of edges $W \subseteq E$ such that removing W separates s from t (that means s and t end up in different components). This problem can be solved by a max flow algorithm. Here, we shall use a different approach using LP-rounding.

Let $\mathscr{P}$ be the set all simple paths from s to t in the graph. (A path is simple if no vertex is visited more than once by the path.) Observe that a set $W \subseteq E$ is an s,t-cut if and only if every path P in $\mathscr{P}$ contains at least one edge from W.

For every edge $(u, v) \in E$ introduce a variable x_{uv}. The following ILP is an exact formulation of the minimum s,t-cut problem.

$$
\begin{aligned}
\text{(ILP)} \quad \min \quad & Z = \sum_{(u,v) \in E} x_{uv} \\
s.t. \quad & \sum_{(u,v) \in P} x_{uv} \geqslant 1 && \text{for all } P \in \mathscr{P}. \\
& x_{uv} \in \{0, 1\} && \text{for all } (u, v) \in E.
\end{aligned}
$$

In the LP-relaxation, we take $x_{uv} \geqslant 0$ in stead of $x_{uv} \in \{0, 1\}$. The figure shows an example of an LP-solution.



NB. The figure is just an example. All questions below refer to the problem in general.

(a) Explain why it is not immediately clear that the LP-relaxation can be solved in polynomial time.

The LP-relaxation can be solved in polynomial time by using the ellipsoid method together with a separation oracle.

(b) Describe what a separation oracle does in general for an LP. Further, show that this LP-relaxation has a polynomial time separation oracle.

Let x^* be an optimal solution for the LP-relaxation and let Z^* be its value.

- For every edge $(u,v) \in E$, define its *length* as x^*_{uv} (its LP-value).
- Using these lengths, define $L(v)$ as the distance from s to v in G, for every $v \in V$. In other words, $L(v)$ is the length of the shortest path from s to v in G, using the LP-values as lengths of the edges.
- For any γ with $0 \leq \gamma \leq 1$, define the set of vertices $S_\gamma = \{v \in V \mid L(v) \leqslant \gamma\}$, that means, S_γ is the set of vertices at distance at most γ from s.

Min cut algorithm

Step 1: Solve the LP-relaxation $\rightarrow$ x^*, Z^* .
Step 2: Take $\gamma \in [0,1]$ uniformly at random. $\rightarrow$ S_γ.
Step 3: Return the set of edges that have exactly one endpoint in S_γ. Denote this set by W.

(c) Show that W has at most Z^* edges in expectation, that means, $\mathbb{E}(|W|) \leqslant Z^*$.
Hint: Show that for any edge (u,v), the probability that it is in the cut is at most x^*_{uv}.

(d) Explain how the algorithm can be derandomized.
(*This question can be answered even if you did not find an answer for (c).*)

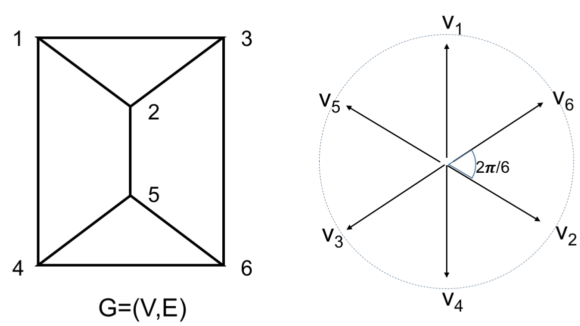
6. **(a)** Give the Quadratic Program for the maximum cut problem.

(b) Give the Vector Program relaxation of the formulation given in (a).

(c) For the graph below, give a set $S \subset V$ such that the number of edges between S and $V \setminus S$ is maximum. (In other words, give a maximum cut.) Argue why it is a maximum cut.

(d) The figure shows a possible solution to the Vector Program of (b) for this graph. What is the value of this solution for the Vector Program? (Show your computation.)

(e) Remember that the the randomized rounding algorithm first computes an optimal solution to the VP and then rounds that VP-solution by taking a hyperplane uniformly at random. Assume that, in stead of taking an optimal VP-solution, we apply this algorithm with the VP-solution shown. Then what is the expected number of edges in the cut? (Show your computation.)



Appendix

The following problems can be solved in polynomial time:

- Linear programs and Vector programs.
- Finding a minimum cut in a graph.
- Finding the shortest path between two points in a graph.
- Finding a minimum spanning tree in a graph.
- Finding a maximum matching in a graph.

The following problems are NP-hard/NP-complete:

- Integer Linear programming
- Hamiltonian Cycle
- Hamiltonian Path
- Vertex Cover
- Set Cover
