MAX PLANCK INSTITUTE FOR INFORMATICS

SIC Saarland Informatics Campus

Lecturers: **Evangelos Kipouridis and Tomasz Kociumaka** **Winter 2025/26**
Tutors: **Anouk Duyster and Yonggang Jiang**

Randomized and Approximation Algorithms, Exercise Sheet 9

https://www.mpi-inf.mpg.de/departments/algorithms-complexity/teaching/winter-2025/26/randomized-and-approximation-algorithms

Total Points: **100** + 35 bonus points Due: **14:00** on Thursday, **January 8**, 2026

Exercise 1 — 5 bonus points

Report how many work-hours were required, from the moment you opened this file to the moment you submitted your solutions.

Exercise 2 — 10 + 15 + 15 + 35 points

A trendy new streaming platform is trying to launch in a hurry. They have licensed a large catalog of shows, and for each show they know which users would be excited enough to subscribe if that show were featured prominently on the front page.

The platform can feature exactly k shows at launch. A user will subscribe if *at least one* of their favorite shows is featured. The growth team wants to pick k shows that will attract as many users as possible.

Formally, you are given:

- a universe U of users,
- a collection of subsets $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$, where each $S_i \subseteq U$ is the set of users who would subscribe if show i were featured,
- an integer k.

The goal is to select $X \subseteq \mathcal{S}$ with $|X| = k$, maximizing $\left|\bigcup_{S \in X} S\right|$.

Task. We want to design a polynomial-time $\frac{1}{2}$ approximation for the problem. Proceed as follows: start with an arbitrary selection of k shows. As long as you can "swap" a show you have selected with an unselected show and increase the attracted number of users, perform this swap. Let $\{j_1, \ldots, j_k\}$ be the k shows the algorithm outputs, A_i be the set of users satisfied by show j_i, $\{j'_1, \ldots, j'_k\}$ be the k shows of an optimal solution, and O_i be the set of users satisfied by show j'_i. Finally, let $A = \bigcup_{i=1}^{k} A_i$ and $O = \bigcup_{i=1}^{k} O_i$.

a Prove that the above local-search algorithm terminates in polynomial time.

b Show that $|A| \geq \sum_{i=1}^{k} \left(|A| - |\bigcup_{j \neq i} A_j|\right)$.

c Show that $\sum_{i=1}^{k} \left(|A \cup O_i| - |A|\right) \geq |O| - |A|$.

d Use the previous facts, along with the local optimality of the output, to prove that $|A| \geq \frac{1}{2}|O|$.

Exercise 3 — 10 + 15 points + 30 bonus points

Correlation Clustering is arguably one of the most important clustering problems. It was introduced in 2002, and has been extensively studied ever since. It took twenty years to achieve a 2 approximation. In this exercise we show how a conceptually simpler local-search algorithm also achieves a 2 approximation.

The input in Correlation Clustering is a simple graph $G = (V, E)$. An edge uv denotes that u and v are similar, while a non-edge denotes that u and v are not similar.

We want to output a partition (also called a clustering) $\mathcal{C} = \{C_1, \ldots, C_{|\mathcal{C}|}\}$ of the nodes. We call each C_i a cluster, and let $\mathcal{C}(u)$ denote the (exactly one) cluster in $\mathcal{C}$ containing node u.

The goal is to minimize the number of edges across different clusters, plus the number of non-edges within a cluster. More formally, the cost of a clustering $\mathcal{C}$ is

$$cost(\mathcal{C}) = \big|\{uv \in E \mid \mathcal{C}(u) \neq C(v)\}\big| + \big|\{uv \notin E \mid \mathcal{C}(u) = \mathcal{C}(v)\}\big|$$

The goal is to find a clustering $\mathcal{C}$ that minimizes $cost(\mathcal{C})$.

Let $OPT = \{O_1, \ldots O_{|OPT|}\}$ denote an optimal clustering. Assume that in polynomial time you can obtain a set family $\mathcal{F} = \{S_1, \ldots, S_\ell\}$ such that:

- All S_j are subsets of V.
- For every optimal cluster O_i, there exists a j such that $O_i = S_j$.

We now describe a local search approach: Start from an arbitrary clustering. At each step of the algorithm, try to locally improve your current clustering by imposing some set from $\mathcal{F}$. Imposing a set S in a clustering $\mathcal{C} = \{C_1, \ldots, C_{|\mathcal{C}|}\}$ means that we first remove S from every cluster of $\mathcal{C}$, and then add S in the resulting clustering. More formally, imposing S in $\mathcal{C}$ gives us the clustering $\mathcal{C}' = \{C_1 \setminus S, \ldots, C_{|\mathcal{C}|} \setminus S, S\}$. We stop when there is no set from $\mathcal{F}$ we can impose to improve our current clustering.

Task. Prove that the aforementioned algorithm is a polynomial time 2 approximation for Correlation Clustering.

a Prove that the algorithm runs in polynomial time. You can assume that obtaining $\mathcal{F}$ can be done in polynomial time.

b Let $\mathcal{C}$ be a clustering, and $\mathcal{L}_i$ be the clustering we obtain when imposing OPT_i in $\mathcal{C}$. Prove that

$$cost(\mathcal{L}_i) = \big|\{uv \in E \mid \mathcal{C}(u) \neq C(v), u \notin OPT_i, v \notin OPT_i\}\big| + \big|\{uv \notin E \mid \mathcal{C}(u) = \mathcal{C}(v), u \notin OPT_i, v \notin OPT_i\}\big| + \big|\{uv \in E \mid u \in OPT_i, v \notin OPT_i\}\big| + \big|\{uv \notin E \mid u \in OPT_i, v \in OPT_i\}\big|$$

c Prove that the algorithm is a factor 2 approximation. *Hint:* Sum over all $cost(\mathcal{L}_i)$.