TD 9 : Algorithmes d'approximation, en ligne et probabiliste

1 k-Centre (Metric k-Center)

We are given a metric space (X, d) and points $\mathbb{P} = \{p_1, \ldots, p_n\} \subset X$. We want to choose k points $Y = \{y_1, \ldots, y_k\} \subset \mathbb{P}$ so that

$$\max_{p \in \mathbb{P}} d(p, Y)$$

is minimized.

Problem 1.1 (Warm-up)**.** For any $Y \subset \mathbb{P}$, if R is the minimum value such that $\mathbb{P}$ is covered by the union of the circles of radius R centered at points in Y, then show that $R = \max_{p \in \mathbb{P}} d(p, Y)$.

Then the question can be read as : minimize R such that there exists $Y \subset X$ such that $\mathbb{P}$ is covered by the union of the disks of radius R centered at points in Y.

Problem 1.2. Give a 2-approximation algorithm for the metric k-center problem.

Problem 1.3. Show that unless $\mathbf{P} = \mathbf{NP}$, there is no $(2 - \varepsilon)$-approximation algorithm for any $\varepsilon > 0$ that runs in time polynomial in $|\mathbb{P}|$. *Hint :* Use the Dominating Set problem.

> **Dominating Set problem :**
> Given an undirected graph $G = (V, E)$, a subset of vertices $D \subseteq V$ is called a *dominating set* if for every vertex $u \in V \setminus D$, there is a vertex $v \in D$ such that $\{u, v\} \in E$.
> The problem of deciding for a given k whether there exists a dominating set of size at most k is known to be **NP**-complete.

2 Problème de la location de skis (Ski Rental Problem)

You are going skiing in the mountains and you want to keep skiing until the day you crash badly. Renting a ski involves paying 1 unit of currency at the start of each day, and buying it involves paying $B > 1 \in \mathbb{N}$ units at the start of the day. Suppose you ski for x (entire) days, but x is not known in advance. What is the best strategy for buying (or not buying) the skis ?

3 Méthode probabiliste (The Probabilistic Method)

3.1 Basic argument

Problem 3.1 (Lower bounding Ramsey number $R(k, k)$)**.** If $\binom{n}{k} 2^{1-\binom{k}{2}} < 1$, then it is possible to color the edges of K_n with two colors such that it has no monochromatic K_k subgraph.

3.2 Expectation argument

Problem 3.2. Suppose we have a probability space $\mathcal{S}$ (with discrete probability distribution) and a random variable X defined on $\mathcal{S}$ such that $\mathbb{E}[X] = \mu$. Then show that $\mathbb{P}(X \geq \mu) > 0$ and $\mathbb{P}(X \leq \mu) > 0$.

Problem 3.3 (Existence of a large cut)**.** Given an undirected graph G with n vertices and m edges, show that there is a partition of V into two disjoint sets A and B such that at least $m/2$ edges connect a vertex in A to a vertex in B, i.e., there is a cut with value at least $m/2$.

Problem 3.4 (Las-Vegas algorithm)**.** A sample involves assigning every vertex in V to one of A or B independently and uniformly at random. What is the expected number of samples before one obtains a cut with value at least $m/2$?

3.3 Derandomization

Problem 3.5 (Derandomization)**.** The goal of this problem is to use Problem 3.3 to explicitly construct a cut of size $m/2$.

Let X be the random variable counting the number of edges in the cut (A, B). Suppose the vertices are ordered arbitrarily as $v_1, \ldots, v_n$. We will assign them one-by-one to either A or B. Give a deterministic algorithm using the conditional expectations $\mathbb{E}[X \mid \text{choices for } v_1, \ldots v_{i-1}]$ in order to construct a cut of size at least $m/2$ in polynomial time.

3.4 Because why not

Problem 3.6 (Improving Problem 3.3)**.** Show that if G contains $2n$ vertices and m edges then there exists a partition giving a cut of size at least $mn/(2n-1)$, and if G contains $2n+1$ vertices and m edges then there exists a partition giving a cut of size at least $m(n+1)/(2n+1)$.

Remark. If you are a fan of this method, see the book of the same name by Alon and Spencer, and for a compilation of problems, see "Unexpected Uses of Probability" by Ravi Boppana.

4 Gestion de mémoires caches : Least Recently Used (LRU)

Consider a cache of size k that can hold up to k pages. We are given a sequence of page requests $\sigma = (p_1, \ldots, p_n)$ where each p_i is a request to a page.

Least Recently Used (LRU) algorithm :
- If a requested page is already in the cache (a cache hit), no eviction occurs.
- If the requested page is not in the cache (a cache miss), the algorithm evicts the page for which the longest time has passed since it was last requested and inserts the requested page into the cache.

Assume that the cost of serving a request is 0 if it is a cache hit and 1 if it is a cache miss. What is the competitive ratio of the LRU caching algorithm ?