TopMath
Mathematik mit Promotion

Approximation Algorithms for the Non-Uniform k-Center Problem

Katja Ettmayr

Abstract

Clustering is a field of combinatorial optimization that concerns itself with grouping data points into different clusters. Usually, one tries to partition the data points into k clusters, where each cluster has a designated center, determining the radius of the cluster as the maximum distance between the center and any contained point. In the **Non-Uniform k-Center (NUkC)** problem, we are given t different radii with corresponding multiplicities and seek to find the minimum factor with which we can scale the radii such that all input points can be covered using the specified number of balls with scaled radius. This problem was introduced in [1], and while it is conjectured that there should be a **constant factor approximation** for t-NUkC if t is a fixed constant, so far, such algorithms could only be developed for $t \leq 4$. We study t-NUkC from different perspectives: we consider the problem in the **Euclidean setting**, providing an algorithm that finds with high probability a solution that uses only few balls more than allowed, which are almost optimally scaled. We also discuss the natural **linear program** modelling t-NUkC, its integrality gap, and two bicriteria approximation algorithms based on it. Then, we focus on the notion of **well-separateness**: In this setting, we assume that for some radius, only a specific subset of input points is admissible for placing the centers of the respective radius, and the inter-point distance in this subset is relatively large.

Formal Definitions

Metric Balls. Given a metric space (X, d), some point $c \in X$, and $r \geq 0$, the *ball with radius r centered in c* is $B(c, r) := \{p \in X \mid d(c, p) \leq r\}$. A ball B *covers* a point $p \in X$ if $p \in B$.

t-NUkC Problem. The input is an instance $I = ((X, d), (r_1, k_1), \ldots, (r_t, k_t))$, where (X, d) is a finite metric space, $r_1 \geq \ldots \geq r_t \geq 0$, and $k_i \in \mathbb{N}$ for all $i \in [t]$. We want to cover all points in X by centering up to k_i balls of radius $\alpha \cdot r_i$ for all $i \in [t]$ in X, where α is called a *dilation factor*. The goal is to find such placement (yielding a *solution*) of minimum dilation factor. (NP-hard)

Approximating t-NUkC. A *β-approximation algorithm* for t-NUkC is a polynomial time algorithm that, given any instance I of t-NUkC, provides a solution of dilation at most $\beta \cdot \alpha^*$, where α^* denotes the minimum dilation of any solution to I.

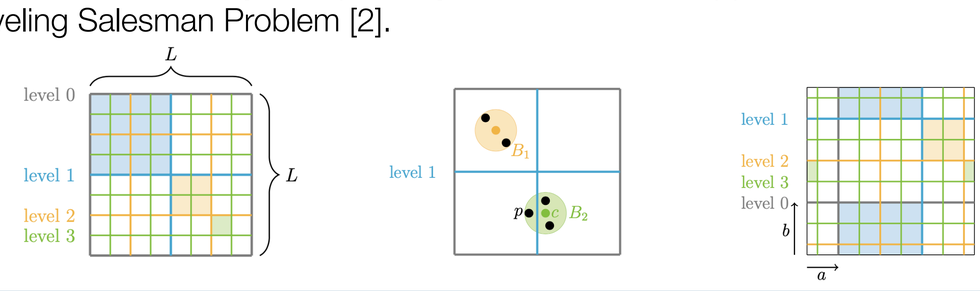
Bicriteria Approximations for t-NUkC. A *(γ, β)-bicriteria approximation algorithm* for t-NUkC is a polynomial time algorithm that provides for each instance of t-NUkC a solution that uses at most γ times too many balls of each radius type and dilates the radii by a factor of at most $\beta \cdot \alpha^*$.

Euclidean Setting

Euclidean t-NUkC. We assume that $X \subseteq \mathbb{Z}^2$ and that d is the Euclidean metric on $\mathbb{R}^2$ restricted to X. In contrast to the general t-NUkC problem, we allow the balls of a solution to be centered somewhere in the place instead of only in X.

Approach inspired by Arora's TSP

We may assume without loss of generality that we work with instances for which a solution of dilation $\alpha = 1$ exists. It remains to "approximately find" this solution. To do this, the plane gets restricted to a square containing all points in X. Then, we proceed in iterations: in each iteration, the positions and radii of the balls from the solution that are "large" in comparison to the length of the current square are "guessed" (at an expense of a factor of $(1 + \epsilon)$ in the dilation, we install a fine grid in the current square and allow balls to be entered only at grid points, making the guessing efficient). After the guessing of the large balls, the square gets dissected into four smaller squares, and the process is repeated – until we obtain unit squares. The goal of the dissection is to create independent smaller squares – in reality, we might need to use more balls than the solution does due to the dissection. To make sure that the amount of balls of radius r_i is only $(1 + O(\epsilon))k_i$ for each i, we work with *randomized* dissections. Using these dissections recursively is inspired by Aurora's famous approach for the Euclidean Traveling Salesman Problem [2].



Working with Linear Programs (LP)

LPs and Approximation. Many problems in combinatorial optimization can be modelled in a natural way by an Integer Program (IP). Considering the corresponding LP relaxation is a common strategy in approximating problems: Often, one can compute a feasible solution to the LP relaxation and round it in a certain way, obtaining a relatively "cheap" solution to the IP.

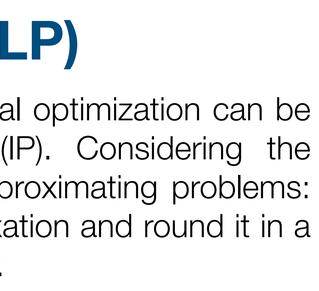
Natural LP relaxation. For each point $p \in X$ and $i \in [t]$, the variable $x_{p,i}$ describes the share by which the ball $B(p, r_i)$ is "open" in a fractional solution

$$\min \alpha \quad \text{s. t.} \sum_{p \in X} x_{p,i} \leq k_i \qquad \forall i \in [t]$$

$$\sum_{i \in [t]} \sum_{q \in B(p, \alpha r_i)} x_{q,i} \geq 1 \qquad \forall p \in X$$

$$x_{p,i} \geq 0 \qquad \forall p \in X, i \in [t]$$

Integrality Gap. The *integrality gap* of an Integer Program (IP) and its corresponding LP relaxation is the maximum ratio between the optimum value to the IP and the optimum value to the LP. Generally, it is a lower bound on the approximation ratio obtainable from rounding LP solutions. Unfortunately, the integrality gap of the natural LP relaxation of t-NUkC is unbounded for $t \geq 3$ as can be shown with the following 3-NUkC instance:



Bicriteria Approximations. Using the LP, there is a very easy $(t,2)$-bicriteria approximation algorithm. Using more intricate strategies, one can as well obtain a bicriteria approximation algorithm opening at most $t + 1$ balls too many, dilating them by a factor of at most 8.

Well-separateness

t-NUkC with restricted center locations. In this variant, for each $i \in [t]$ there is given a set $Y_i \subseteq X$ of potential centers where balls of radius r_i may be placed.

An instance is said to be *well-separated* with respect to some index $i \in [t]$ if the distance between the points in Y_i is sufficiently large; specifically, if $d(u, v) \geq 4 \cdot r_i$ for all $u, v \in Y_i$.

Handling highly well-separated instances. If an instance of t-NUkC is well-separated with respect to all indices (and additionally satisfies a certain condition on the radii), we can find a 2-approximate solution in polynomial time with the use of a dynamic program. Moreover, if the instance is well-separated with respect to all indices but $t - 1$, it is possible to find a 4-approximate solution.

Making instances well-separated. Unfortunately, it is not clear at all how to reduce general t-NUkC instance to instances that are well-separated with respect to all indices. However, the literature yields ideas that can be translated into the language of well-separation, showing how to make an instance well-separated with respect to 1, $t - 1$, and t at the same time.

New Approximation for 3-NUkC. Combining the results on how to make certain layers of an instance well-separated and on how to handle highly well-separated instance, we immediately obtain a new approximation algorithm based on the notion of well-separation. While the existence of an approximation algorithm for 3-NUkC and even 4-NUkC was already known, we hope that the collection of tools concerning well-separateness can be enlarged in the future and used to develop approximation algorithms for $t \geq 5$.

Selected References

[1] Chakrabarty, Goyal, Krishnaswamy. "The Non-Uniform k-Center Problem". In: *ACM Transactions on Algorithms* 16.4 (2020).
[2] S. Arora. "Polynomial time approximation schemes for euclidean TSP and other geometric problems". In: *Proceedings of 37th Conference on Foundations of Computer Science* (1996)