Effectiveness of Local Search for Geometric Optimization

Vincent Cohen-Addad and Claire Mathieu*

Département d'Informatique, UMR CNRS 8548
École Normale Supérieure, Paris, France
{vcohen, cmathieu}@di.ens.fr

Abstract

What is the effectiveness of local search algorithms for geometric problems in the plane? We prove that local search with neighborhoods of magnitude $1/\epsilon^c$ is an approximation scheme for the following problems in the Euclidean plane: TSP with random inputs, Steiner tree with random inputs, uniform facility location (with worst case inputs), and bicriteria k-median (also with worst case inputs). The randomness assumption is necessary for TSP.

1998 ACM Subject Classification I.3.5 Computational Geometry and Object Modeling, F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Local Search, PTAS, Facility Location, k-Median, TSP, Steiner Tree

Digital Object Identifier 10.4230/LIPIcs.SOCG.2015.329

1 Introduction

Local search. Local search techniques are popular heuristics for hard combinatorial optimization problems. Given a feasible solution, the algorithm repeatedly performs operations from the given class, each improving the cost of the current solution, until a solution is reached for which no operation yields an improvement (a locally optimal solution). Alternatively, we can view this as a neighborhood search process, where each solution has an associated neighborhood of adjacent solutions, i.e., those that can be reached with a single operation, and one moves to a better neighbor until none. Such techniques are easy to implement, easy to parallelize, and fast and give good results. One advantageous feature of local search algorithms is their flexibility; they can be applied to arbitrary cost functions, even in the presence of additional constraints. However, there has long been a gap between worst-case guarantees and real-world experience. Thus, it is interesting to analyze such algorithms rigorously and, even in settings where alternative, theoretically optimal polynomial-time algorithms are known.

Problems studied. We focus on Euclidean problems in the plane (the results extend to small dimensions), and study clustering and network connectivity type problems: the traveling salesman problem (TSP), Steiner tree, facility location, and k-median. The *traveling salesman* problem is to connect n input points with a tour of minimum total length. The *Steiner tree* problem, given n terminal points, is to choose additional *Steiner* points so as to minimize the length of the minimum tree spanning terminal and Steiner points. The *facility location* problem, given n client points and a facility opening cost f, chooses how many facilities to open and where to open them to minimize the combination of the cost of opening facilities

* Partially supported by ANR RDAM.

31st International Symposium on Computational Geometry (SoCG'15).
Editors: Lars Arge and János Pach; pp. 329–344

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and of the total distance from each client to the nearest open facility. The *k-median* problem, given n points and an integer k, chooses where to open k facilities so as to minimize the total distance from each client to the nearest open facility.

Algorithms. Our goal is to prove, under minimal assumptions, that local search finds solutions whose cost is within a $(1+\epsilon)$ factor of optimal. For that goal, local search must do a little more: instead of modifying the current solution by swapping a single point, edge or edge pair (depending on the problem) in and out of the solution, our version of local search swaps up to $1/\epsilon^c$ points, edges or edge pairs. This is a standard variation of local search (particularly for the traveling salesman tour), whereby each iteration is slowed down due to an increase in the size of the neighborhood, but the local optimum tends to be reached after fewer iterations and is of higher quality. Moreover, most implementations of local search do not continue iterating all the way to a local optimum, but stop once the gain obtained by each additional iteration is essentially negligible. Our algorithm thus has a stopping condition, when no local exchange could improve the cost by more than a factor of $1-1/n$. Then, the runtime is polynomial, at most $n^{1/\epsilon^{O(1)}}$.

Results. Our results are as follows.

1. For TSP, we assume that the input points are random uniform in $[0,1]^2$. Here local search swaps $O(1/\epsilon^c)$ edges in the tour. Then local search finds a solution with cost $(1+\mathcal{O}(\epsilon))OPT$. The proof is not difficult and serves as a warm-up to the later sections. The random input assumption is necessary : in the worst-case setting, we give an example where a locally optimal solution has cost more than $(2-\epsilon)OPT$.
2. Similarly, for Steiner tree, assuming random uniform input, again local search finds a solution with cost $(1+\epsilon)OPT$.
3. For facility location, we prove the following: consider the version of local search where local moves consist of adding, deleting or swapping $O(1/\epsilon^c)$ facilities. Then, even for worst case inputs, local search finds a solution with cost $(1+\epsilon)OPT$. This is the core result of the paper. We transform the dissection technique from Kolliopoulos and Rao [14] into a tool for analyzing local search.
4. For k-median, our result is similar, except that local search uses $(1+\epsilon)k$ medians instead of k, so that result is bicriteria. This is a technical, variant of the facility location result.

Related work

TSP and Steiner Tree. The TSP problem in the Euclidean plane has a long history, including work with local search [9, 17, 18]. Most relevant is the work of Karp [13] giving a simple construction of a near-optimal tour when points are drawn from a random distribution. That work has been subsumed by the approximation schemes of Arora [1] (and its improvements [2, 23]) and of Mitchell [21], using a hierarchical dissection technique. Arora noted the relation between that technique and local search, observing:

> *Local-exchange algorithms for the TSP work by identifying possible edge exchanges in the current tour that lower the cost [...]. Our dynamic programming algorithm can be restated as a slightly more inefficient backtracking [...]. Thus it resembles k-OPT for $k = O(c)$, except that cost-increasing exchanges have to be allowed in order to undo bad guesses. Maybe it is closer in spirit to more ad-hoc heuristics such as genetic algorithms, which do allow cost-increasing exchanges.*

In fact, even with neighborhoods of size $f(\epsilon)$, even in the Euclidean plane, local search for TSP can get stuck in a local optimum whose value is far from the global optimum. However, in the case of random inputs the intuition is correct. Local search algorithms have been widely studied for TSP, but mostly for either a local neighborhood limited to size of 2 or 3 (the 2-OPT or 3-OPT algorithms), or for the general metric case. Those studies lead to proofs of constant factor approximations, see [6, 11, 20, 18, 25]. In particular, in [6], it is proved (by example) that for Euclidean TSP 2-OPT cannot be a constant-factor approximation in the worst case. For the metric Steiner Tree problem, the best approximation algorithm up to 2010 was a constant factor approximation due to Robins and Zelikovsky and was by local search [24].

Facility Location and k-Median. For clustering problems – facility location and k-median – there has also been much prior work. A proof of NP-Hardness of k-median even in the Euclidean setting is given in [19]. The first theoretical guarantees for local search algorithms for clustering problems are due to Korupolu et al. [15]. They show that the local search algorithm which allows swaps of size p is a constant factor approximation for the metric case of the k-Median and Facility Location problems. However, for k-Median the algorithm requires a constant-factor blowup in the parameter k. By further refining the analysis, Charikar et al. [7] improved the approximation ratio. More recently, Arya et al. showed in [3] that the local search algorithm which allows swaps of size p is a $3 + 2/p$-approximation without any blowup in the number of medians. Nevertheless, no better results were known for the Euclidean case (See the survey paper [26]). Kolliopoulos and Rao define in [14] a recursive "adaptive" dissection of a square enclosing the input points. At each dissection step [1], they cut the longer side of each rectangle produced by the previous step in such a way that each of the two parts has roughly the same surface area. Our analysis uses a new version of their dissection algorithm to analyze the local search algoritm.

Other related work. The question of the efficiency of local search for Euclidean problems was already posed by Mustafa and Ray and Chan and Har-Peled. They proved that local search (with local neighborhood enabling moves of size $\Theta(1/\epsilon)$) gives approximation schemes for hitting circular disks in two dimensions with the fewest points, for several other Euclidean hitting set problems [22], and for independent sets of pseudo-disks [5]. This led to further PTASs by local search for dominating set in disks graph [10] and for terrain guarding [16]. Those papers rely on the combinatorial properties of bipartite planar graphs. Our analysis technique is different since we rely on dissections.

One problem related to facility location is k-means. For k-means, Kanungo, Mount, Netanyahu and Piatko [12] proved that local search gives a constant factor approximation. Much remains to be understood.

We also note that there exists proofs of constant factor approximation by local search for the metric capacitated facility location [8].

Plan. The paper is organized as follows: in the next section, as a warm-up we prove the results on TSP and Steiner tree for random inputs. We then analyze local search for facility location, proposing a new recursive dissection. We suitably extend lemmas from [14]. The meat of that section is the proof of Proposition 4.2, which is our main technical contribution.

[1] There is also a "sub-rectangle" step not described here.

We end with the k-median result, that requires additional ideas to deal with the cardinality constraint.

2 Polynomial-Time Local Search Algorithms

Throughout this paper, we denote by $L \triangle L'$ the symmetric difference of the sets L and L'. We present the local search algorithm that is considered in this paper (see Algorithm 1 below).

Algorithm 1 Local Search (ε)

Input: A set $\mathcal{C}$ of points in the Euclidean plane
$S \leftarrow$ Arbitrary feasible solution (of cost at most $\mathcal{O}(2^n \mathrm{OPT})$).
while $\exists\ S'$ s.t. Condition(S', ε) **and** cost$(S') \leq (1 - 1/n)$ cost(S)
do
    $S \leftarrow S'$
end while
Output: S

Note that the type of S, Condition, $f(\varepsilon)$ and Cost(S) are problem dependent. Namely,

- for Facility Location, S is a set of points, Condition(S', ε) is $|S \triangle S'| = \mathcal{O}(1/\varepsilon^3)$ and $\mathrm{Cost}(S) = |S| + \sum_{c \in \mathcal{C}} \min_{s \in S} d(c, s)$;
- for k-Median, S is a set of points, Condition(S', ε) is $|S \triangle S'| = \mathcal{O}(1/\varepsilon^9)$ and $|S'| \leq (1+3\varepsilon)k$ and $\mathrm{Cost}(S) = \sum_{c \in \mathcal{C}} \min_{s \in S} d(c, s)$;
- for TSP S is a set of edges, Condition(S', ε) is $|S \triangle S'| = \mathcal{O}(1/\varepsilon^2)$ and "S' is a tour and there is no two edges intersecting" (if the initial tour contains intersecting edges we start by modifying the tour so that no two edges intersect) and $\mathrm{Cost}(S) = \sum_{s \in S} \mathrm{length}(s)$;
- for Steiner Tree, S is a set of points, Condition(S', ε) is $|S \triangle S'| = \mathcal{O}(1/\varepsilon^2)$ and $|S'| \leq n$ (if the initial set of Steiner vertices is greater than n, we greedily remove Steiner vertices until the set has size n) and $\mathrm{Cost}(S) = \mathrm{MST}(S \cup \mathcal{C})$, where $\mathrm{MST}(S \cup \mathcal{C})$ is the length of the minimum spanning tree of the points in $S \cup \mathcal{C}$.

We now focus on the guarantees on the execution time of the algorithms presented in this paper. The proof of the following Lemma is deferred to the Appendix.

▶ **Lemma 2.1.** *The number of iterations of Algorithm 1 is polynomial for the Facility Location, k-Median, Traveling Salesman and Steiner Tree Problems.*

▶ Remark. Up to discretizing the plane and replacing $(1 - 1/n)$ by $(1 - \Theta(1/n))$, finding S' takes time $\mathcal{O}(n^{\mathcal{O}(1/\varepsilon^c)}\varepsilon^{-1})$, for some constant c which depends on the algorithm.

3 Euclidean Traveling Salesman Problem and Steiner Tree

▶ **Theorem 3.1.** *Consider a set of points chosen independently and uniformly in* $[0, 1]^2$. *Algorithm 1 produces:*

- *In the case of the Traveling Salesman problem, a tour whose length is at most* $(1 + \mathcal{O}(\varepsilon))T_{OPT}$, *where* T_{OPT} *is the length of the optimal solution.*
- *In the case of the Steiner Tree problem, a tree whose length is at most* $(1 + \mathcal{O}(\varepsilon))T_{OPT}$, *where* T_{OPT} *is the length of the optimal solution.*

To prove Theorem 3.1, we first prove the following result.

▶ **Theorem 3.2.** *Consider an arbitrary set of points in* $[0,1]^2$*. Algorithm 1 produces:*

- *In the case of the Traveling Salesman problem, a tour whose length is at most* $(1+\mathcal{O}(\varepsilon^2))T_{OPT}+\mathcal{O}(\varepsilon\sqrt{n})$*, where* T_{OPT} *is the length of the optimal solution.*
- *In the case of the Steiner Tree problem, a tree whose length is at most* $(1+\mathcal{O}(\varepsilon^2))T_{OPT}+\mathcal{O}(\varepsilon\sqrt{n})$*, where* T_{OPT} *is the length of the optimal solution.*

We model a random distribution of points in a region $\mathcal{P}$ of the plane by a two-dimensional Poisson distribution $\Pi_n(\mathcal{P})$. The distribution $\Pi_n(\mathcal{P})$ is determined by the following assumptions:

1. the numbers of points occurring in two or more disjoint sub-regions are distributed independently of each other;
2. the expected number of points in a region A is $nv(A)$ where $v(A)$ is the area of A; and
3. as $v(A)$ tends to zero, the probability of more than one point occurring in A tends to zero faster than $v(A)$.

From these assumptions it follows that $\Pr[A \text{ contains exactly } m \text{ points}] = e^{-\lambda}\lambda^m/m!$, where $\lambda = nv(A)$. The following result is known.

▶ **Theorem 3.3** ([4])**.** *Let* $\mathcal{P}$ *be a set of* n *points distributed according to a two-dimensional Poisson distribution* $\Pi_n(\mathcal{P})$ *in* $[0,1]^2$ *and let* $T_n(\mathcal{P})$ *be the random variable that denotes the length of the shortest tour through the points in* $\mathcal{P}$*. There exists a positive constant* β *(independent of* $\mathcal{P}$*) such that* $T_n(\mathcal{P})/\sqrt{n} \to \beta$ *with probability 1.*

Assuming Theorems 3.2 and 3.3, we can prove Theorem 3.1.

Proof of Theorem 3.1. We focus on the Traveling Salesman case. Let L be the tour produced by Algorithm 1 and T_{OPT} be the optimal tour. By Theorem 3.3, we have that $\text{Cost}(T_{\text{OPT}}) = \mathcal{O}(\sqrt{n})$ with probability 1. Hence, Theorem 3.2 implies

$$(1-\varepsilon^2)\cdot\text{Cost}(L) \leq \text{Cost}(T_{\text{OPT}}) + \mathcal{O}(\varepsilon\sqrt{n}) = (1+\mathcal{O}(\varepsilon))\cdot\text{Cost}(T_{\text{OPT}}).$$

We now consider the random variable $ST_n(\mathcal{P})$ that denotes the length of the shortest Steiner Tree through the points in $\mathcal{P}$. Since the length of the optimal Steiner Tree is at least half the length of the optimal Traveling Salesman Tour, Theorem 3.3 implies that there exists a constant δ such that $ST_n(\mathcal{P})/\sqrt{n} \geq \delta$ with probability 1. Then, the exact same reasoning applies to prove the Steiner Tree case. ◀

The rest of the section is dedicated to the proof of Theorem 3.2. To this aim, we define a recursive dissection of the unit square according to a set of points $\mathcal{P}$. At each step we cut the longer side of each rectangle produced by the previous step in such a way that each of the two parts contains half the points of $\mathcal{P}$ that lie in the rectangle. The process stops when each rectangle contains $\Theta(1/\varepsilon^2)$ points of $\mathcal{P}$. We now consider the final rectangles and we refer to them as *boxes*. Let $\mathcal{B}$ be the set of boxes.

▶ **Lemma 3.4** ([13])**.** $\sum_{b\in\mathcal{B}} |\partial b| = \mathcal{O}(\varepsilon\sqrt{|\mathcal{P}|})$*, where* $|\partial b|$ *is the perimeter of box* b *and* $|\mathcal{P}|$ *is the number of points in* $\mathcal{P}$*.*

For any set of segments S and box b and for each segment s, let s_b be the part of s that lies inside b. We define $\text{In}(S,b) := \{s_b \mid s \in S \text{ and } s \text{ has at least one endpoint in } b\}$ and $\text{Cross}(S,b) := \{s_b \mid s \in S \text{ and } s \text{ has no endpoint in } b\}$. Moreover we define $\text{Out}(S,b) := \{s_{b'} \mid s \in S \text{ and } b \neq b'\}$. Additionally, let $S(b) = \sum_{s\in S}\text{length}(s_b)$.

We can now prove the two following structural Lemmas. See Fig. 1 for an illustration of the proof.

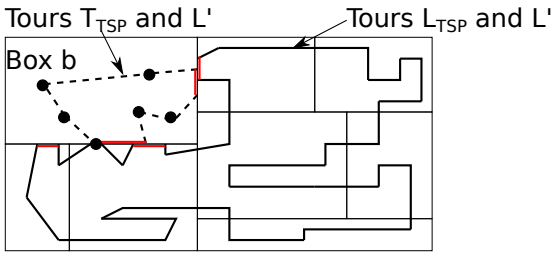

Figure 1 The solid black segments form the tour L_{TSP} outside b. The dotted line segments are the tour T_{TSP} inside b. The red segments are the one needed to connect the two tours.

▶ **Lemma 3.5.** *Let L_{ST} be a locally optimal solution to the Steiner Tree problem and let T_{ST} be any Steiner Tree. Let $\mathcal{B}$ be a set of boxes produced by a dissection of $\mathcal{P} \cup L_{ST} \cup T_{ST}$. Using the same notation for a set of segments and their total length, we then have for any box $b \in \mathcal{B}$*

$$(1 - \mathcal{O}(\varepsilon^2))L_{ST}(b) \leq In(T_{ST}, b) + |\partial b| + L_{ST}/n,$$

where $|\partial b|$ is the perimeter of b.

Proof. For each box b, the segments of $\mathrm{Cross}(L_{\mathrm{ST}}, b)$ can be distributed into 6 different classes according to which side of b they intersect.

We divide further. Since the segments of a class are pairwise disjoint, there is a natural ordering of the segments inside each class. For each class that contains more than $1/\varepsilon^2$ segments, we partition them into subsets that contain $\Theta(1/\varepsilon^2)$ consecutive segments (in the natural order of the class). We define a sub-box for each subset of each class as follows. Let s and s' be the two extreme segments of the set in the ordering of the class. The sides of the sub-box associated to this subset consists of s and s' and the two shortest paths p, p' along the sides of b that connects the endpoints of s and s'.

Remark that the sum of the lengths of the sides of all the sub-boxes is at most $|\partial b| + \mathcal{O}(\varepsilon^2 L_{\mathrm{ST}}(b))$. For each sub-box b_0, let L' be the set of vertices of L_{ST} that are outside b_0, plus the set of vertices of T_{ST} that are inside b_0, plus the set of the intersection points of the edges of L_{ST} and T_{ST} with the sides of b_0. Thus, $L' \leq \mathrm{Out}(L_{\mathrm{ST}}, b_0) + \mathrm{In}(T_{\mathrm{ST}}, b_0) + |\partial b_0|$. Moreover, we have $|L_{\mathrm{ST}} \triangle L'| = \mathcal{O}(1/\varepsilon^2)$ and the local near-optimality argument applies. Namely, we obtain that $(1 - 1/n)L_{\mathrm{ST}} \leq L'$, and so

$$-1/n \cdot L_{\mathrm{ST}} + \mathrm{In}(L_{\mathrm{ST}}, b_0) + \mathrm{Cross}(L_{\mathrm{ST}}, b_0) \leq \mathrm{In}(T_{\mathrm{ST}}, b_0) + |\partial b_0|.$$

We now sum over all sub-boxes of box b and we obtain

$$L_{\mathrm{ST}}(b) = \mathrm{In}(L_{\mathrm{ST}}, b_0) + \mathrm{Cross}(L_{\mathrm{ST}}, b_0) \leq \mathrm{In}(T_{\mathrm{ST}}, b) + |\partial b| + \mathcal{O}(\varepsilon^2 L_{\mathrm{ST}}(b)) + L_{\mathrm{ST}}/n.$$

◀

▶ **Lemma 3.6.** *Let L_{TSP} be a locally optimal solution to the Traveling Salesman problem and let T_{TSP} be any tour. Let $\mathcal{B}$ be a set of boxes produced by a dissection of $\mathcal{P}$. Using the same notation for a set of segments and their total length, we then have for any box $b \in \mathcal{B}$*

$$(1 - \mathcal{O}(\varepsilon^2))L_{TSP}(b) \leq In(T_{TSP}, b) + 3|\partial b|/2 + L_{TSP}/n,$$

where $|\partial b|$ is the perimeter of b.

Proof. We again further divide the boxes into sub-boxes as we did for Lemma 3.5. For each sub-box b_0, we define a tour L' obtained by a traversal of the following Eulerian graph. The graph vertices are $\mathcal{P}$, plus the corners of ∂b_0, plus all points of intersection of L_{TSP} and T_{TSP} with ∂b_0. The edges are the segments of $\text{Out}(L_{\text{TSP}}, b_0)$, plus the segments of $\text{In}(T_{\text{TSP}}, b_0)$, plus ∂b_0 (so that the result is connected), plus a minimum length matching of the odd vertices of ∂b_0 (so that the result is Eulerian). Thus, $L' \leq \text{Out}(L_{\text{TSP}}, b_0) + \text{In}(T_{\text{TSP}}, b_0) + 3|\partial b_0|/2$.

Since the number of edges of L intersecting b_0 is $\mathcal{O}(1/\varepsilon^2)$ and the number of edges in $\text{In}(T_{\text{TSP}}, b_0)$ is $\mathcal{O}(1/\varepsilon^2)$, we have $|L_{\text{TSP}} \triangle L'| = \mathcal{O}(1/\varepsilon^2)$ and the local near-optimality argument applies. Namely, we obtain $(1 - 1/n)L_{\text{TSP}} \leq L'$, and so

$$-1/n \cdot L_{\text{TSP}} + \text{In}(L_{\text{TSP}}, b_0) + \text{Cross}(L_{\text{TSP}}, b_0) \leq \text{In}(T_{\text{TSP}}, b_0) + 3|\partial b_0|/2.$$

We now sum over all sub-boxes of box b and we obtain

$$L_{\text{TSP}}(b) = \text{In}(L_{\text{TSP}}, b) + \text{Cross}(L_{\text{TSP}}, b) \leq \text{In}(T_{\text{TSP}}, b) + 3|\partial b|/2 + \mathcal{O}(\varepsilon^2 L_{\text{TSP}}(b)) + L_{\text{TSP}}/n.$$

◀

We can now prove Theorem 3.2.

Proof of Theorem 3.2. We first consider the Traveling Salesman case. Let L_{TSP} be a tour produced by Algorithm 1 and T_{TSP} be any tour. Lemma 3.6 implies that for any box b, we have

$$(1 - \mathcal{O}(\varepsilon^2))L_{\text{TSP}}(b) \leq \text{In}(T_{\text{TSP}}, b) + 3|\partial b|/2 + L_{\text{TSP}}/n.$$

Since there are $\mathcal{O}(\varepsilon^2 n)$ boxes in total, by summing over all boxes, we obtain

$$-\mathcal{O}(\varepsilon^2 L_{\text{TSP}}) + \sum_{b \in \mathcal{B}} L_{\text{TSP}}(b) = (1 - \mathcal{O}(\varepsilon^2))L_{\text{TSP}} \leq \sum_{b \in \mathcal{B}} (\text{In}(T_{\text{TSP}}, b) + 3|\partial b|/2) \leq T_{\text{TSP}} + \frac{3}{2} \sum_{b \in \mathcal{B}} |\partial b|.$$

By Lemma 3.4, $\sum_{b \in \mathcal{B}} |\partial b| = \mathcal{O}(\varepsilon\sqrt{n})$ and so,

$$(1 - \mathcal{O}(\varepsilon^2)) \cdot L_{\text{TSP}} \leq T_{\text{TSP}} + \mathcal{O}(\varepsilon\sqrt{n}).$$

To prove the Steiner Tree case, it is sufficient to notice that the total number of vertices in $\mathcal{P} \cup L_{\text{ST}} \cup T_{\text{ST}}$ is at most $3n$. It follows that the total number of boxes is $\mathcal{O}(\varepsilon^2 n)$ and by Lemma 3.4, $\sum_{b \in \mathcal{B}} |\partial b| = \mathcal{O}(\varepsilon\sqrt{n})$. We apply a reasoning similar to the one for the TSP case to conclude the proof. ◀

Notice that we do not assume that the points are randomly distributed in the $[0, 1]^2$ for the proofs of Lemmas 3.5 and 3.6 and Theorem 3.2, thus they hold in the worst-case.

▶ Remark. One can ask whether it is possible to prove that the local search for TSP is a PTAS without the random input assumption. However, there exists a set of points such that there is a local optimum whose length is at least $(2 - o(\varepsilon))\text{Cost}(\text{OPT})$.

4 Clustering Problems

We now tackle the analysis of the local search algorithm for some Clustering problems. Recall that L and G denote the local and global optima respectively. In the following, for each facility l of L (resp. G), we denote by $V_L(l)$ (resp. $V_G(l)$) the Voronoi cell of l in the Voronoi diagram induced by L (resp. G). We extend this notation to any subset F of L, namely, $V_L(F)$ denotes the union of the Voronoi cells of the facilities of F induced by L. We define

a recursive randomized decomposition (Algorithm 2) based on L and G (and the Voronoi cells induced by L). This decomposition produces a tree encoded by the function Children(), where each node is associated to a region of the Euclidean plane. In the first step of the dissection, B is the smallest square that contains all the facilities of $L \cup G$. At every recursive call of the procedure for (B_r, L_r, G_r), the algorithm maintains the following invariants:

- B_r is a rectangle of bounded aspect ratio;
- L_r consists of all the facilities of L that are contained in B_r;
- G_r consists of all the facilities of G that are contained in B_r, plus some facilities of G that belong to $V_L(L_r)$.

Algorithm 2 Recursive Adaptive Dissection Algorithm

procedure ADAPTIVE_DISSECTION(B, L, G, V_L)
  if $|L| + |G| \geq 1/2\varepsilon^2$ **then**
    if $|L| > 1/2\varepsilon$ **then**
      Sub-Rectangle Process:
      $B' \leftarrow$ minimal rectangle containing all facilities of L in B
      $b' \leftarrow$ maximum side-length of B'
      $B'_+ \leftarrow$ Rectangle centered on B' and extended by $b'/3$ in all four directions.
      $B'' \leftarrow B'_+ \cap B$
      Cut-Rectangle Process:
      $s'' \leftarrow$maximum side-length of B''
      $\ell \leftarrow$line segment that is orthogonal to the side of length s'' and intersects it in a random position in the middle $s''/3$.
      Cut B'' into two rectangles B_1 and B_2 with ℓ.

      Children(B) $\leftarrow \{B_1, B_2\}$
      $L_1 \leftarrow L \cap B_1$
      $L_2 \leftarrow L \cap B_2$
      $G_1 \leftarrow G \cap \{g \mid g \in V_L(L_1)$ and $g \notin B_2\}$
      $G_2 \leftarrow G \setminus G_1$
      DISSECTION(B_1, L_1, G_1, V_L)
      DISSECTION(B_2, L_2, G_2, V_L)
    else
      Partition Process:
      Children(B) $\leftarrow$ Arbitrary partition of the facilities of $L \cup G$ in parts of size in $[1/2\varepsilon^2, 1/\varepsilon^2]$
    end if
  end if
end procedure

Regions. We now introduce the crucial definition of *regions* of a dissection tree $\mathcal{T}$ of solutions L and G. For any node N of the dissection produced by the Partition Process, we consider that the associated rectangle is the bounding box of the facilities of $L_N \cup G_N$. We assign labels to the nodes of the tree. The label of a leaf B is $|L_B| + |G_B|$. Then we proceed bottom-up, for each node of the tree, the labels of a node is equal to the sum of the labels of its two children. Once a node has a label greater than $1/2\varepsilon^2$, we say that this node is a *region node* of the tree and set its label to 0. We define the regions according to the region

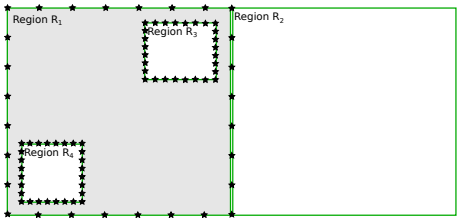

Figure 2 Details of the regions and portals associated to a dissection. The star-shaped points are the portals associated to Region R_1. Regions R_2, R_3, R_4 are the only regions sharing portals with region R_1. All the regions are disjoint.

nodes. For each region node R, the associated region is the rectangle defined by the node minus the regions of its descendants, namely minus the rectangles of nodes of label 0 that are descendants of R. See Fig. 2 for an illustration of the regions. In the following, we denote by $\mathcal{R}$ the set of regions.

Portals. Let $\mathcal{D}$ be a dissection produced by Algorithm 2. For any region R of $\mathcal{D}$ not produced by the Partition Process, we place p equally-spaced *portals* along each boundary of R. We refer to the dissection $\mathcal{D}$ along with the associated portals as $\mathcal{D}_p$. See Fig. 2 for more details on the regions and portals.

Definitions and Notations. For any clustering problem, we denote by $\mathcal{C}$ the sets of the input points. We refer to an input point as a *client*. A solution to a clustering problem is a set of facilities $S \subset \mathbb{R}^2$.

For any solution S and any client c, we denote by c_S the distance from client c to the closest facility of S: $c_S = \min_{s \in S} d(c, s)$. The service cost of a solution S to a clustering problem is $\sum_{c \in \mathcal{C}} c_S$. Additionally, for any solution S and client c, we define $c(S)$ as the facility of S that serves c in solution S, namely $c(S) := \text{argmin}_{s \in S} d(c, s)$

Let B be the smallest rectangle that contains all the clients. Let L and G be two sets of facilities. We now give the definition of an assignment which is crucial for the main proposition.

▶ **Definition 4.1.** We define an *assignment* as a function that maps the clients to the facility of $L \cup G$.

Let E_0 be the assignment that maps each client c to the facility of $\{c(L), c(G)\}$ that is the farther, namely, $\forall c \in \mathcal{C}$, $E_0(c) = \text{argmax}(dist(c, c(G)), dist(c, c(L)))$.

We show the following proposition which is the technical center of the proof.

▶ **Proposition 4.2.** *Let $1/\varepsilon^2 > 0$ be an integer, G and L be two sets of facilities. Let $\mathcal{D}_{1/\varepsilon^2}$ be a dissection tree with portals. There exists an assignment E that satisfies the following properties. Let R be a region not produced by the Partition Process. If a client c is such that $c(L) \in R$ and $c(G) \notin R$ then $E(c)$ is either a portal of R or a facility of $L \setminus R$.*
Moreover,

$$\mathbb{E}[\sum_{c \in \mathcal{C}} |dist(c, E(c)) - dist(c, E_0(c))|] = \sum_{c \in \mathcal{C}} \mathcal{O}(\varepsilon^2 \log(1/\varepsilon^2) \cdot (c_G + c_L)).$$

We start by proving some properties of Algorithm 2[2]. The proofs of the following Lemmas are deferred to the Appendix.

▶ **Definition 4.3** (Aspect Ratio)**.** We define the aspect ratio of a rectangle R that has sides of lengths r and r' as $\max(\frac{r}{r'}, \frac{r'}{r})$.

▶ **Lemma 4.4.** *Let R be a rectangle produced by either the Sub-Rectangle or the Cut-Rectangle process of Algorithm 2. The aspect ratio of R is at most 5.*

▶ **Lemma 4.5** ([14])**.** *Let $l \in L$ be a facility and $v \in \mathbb{R}^2$ be any point. Let d be the distance between v and l. If a cutting line segment s produced by the Sub-Rectangle process during Algorithm 2 separates v and l for the first time, then length$(s) \leq 5d$.*

▶ **Lemma 4.6.** *Let L be a set of facilities. Let $v \in \mathbb{R}^2$, $l \in L$, $d_0 = dist(v, l)$. Suppose that, in Algorithm 2, v and l are first separated by a line s that is vertical and that l is to the right of s. Let d_1 be the distance from v to the closest open facility located to its left. Then, the length of s is either: (i) larger than $d_1/4$ or (ii) smaller than $12d_0$.*

▶ **Lemma 4.7** ([14])**.** *Let Event$_0(d, s)$ denote the event that an edge e of length d is separated by a cutting line of side-length s that is produced by Cut-Rectangle.*
Then, $Pr[Event_0(d, s)] \leq 3d/s$.

We now show the proof of the Structure Theorem.

Proof of Proposition 4.2. Let $p := 1/\varepsilon^2$. By linearity of expectation, we only need to show this on a per-client basis.

Let c be a client and R a region containing $l := c(L)$ but not $g := c(G)$. Let B be the first box of the dissection, in top-down order, that contains l but not g, and let s be the side of B that is crossed by $[l, g]$. We have: $dist(g, l) \leq dist(g, c) + dist(c, l) = c_G + c_L$. Up to a rotation of center g, l is to the north-west of g. Let u, w be the closest facilities of L respectively to the south and to the east of g.

To construct E, we start with $E := E_0$, and modify E one client at a time so that each client satisfies the first property, and we bound the corresponding expected cost increase. The initial cost of E is $\sum_{c \in \mathcal{C}} \max(c_G, c_L)$. We modify $E(c)$ depending on whether s is vertical or horizontal and according to the length of s. We first provide an upper bound on the expected cost increase induced by $E(c)$ for the case where s is vertical. It is easy to see that, when s is horizontal, applying the same reasoning on w instead of u leads to an identical cost increase and thus, the total cost increase is at most twice the cost increase computed for the case where s is vertical.

By Lemma 4.6, the following cases cover all possibilities for the case where s is vertical.

- s is vertical and s was produced by Sub-Rectangle. Then we define $E(c)$ as the portal on s that is closest to $[g, l]$. By Lemma 4.5, the cost increase is at most $\mathcal{O}((c_G + c_L)/p)$.
- s is vertical and s was produced by Cut-Rectangle and its length is at most $12(c_L + c_G)$. Then again we define $E(c)$ as the portal on s that is closest to $[g, l]$. By assumption, again the cost increase is at most $\mathcal{O}((c_G + c_L)/p)$.
- s is vertical and s was produced by Cut-Rectangle and its length is greater that $12(c_L + c_G)$. Lemma 4.6 implies that s has length greater than $d_u/4$. If the length of s is in $[d_u/4, pd_u]$. Then again we define $E(c)$ as the portal on s that is closest to $[g, l]$. Let $\mathcal{E}_0$ be the event

[2] Lemma 4.5 is essentially Lemma 4 from [14] but a careful writing of the details of the calculation reveals slightly different constants.

that $d_u/4 \leq |s| \leq p \cdot d_u$ and s is vertical. The expected cost increase in this case is, by Lemma 4.7, at most

$$\sum_{\substack{d_u/4 \leq i \leq p \cdot d_u \\ \text{s.t } i/d_u \text{ is power of 2}}} pr[|s| = i \text{ and } \mathcal{E}_0] \cdot (i/p) \leq \mathcal{O}(\log(p)/p \cdot (c_G + c_L)).$$

- We now turn to the last case. Namely, s was produced by Cut-Rectangle and its length is greater than or equal to $p \cdot d_u$. We define $E(c)$ depending on whether u is in R or not. This leads to two different sub-cases.

1. $u \notin R$. Then we define $E(c) := u$. The cost is bounded by the cost to go to g ($\max(c_G, c_L)$) plus the cost to go from g to u, which is d_u. Let $\mathcal{E}_1$ be the event that $u \notin R$ and $p \cdot d_u < |s|$ and s is vertical. The cost increase is, by Lemma 4.7, at most,

$$\sum_{\substack{i > p \cdot d_u \\ \text{s.t } i/d_u \text{ is power of 2}}} pr[|s| = i \text{ and } \mathcal{E}_1] \cdot (d_u) \leq \mathcal{O}((c_G + c_L)/p).$$

2. $u \in R$. Let d denotes the first line that separates u from g. Since u is to the right of g, d is different from s and has size at least d_u. We have two sub-cases.
First, if d was produced before s in the dissection, then we also have $|d| > |s|$. Let $\mathcal{E}_2$ be the event $|d| > |s| > p \cdot d_u$ and s is vertical. We now fix d. We assign $E(c)$ to be the closest portal on R, the expected cost increase conditioned upon d is then at most:

$$\sum_{\substack{p \cdot d_u < i \leq |d| \\ \text{s.t } i/d_u \text{ is power of 2}}} pr[|s| = i \text{ and } \mathcal{E}_2] \cdot (i/p) \leq \mathcal{O}(\log(\frac{|d|}{p \cdot d_u}) \cdot (c_G + c_L)/p).$$

We then remove the conditioning on d. If d was produced by the Sub-Rectangle process, then $p \cdot d_u < |d| \leq 5d_u$ by Lemma 4.5 and the expected cost increase is at most $\mathcal{O}((c_G + c_L)/p)$. Otherwise, d was produced by the Cut-Rectangle process, and then the expected cost increase is at most

$$\sum_{\substack{i > p \cdot d_u \\ \text{s.t } i/d_u \text{ is power of 2}}} pr[|d| = i \text{ and } \mathcal{E}_2] \cdot \mathcal{O}(\log(\frac{i}{p \cdot d_u}) \cdot (c_G + c_L)/p) \leq \mathcal{O}((c_G + c_L)/p).$$

Second, if d was produced after s in the dissection, namely $|s| > |d|$. Let $\mathcal{E}_3$ denote the event that $|s| > |d|$ and $|s| > p \cdot d_u$ and s is vertical. We assign c to the closest portal located on d, which is at distance at most $d_u + |d|/p$ from g (and so at distance at most $c_G + d_u + |d|/p$ from c). We start by fixing s. The expected cost conditioned upon s is then (no matter how d was produced), at most

$$\sum_{\substack{d_u < i < |s| \\ \text{s.t } i/d_u \text{ is power of 2}}} pr[|d| = i \text{ and } \mathcal{E}_3] \cdot (d_u + i/p)$$

We then remove the conditioning on s, which leads to an expected cost of at most

$$\sum_{\substack{j > p \cdot d_u \\ \text{s.t } i/d_u \text{ is power of 2}}} pr[|s| = j \text{ and } \mathcal{E}_3] \sum_{d_u < i < j} 3(d_u/i) \cdot (d_u + i/p) \leq \mathcal{O}((c_G + c_L)/p)$$

Thus, the total expected cost increase for E is at most $\mathcal{O}((\log(p)/p) \cdot (c_G + c_L))$.

◀

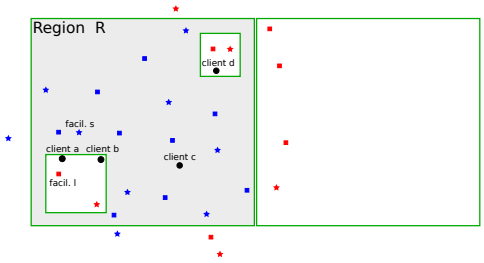

Figure 3 Details of the partitioning of the client. The star-shaped points are the facilities of G and the square-shaped one are the facilities of L. The blue star-shaped and square-shaped belong to respectively G_R and L_R. Since client a is closer to facility l than to facility s, it belongs to the set C_L. Moreover, it is served in L by a facility that does not belong to $V_L(L_R)$, and so, it is not included in set C_R. Client b is closer to facility s than to facility l and so, it is included in set C_R albeit it is served by a facility located on another region in L. Client c is served by a facilities that belongs to $V_L(L_R)$ (in L and G) and so, it belongs to C_R. Finally, client d does not belong to $V_G(G_R)$ and so, is no included in set C_R.

Partitioning the Clients and the Facilities. Before going further, we need to define a partition of the clients and the facilities according to the dissection produced by Algorithm 2.

We partition the clients into two sets C_G and C_L. C_G contains the clients that are closer to a facility of G than to a facility of L and C_L contains the rest of the clients, namely $C_G := \{c \mid c_L = \max(c_L, c_G)\}$ and $C_L := \{c \mid c_G \neq \min(c_L, c_G)\}$. Let $\mathcal{D}$ be a dissection produced by Algorithm 2 and the set of its associated regions $\mathcal{R}$. For any region R, we denote $C_G(R)$ the set of clients that are served by G_R in G and that do not lay on a region not in P. Furthermore, we define $C_L(R)$ as the set of clients that are served by L_R in L and let $C_R := V_G(G_R) \setminus (C_L \cap (V_L(L \setminus L_R))$ [3]. This set contains the clients served by G_R in G except those that belong to C_L and that are served by $L \setminus L_R$ in L. See Fig. 3 for an illustration. Additionally, we define $\Delta_R := V_L(L_R) \setminus V_G(G_R)$.

4.1 Facility Location

We now prove the approximation ratio of Algorithm 1 for facility location.

▶ **Theorem 4.8.** *For Facility Location, Algorithm 1 produces a solution L of cost at most $(1 + \mathcal{O}(\varepsilon)) \cdot Cost(OPT)$.*

Proof. Let OPT be a globally optimum solution and L be a locally optimum solution. By Proposition 4.2, for any $p > 0$ there exists an assignment E for each random dissection $\mathcal{D}_p$ with portals of $L \cup \mathrm{OPT}$, such that for any client c and region R, if $c(L) \in R$ and $c(G) \notin R$ then c is served by a portal of R or a facility of $L \setminus R$ in E and the expected cost of E is at most $\mathbb{E} = \sum_{c \in \mathcal{C}} \max(c_L, c_G) + \mathcal{O}(\log(p)/p \cdot (\sum_{c \in \mathcal{C}} (c_G + c_L)))$. This implies that there exists a dissection $\mathcal{D}_p$ for which E has value at most $\mathbb{E}$.

Throughout the proof, we consider this dissection $\mathcal{D}_p$ and fix $\varepsilon := \log(p)/p$. Let $\mathcal{R}$ be the set of regions associated to $\mathcal{D}_p$. We start by constructing a solution G based on OPT and we compare the cost of L to the cost of G. The solution G contains all the facilities of OPT plus some extra facilities. First, it has one facility at each portal of $\mathcal{D}_p$. Moreover, for each region R that is produced by the Partition Process, we open the facilities of L_R. Recall that for

[3] This can be rewritten as $C_R := V_G(G_R) \cap (C_G \cup V_L(L_R))$.

each of these regions, $|L_R| \leq 1/\varepsilon$. We keep the same assignment for the clients. Since there are $\mathcal{O}(\varepsilon^2(|G|+|L|))$ regions and that for each region G uses at most $1/\varepsilon$ extra facilities, the cost of G is at most $\text{Cost}(\text{OPT})+ \mathcal{O}(\varepsilon(|\text{OPT}|+|L|)f)$. We now prove that the cost of L is at most $(1+\mathcal{O}(\varepsilon))/(1-\mathcal{O}(\varepsilon))$ times the cost of G, namely

$$|L|\cdot f + \sum_{c\in\mathcal{C}} c_L \leq (\frac{1+\mathcal{O}(\varepsilon)}{1-\mathcal{O}(\varepsilon)})(|G|\cdot f + \sum_{c\in\mathcal{C}} c_G).$$

We focus on the cost of a region R. We show that, by local optimality, for each region R, replacing solution L by solution G does not lead to a much better cost. We serve the clients of C_R optimally (namely by the facilities that serve them in G) and the clients of $L_R \setminus G_R$ by the facilities located on the portals of R or by the facilities of $L \setminus L_R$, depending on whether they belong to C_L or C_G and according to the assignment E. Since $|L_R \setminus G_R| + |G_R \setminus L_R| = \mathcal{O}(\varepsilon^{-3})$, the locality argument applies. Namely, we have

$$(|G_R|-|L_R|)f + \sum_{c\notin C_R\cup\Delta_R} c_L + \sum_{c\in C_R} c_G + \sum_{c\in\Delta_R} c_E \geq (1-1/n)(|L|f + \sum_c c_L).$$

The rest of the proof is mainly computational and can be found in the appendix. ◀

4.2 k-Median

Let L and OPT be respectively local and global optimal solutions to the k-Median problem. We start with a technical Lemma which allows us to find "clusters" of regions of the plane that have roughly the same number of facilities of L and G. See Fig. 4 for an illustration. The proof of the Lemma is deferred to the Appendix.

▶ **Lemma 4.9** (Balanced Clustering). *Let $\mathcal{R} = \{r_1, ..., r_p\}$ be a collection of disjoint sets. Each set contains elements of type either L or G and has size at least $1/2\varepsilon^2$ and at most $1/\varepsilon^2$. The total number of elements of type L is $(1+3\varepsilon)$ times higher than the number of elements of type G.*

There exists a clustering of $\{r_1, ..., r_p\}$ in clusters satisfying the following two properties. For any cluster C,

- *C contains at most $\mathcal{O}(1/\varepsilon^5)$ elements of $\mathcal{R}$, namely $|C| = \mathcal{O}(1/\varepsilon^5)$;*
- *the difference between the number of elements of L in the sets contained in C and the number of elements of G in the sets contained in C is at least $|C|/\varepsilon$:*

$$\sum_{r_i\in C} |r_i \cap L| - \sum_{r_i\in C} |r_i \cap G| \geq |C|/\varepsilon,$$

for any $1/\varepsilon \in \mathbb{N}$.

▶ **Theorem 4.10.** *For k-Median, Algorithm 1 for k-Median produces a solution L of cost at most $(1+\mathcal{O}(\varepsilon)Cost(OPT)$ using at most $1+\mathcal{O}(\varepsilon))k$ Medians.*

Proof. Remark first that solution L uses $(1+\mathcal{O}(\varepsilon))k$ facilities. We now show that the cost of solution L is at most $1+\mathcal{O}(\varepsilon)$ times higher than the cost of the optimal solution.
Recall that by Proposition 4.2, for any $p>0$ there exists an assignment E for each random dissection $\mathcal{D}_p$ of $L \cup \text{OPT}$ with portals, such that for any client c and region R, if $c(L) \in R$ and $c(\text{OPT}) \notin R$ then c is served by a portal of R or a facility of $L \setminus R$ in E and the expected cost of E is at most $\mathbb{E} = \sum_{c\in\mathcal{C}} \max(c_L, c_{\text{OPT}}) + \mathcal{O}(\log(p)/p \cdot (\sum_{c\in\mathcal{C}} (c_{\text{OPT}} + c_L)))$.

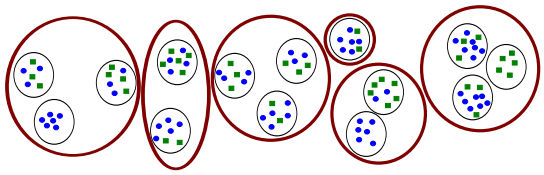

Figure 4 The circle-shaped points are the elements of type L and the square-shaped ones the elements of type G. The black circles mark the sets $\{r_1, \ldots, r_p\}$ and the red ones show a clustering of those sets that satisfy the property of Lemma 4.9.

This implies that there exists a dissection $\mathcal{D}_p$ for which E has value at most $\mathbb{E}$. Throughout the proof, we consider such a dissection $\mathcal{D}_p$ and fix $\varepsilon := \log(p)/p$. Let $\mathcal{R}$ be the set of regions associated to $\mathcal{D}_p$. We prove that the cost of L is at most $(1 + \mathcal{O}(\varepsilon))/(1 - \mathcal{O}(\varepsilon))$ times the cost of S, namely

$$\sum_{c \in \mathcal{C}} c_L \leq \frac{1 + \mathcal{O}(\varepsilon)}{1 - \mathcal{O}(\varepsilon)} \sum_{c \in \mathcal{C}} c_{\mathrm{OPT}}.$$

Let $\mathcal{P}$ be a clustering of the regions satisfying the properties of Lemma 4.9 (depending on L and OPT). We start by constructing a solution G based on OPT and we compare the cost of L to the cost of G. We construct G in a similar way to in the proof of Theorem 4.8. Namely, the solution G contains all the facilities of OPT plus some extra facilities: one facility at each portal of $\mathcal{D}_p$ and for each region R that is produced by the Partition Process, we open the facilities of L_R. Recall that for each of these regions, $|L_R| \leq 1/\varepsilon$. We keep the same assignment for the clients. We now compare the costs of L and G. To do so, we consider all the regions of each cluster of the clustering $\mathcal{P}$ at the same time. Namely for each cluster R, L uses at least as many facilities as G. Therefore $|S_P \setminus L| + |L \setminus S_P| = \mathcal{O}(1/\varepsilon^9)$ and the locality argument applies. The rest of the proof is similar to the proof of 4.8 and is mainly computational and can be found in the Appendix. ◀

Higher Dimensions. Previous results generalize to any dimension d. It leads to algorithms that have exponential dependency in d. More precisely, for any dimension d, more portals are needed to maintain the expected cost increase for the assignment E provided by the Structure Theorem. Each of the $2d$ faces of each region has to count p^{d-1} portals. Proposition 4.2 generalizes to any dimension d with $\mathcal{O}(dp^{d-1})$ portals instead of p. For Facility Location, Condition(S', ε) has to be adapted to $|S \setminus S| + |S \setminus S| = \mathcal{O}(d/\varepsilon^{d+1})$. Thus, Theorem 4.8 still applies to show that the adapted Algorithm provides a $(1 + \mathcal{O}(\varepsilon))$ approximation. For the k-Median problem, Condition(S', ε) has to be adapted to $|S'| \leq (1 + 3\varepsilon)k$ and $|S \setminus S'| + |S' \setminus S| = \mathcal{O}(d/\varepsilon^{7+d})$. Theorem 4.10 still applies to prove the approximation ratio of the adapted Algorithm.

References

1 S. Arora. Polynomial time approximation schemes for euclidean TSP and other geometric problems. In *Symp. on Foundations of Computer Science, FOCS'96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 2–11, 1996.

2 S. Arora. Nearly linear time approximation schemes for euclidean TSP and other geometric problems. In *Symp. on Foundations of Computer Science, FOCS'97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 554–563, 1997.

3 V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.

4 J. Beardwood, J. H. Halton, and J. M. Hammersley. The shortest path through many points. *Mathematical Proc. of the Cambridge Philosophical Society*, 55:299–327, 1959.

5 T. M. Chan and S. Har-Peled. Approximation algorithms for maximum independent set of pseudo-disks. In *Proc. of the Symp. on Computational Geometry*, SCG'09, pages 333–340. ACM, 2009.

6 B. Chandra, H. J. Karloff, and C. A. Tovey. New results on the old k-opt algorithm for the TSP. In *Proc. of the ACM-SIAM Symp. on Discrete Algorithms. 23-25 January 1994, Arlington, Virginia.*, pages 150–159, 1994.

7 M. Charikar and S. Guha. Improved combinatorial algorithms for facility location problems. *SIAM J. Comput.*, 34(4):803–824, 2005.

8 F. A. Chudak and D. P. Williamson. Improved approximation algorithms for capacitated facility location problems. *Math. Program.*, 102(2):207–222, March 2005.

9 G. A. Croes. A method for solving traveling salesman problems. *Operations Research*, 6(6):791–812, 1958.

10 M. Gibson and I. A. Pirwani. Algorithms for dominating set in disk graphs: Breaking the log$n$ barrier – (extended abstract). In *Algorithms – ESA 2010, European Symp., Liverpool, UK, September 6-8, 2010. Proc., Part I*, pages 243–254, 2010.

11 D. S Johnson and L. A McGeoch. The traveling salesman problem : A case study in local optimization. *Local Search in Combinatorial Optimization*, 1:215–310, 1997.

12 T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. A local search approximation algorithm for k-means clustering. *Comput. Geom.*, 28(2-3):89–112, 2004.

13 R. M. Karp. Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane. *Mathematics of Operations Research*, 2(3):209–224, 1977.

14 S. G. Kolliopoulos and S. Rao. A nearly linear-time approximation scheme for the euclidean k-median problem. *SIAM J. Comput.*, 37(3):757–782, 2007.

15 M. R. Korupolu, C. G. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. *J. Algorithms*, 37(1):146–188, 2000.

16 E. Krohn, M. Gibson, G. Kanade, and K. R. Varadarajan. Guarding terrains via local search. *JoCG*, 5(1):168–178, 2014.

17 S. Lin. Computer solutions of the traveling salesman problem. *Bell System Technical Journal, The*, 44(10):2245–2269, 1965.

18 S. Lin and B. W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516, 1973.

19 N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM J. Comput.*, 13(1):182–196, 1984.

20 O. Mersmann, B. Bischl, J. Bossek, H. Trautmann, M. Wagner, and F. Neumann. Local search and the traveling salesman problem: A feature-based characterization of problem hardness. In *Learning and Intelligent Optimization – 6th International Conference, LION 6, Paris, France, January 16-20, 2012, Revised Selected Papers*, pages 115–129, 2012.

21 J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric tsp, k-mst, and related problems. *SIAM J. Comput.*, 28(4):1298–1309, 1999.

22 N. H. Mustafa and S. Ray. PTAS for geometric hitting set problems via local search. In *Proc. of the ACM Symp. on Computational Geometry, Aarhus, Denmark*, pages 17–22, 2009.

23 S. Rao and W. D. Smith. Approximating geometrical graphs via "spanners" and "banyans". In *Proc. of the ACM Symp. on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 540–550, 1998.

24 G. Robins and A. Zelikovsky. Improved steiner tree approximation in graphs. In *Proc. of the ACM-SIAM Symp. on Discrete Algorithms*, SODA, pages 770–779. SIAM, 2000.

25 D. J. Rosenkrantz, R. Edwin Stearns, and P. M. Lewis II. An analysis of several heuristics for the traveling salesman problem. *SIAM J. Comput.*, 6(3):563–581, 1977.

26 J. Vygen. Approximation algorithms for facility location problems. Technical Report 05950, Research Institute for Discrete Mathematics, University of Bonn, 2005.