The Unreasonable Success of Local Search: Geometric Optimization

Vincent Cohen-Addad and Claire Mathieu

Abstract

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

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

1 Introduction

Local search. Local search techniques are very popular heuristics for hard combinatorial optimization problems. Given a feasible solution, the algorithm then repeatedly performs operations from the given class, so long as each improves 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 in a single move, and one continually moves to a better neighbor until no better neighbors exist. Such techniques are easy to implement, easy to parallelize, and in practice they are fast and give good results. One particularly advantageous feature of local search algorithms is their flexibility; they can be applied to arbitrary cost functions, even in the presence of complicated additional constraints. For all those reasons, it is interesting to analyze such algorithms rigorously and prove bounds on their performance, even in settings where alternative, theoretically optimal polynomial-time algorithms are known.

Problems studied. We focus on Euclidian problems in the plane (the results extend to small dimension), and study clustering and network connectivity type problems. In that setting, we contribute to the understanding of the success of local search algorithms for 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 add 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 $f$, must choose how many facilities to open and where to open them to minimize the combination of cost of opening facilities and total distance from each client to the nearest open facility. The K-median problem, given $n$ points and an integer $k$, must choose 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 near-optimal solutions, namely, solutions whose cost is within a $(1 + \epsilon)$ factor of optimal. For that goal, local search must do a little more than just study the immediate 1-exchange local neighborhood: 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 local search swaps up to $1/\epsilon c$ points, edges or edge pairs. This is a variation of local search that is standard (particularly for the traveling salesman tour), whereby each iteration is slowed down by the increase in size of the neighborhood, but the eventual 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: we declare ourselves satisfied with the current solution when no local exchange could improve it by more than a factor of \(1 - 1/n\). Thanks to that stopping condition, the runtime is polynomial, at most \(n^{1/\epsilon^2(n)}\).

**Results.** Our results are as follows.

1. For TSP, we assume that the input points are random uniform. 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 warmup 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. In the analysis, we define a new hierarchical dissection, related to the one designed by Kolliopoulos and Rao [12].

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, slightly more difficult variant of the facility location result.

**Related work.** The TSP problem in the Euclidean plane has a long and distinguished history, including work with local search [8][14][15]. Most relevant for our analysis is the work of Karp [11] giving a simple construction for a near-optimal tour when points are drawn from a random distribution. That work has long been subsumed by the approximation schemes of Arora [1] (and its improvements [2][21]) and of Mitchell [19], and the hierarchical dissection technique. Arora noted the relation between his 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 \(k\)-\textit{OPT} for \(k = O(\epsilon),\) 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.*

It is a somewhat guarded note. Indeed, even with neighborhoods of size \(f(\epsilon)\), even in the Euclidean plane, local search for TSP can get stuck in local optimal whose value is far from the global optimum (See Figure 1 and Lemma 3.7). However, in the case of random inputs the intuition is correct and the argument is not hard.

Local search algorithms have been widely studied for TSP, but mostly for either a local neighborhood limited to 2-OPT or 3-OPT, or for the general metric case. Those studies lead to proofs of constant factor approximations. See [6][10][13][15][22]. In particular, in [15], it is proved (by example) that for Euclidean TSP 2-OPT cannot be a constant-factor approximation in the worst case unless the input is drawn uniformly at random.

For clustering problems — facility location and \(k\)-median — there has also been much prior work. A proof of hardness of \(k\)-median even in the Euclidean setting is given in [16]. The first theoretical guarantees for local search algorithms for clustering problems are due to [13]. 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 refining further the analysis, Charikar et al. improved the approximation ratio and reduced the blowup in [7]. More recently, Arya et al. showed in [4] that the local search algorithm which allows swaps of size $p$ is a $3 + 2/p$-approximation without any blowup in the number of median. Nevertheless, nothing no better results were known for the euclidian case of these problems. See the survey paper [23] for more details. The one whose ideas we use in depth to leverage their power is the approximation scheme by Kolliopoulos and Rao [12].

The question of the efficiency of local search for Euclidian problems was already posed in a very similar spirit by Mustafa and Ray. They proved that local search (with local neighborhood enabling moves of size $\Theta(1/\epsilon^3)$) gives approximation schemes for hitting circular disks in two dimensions with the fewest points, and for several other Euclidian hitting set problems [20].

One tantalizing problem related to facility location is $k$-means. For $k$-means, Kanungo, Mount, Netanyahu and Piatko [10] proved that local search gives a constant factor approximation. Much remains to be understood for that problem.

The paper is organized as follows: in the next section, as a warmup we prove the results on TSP and Steiner tree for random inputs. We then proceed to analyze local search for facility location, proposing a new recursive dissection whose definition depends on the local search algorithm output $L$ and the unknown global optimum $G$. We suitably extend lemmas from [12]. The meat of that section is the proof of Theorem 4.1. That is arguably the main technical contribution of this paper. We end with the proof of our $k$-median result, that requires additional ideas to deal with the cardinality constraint.

## 2 Polynomial-Time Local Search Algorithms

We first present the local search algorithm that is considered in this paper.

**Algorithm 1 Local Search ($\epsilon$)**

1. **Input**: A set $C$ of points in the Euclidian plane
2. $S \leftarrow$ Arbitrary minimal feasible solution
3. while $\exists S'$ s.t. Condition($S', \epsilon$) and cost($S'$) $\leq (1 - 1/n)$ cost($S$)
   4. do
   5. $S \leftarrow S'$
   6. end while
7. **Output**: $S$

Note that the type of $S$, $f(\epsilon)$ and $\text{Cost}(S)$ are problem dependent. Namely,

- for Facility Location, $S$ is a set of points, Condition($S', \epsilon$) is $|S' \setminus S'| + |S' \setminus S| = O(1/\epsilon^3)$ and $\text{Cost}(S) = |S| + \sum_{e \in C} \min_{s \in S} d(e, s)$;
- for K-Median, $S$ is a set of points, Condition($S', \epsilon$) is $|S'| \leq (1+3\epsilon)k$ and $|S' \setminus S| + |S' \setminus S| = O(1/\epsilon^3)$ and $\text{Cost}(S) = \sum_{e \in C} \min_{s \in S} d(e, s)$;
- for the Traveling Salesman Problem $S$ is a set of edges, Condition($S', \epsilon$) is “$S'$ is a tour” and $|S' \setminus S'| + |S' \setminus S| = O(1/\epsilon^2)$ and $\text{Cost}(S) = \sum_{s \in S} \text{length}(s)$;
- for the Steiner Tree Problem, $S$ is a set of points, Condition($S', \epsilon$) $|S' \setminus S'| + |S' \setminus S| = O(1/\epsilon^3)$ and $\text{Cost}(S) = \text{MST}(S \cup C)$, where $\text{MST}(S \cup C)$ is the length of the Minimum Spanning Tree of the points in $S \cup C$. 
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 Appendix B.

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

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

3 Euclidean Traveling Salesman Problem and Steiner Tree

This section is dedicated to the analysis of the two following algorithms. Throughout this section we assume that the input points lie in the unit square. Let \(P\) be the set of input points.

We show the two following Theorems.

Theorem 3.1. Algorithm 1 produces a \((1+O(\varepsilon))\)-approximation to the Euclidean Traveling Salesman problem when the input points are randomly distributed in the plane.

Theorem 3.2. Algorithm 1 produces a \((1+O(\varepsilon))\)-approximation to the Euclidean Steiner Tree problem when the input points are randomly distributed in the plane.

We define the following dissection of a bounding box \(B\) in the plane according to a set of point \(C\).

Namely, the dissection recursively divides the plane. At each step we cut the largest side of each square produced by the previous step in such a way that each of the two parts contains half the input points of the square. The process stop when each square contains \(\Theta(1/\varepsilon^2)\) input points. We now consider the final squares and we refer to them as the boxes. Let \(B\) be the set of the boxes.

Lemma 3.3. \(\sum_{b \in B} |b| = O(\varepsilon \sqrt{n})\), where \(|b|\) is the perimeter of box \(b\).

Let \(T\) be a set of segments and \(b\) be a box, we define \(T(b)\) to be the sum of the lengths of the parts of the segments of \(T\) inside \(b\).

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

Lemma 3.4. Let \(L_{TSP}\) be a locally optimal solution to the Traveling Salesman Problem and \(T_{TSP}\) be any tour. Let \(L_{ST}\) be a locally optimal solution to the Steiner Tree Problem \(T_{ST}\) be any Steiner Tree. Let \(b\) be a box and \(|b|\) be the perimeter of \(b\). Then,
-1/n \cdot L_{\text{TSP}}(b) + L_b \leq T_{\text{TSP}}(b) + 2|b|;
-1/n \cdot L_{\text{ST}}(b) + L_b \leq T_{\text{ST}}(b) + |b|.

**Proof.** Remark first that, since \(b\) contains \(\Theta(1/\varepsilon^2)\) points, both \(L_{\text{TSP}}\) and \(L_{\text{ST}}\) have at most \(\Theta(1/\varepsilon^2)\) edges inside \(b\). We first consider the tour \(L'\), defined as follows: outside \(b\), \(L' := L_{\text{TSP}}\) and inside \(b\), \(L' := T_{\text{TSP}}\). We now show that the sum of the lengths of the segments we need to add to make \(L'\) a tour is at most \(2|b|\). We connect the segments outside \(b\) by adding at most the perimeter of \(b\) and proceed identically for the segments inside \(b\). \(L'\) consists now of two tours that visit all the input points and that are connected. By local optimality, we have \((1-1/n)L_{\text{TSP}} \leq L'\), and so \(-1/n \cdot L_{\text{TSP}} + L_{\text{TSP}}(b) \leq T_{\text{TSP}}(b) + 2|b|\). The same reasoning applies to the case of the Steiner Tree Problem, except that we only need to pick each border of the box once to maintain the tree structure. 

We can now state the following Theorem.

**Theorem 3.5.** Algorithm 1 produces:

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

**Proof.** We first consider the Traveling Salesman case. Let \(L\) be a tour produced by algorithm 1 and \(T\) be any tour. Lemma 3.4 implies that for any box \(b\), we have \(-1/n \cdot L + L(b) \leq T(b) + 2|b|\). Since there are \(\varepsilon^2 \cdot n\) boxes in total, by summing over all boxes, we obtain:

\[-\varepsilon^2 \cdot L + \sum_{b \in B} L(b) \leq \sum_{b \in B} (T(b) + 2|b|)\]

and so,

\[(1 - \varepsilon^2) \cdot L \leq T + \mathcal{O}(\varepsilon \sqrt{n})\]

The exact same reasoning applies for the Steiner Tree case.
Corollary 1. There exists a positive constant $\delta$ (independent of $X$) such that $ST_n(X)/\sqrt{nv(X)} \rightarrow \delta$ with probability 1.

We now prove the two Theorems stated at the beginning of the section.

**Proof of Theorem 3.5** We first focus on the Traveling Salesman case. Let $L$ be the tour produced by algorithm 1 and $T_{OPT}$ be the optimal tour. By Theorem 3.6, we have that $\text{Cost}(T_{OPT}) = O(\sqrt{n})$ with probability 1. Hence, Theorem 3.1 implies

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

The exact same reasoning applies to the Steiner Tree case.

Finally, we show that Algorithm 1 can produce a tour whose length is as bad as twice the optimal tour if the input points can be arbitrarily placed in the plane (See Fig. 5 for more details).

**Lemma 3.7.** Algorithm 1 can produce a tour whose length is as bad as $2 + O(\varepsilon)$ times the optimal tour.

**4 Clustering Problems**

**4.1 Definitions and Notations**

For any clustering problem, we denote by $C$ 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 $\sum_{c \in C} c_S$. Additionally, for any solution $S$ and client $c$, we define $c(S)$ as the facility of $S$ that serves $c$ in solution $S$, namely $c(S) := \arg\min_{s \in S} d(c,s)$.

Let $B$ be the smallest rectangle that contains all the clients. Let $L$ and $G$ be two sets of facilities. Moreover, 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 definition to any subset $F$ of $L$, namely $V_L(F)$ is the union of the Voronoi cells of the facilities of $F$ induced by $L$.

**4.2 Structure Theorem**

We define a recursive randomized decomposition (Procedure 2) according to $L \cup G$. This decomposition produces a tree encoded by the function Children().
There exists an assignment cility of section tree with portals. Let partition Process, if assign labels to the nodes of the tree. The label of a leaf consider that the associated rectangle is the bounding box of facilities of L solutions

Algorithm 2 Recursive Adaptative Dissection Algorithm

```plaintext
1: procedure ADAPTATIVE_DISSECTION(B, L, G)
2:   if |L| + |G| ≥ 1/2ε² then
3:     if |L| ≤ 1/2ε then
4:       Partition Process:
5:       Children(B) ← Arbitrary partition of the facilities of L ∪ G in parts of size in [1/2ε², 1/ε²]
6:     else
7:       Sub-Rectangle Process:
8:       B′ ← minimal rectangle containing all facilities of L in B
9:       s ← maximum sidelength of B′
10:      B′′ ← Rectangle centered on B′ and with sidelength 5s/3
11:     B′′ ← B′′ ∩ B
12:     Cut-Rectangle Process:
13:     s′ ← maximum sidelength of B′′
14:     Cut B′′ into two rectangles B₁ and B₂ with a line segment that is orthogonal to the side of length s′ and intersects it in a random position in the middle s′/3.
15:    Children(B) ← {B₁, B₂}
16:    G_B₁ ← (G \ B₂) ∩ (B ∪ V_L(L ∩ B₁))
17:    G_B₂ ← G \ G_{B₁}
18:   DISSECTION(B₁, L ∩ B₁, G_{B₁})
19:   DISSECTION(B₂, L ∩ B₂, G_{B₂})
20:  end if
21: end if
22: end procedure
```

Regions. We now introduce the crucial definition of regions of a dissection tree 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 ∪ 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 1/2ε², 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 nodes. For each region node R, the associated region is the rectangle defined by the node minus the regions of its descendents, namely minus the rectangle of node of label 0 that is a descendant of R. See Fig. 3 for an illustration of the regions. In the following, we denote by R the set of regions.

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

We show the following structure Theorem of crucial importance.

Theorem 4.1. Let p > 0 be any integer, G and L be two sets of facilities. Let D_p be a dissection tree with portals. Let E_0 be the assignment such that each client c is assigned to the facility of {c(G), c(L)} that is the farthest, namely E_0(c) = \text{argmax}(\text{dist}(c, c(G)), \text{dist}(c, c(L))). There exists an assignment E such for any client c and region R not produced by the Partition Process, if c(L) ∈ R and c(G) ∉ R then c is served by a portal of R or a facility of
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.

\[ L \setminus R \text{ in } E; \text{ and such that the expected difference between the cost of } E \text{ and the cost of } E_0 \text{ is at most} \]

\[ E(\sum_{c \in C} |\text{dist}(c, E(c)) - \text{dist}(c, E_0(c))|) = \sum_{c \in C} \mathcal{O}(\log(p)/p \cdot (c_G + c_L)). \]

We start by proving some properties of Procedure 2. The proofs of the following Lemmas are deferred to Appendix C.

- **Lemma 4.2.** 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 procedure 2 separates \( v \) and \( l \) for the first time, then length \( (s) \leq 5d \).

- **Lemma 4.3.** Let \( L \) be a set of facilities. Let \( v \in \mathbb{R}^2, l \in L, d_0 = \text{dist}(p, l) \). Suppose that \( v \) and \( l \) are separated first by a vertical line \( s \) in the procedure 2 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 \( d_1/2 \) or (ii) smaller than \( 8d_0 \).

- **Lemma 4.4.** Let \( E_0(d, s) \) denote the event that an edge \( e \) of length \( d \) is separated by a cutting line of sidelength \( s \) that is produced by Cut-Rectangle. Then, \( \text{Pr}[E_0(d, s)] \leq 3d/s \).

We now show the proof of the Structure Theorem.

**Proof of Theorem 4.1.** 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: \( \text{dist}(g, l) \leq \text{dist}(g, c) + \text{dist}(c, l) = c_G + c_L \). 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 \( \sum_{c \in C} \max(c_G, c_L) \). 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 4.3, the following cases cover all possibilities for the case where \( s \) is vertical.
\begin{itemize}
  \item $s$ 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 4.2, the cost increase is at most $O((c_G + c_L)/p)$.
  \item $s$ is vertical and $s$ was produced by Cut-Rectangle and its length is at most $8(c_L + c_G)$. 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 $O((c_G + c_L)/p)$.
  \item $s$ is vertical and $s$ was produced by Cut-Rectangle and its length is greater than $8(c_L + c_G)$. Lemma 4.3 implies that $s$ has length greater than $d_u/2$. If the length of $s$ is in $[d_u/2, pd_u]$, then (no matter how $p \cdot d_u$ was produced before $s$ or $s$ was produced after $s$ in the dissection), namely $s$ is in $[d_u/2, pd_u]$, and $s$ is vertical. We now assign $E(c)$ to be the closest portal on $R$, the expected cost increase condition on $d$ is then at most:
  \[
  \sum_{\substack{i > p \cdot d_u \\
    s.t. \ i / d_u \ is \ power \ of \ 2}} p[r | s = i \ and \ E_2] \cdot (d_u) \leq O((c_G + c_L)/p).
  \]
  \end{itemize}

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 4.2 and the expected cost increase is at most $O((c_G + c_L)/p)$. Otherwise, $d$ was produced by the Cut-Rectangle process, and then the expected cost increase is at most:

\[
\sum_{\substack{i > p \cdot d_u \\
    s.t. \ i / d_u \ is \ power \ of \ 2}} p[r | d = i \ and \ E_2] \cdot O(log \left( \frac{i}{p \cdot d_u} \right) \cdot (c_G + c_L)/p) \leq O((c_G + c_L)/p).
\]

Second, if $d$ was produced after $s$ in the dissection, namely $|s| > |d|$. Let $E_3$ denote the event that $|s| > |d|$ 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:

\[
\sum_{\substack{d_u < |s| \\
    s.t. \ i / d_u \ is \ power \ of \ 2}} p[r | d = i \ and \ E_3] \cdot (d_u + i/p)
\]
The Unreasonable Success of Local Search: Geometric Optimization

**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 $C_R$ although it is served by a facility located on another region in $L$. Client $c$ is served by a facility that belongs to $V_L(L_R)$ 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 $C_R$.

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

$$\sum_{j>p} pr[|s| = j \text{ and } E_3] \sum_{d_u < i < j} 3(d_u/i) \cdot (d_u + i/p) \leq O((c_G + c_L)/p)$$

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

**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 Procedure 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 other 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 $D$ be a dissection produced by procedure 2 and the set of its associated regions $R$. 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 $C_R := V_G(G_R) \setminus (C_L \cap (V_L(L_R)))$. This set contains the clients served by $G_R$ in $G$ except those that belong to $C_L$ and that are served by $L_R$ in $L$. See Fig. 3 for an illustration. Additionally, we define $\Delta_R := V_L(L_R) \setminus V_G(G_R)$.

### 4.3 Facility Location

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

**Theorem 4.5.** Algorithm 1 produces a solution $L$ of cost at most $(1 + O(\varepsilon)) \cdot \text{Cost}(\text{OPT})$.

**Proof.** Let OPT be a globally optimum solution and $L$ be a locally optimum solution. By Theorem 4.1 for any $p > 0$ there exists an assignment $E$ for each random dissection $D_p$.

---

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

---
with portals of $L \cup \text{OPT}$, 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 $E = \sum_{c \in C} \max(c_L, c_G) + \mathcal{O}(\log(p)/p \cdot (\sum_{c \in G} (c_G + c_L)))$.

This implies that there exists a dissection $D_p$ for which $E$ has value at most $E$.

Throughout the proof, we consider this dissection $D_p$ and fix $\varepsilon := \log(p)/p$. Let $R$ be the set of regions associated to $D_p$. We start by constructing a solution $G$ based on $\text{OPT}$ and we compare the cost of $L$ to the cost of $G$. The solution $G$ contains all the facilities of $\text{OPT}$ plus some extra facilities. First, it has one facility at each portal of $D_p$. Moreover, 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. 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 $\text{Cost(OPT)} + \mathcal{O}(\varepsilon(|\text{OPT}| + |L|)\varepsilon)$. 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

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

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 $\square$

### 4.4 K-Median

Let $L$ and $\text{OPT}$ 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 Appendix $\square$.

**Lemma 4.6 (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 two following properties. For any cluster $C$,

- $C$ contains at most $\mathcal{O}(1/\varepsilon^n)$ 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 $1/\varepsilon \in \mathbb{N}$. 

For the K-Median problem, Condition(S′,ε) has to be adapted to |S′| ≤ (1 + 3ε)k and |S \ S′| + |S′ \ S| = $O(d/\varepsilon^{d+1})$. Theorem 4.7 still applies to prove the approximation ratio of the adapted Algorithm.

Higher Dimensions. Previous results generalize to any dimension $d$. It leads to algorithms that have exponential dependency in $d$. 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. The Structure Theorem 4.1 generalizes to any dimension $d$ with $O(dp^{d-1})$ portals instead of $p$. For Facility Location, Condition(S′,ε) has to be adapted to $|S \ S'| + |S \ S'| = O(d/\varepsilon^{d+1})$. Thus, Theorem 4.5 still applies to show that the adapted Algorithm provides a $(1 + O(\varepsilon))$ approximation. For the K-Median problem, Condition(S′,ε) has to be adapted to $|S'| \leq (1 + 3\varepsilon)k$ and $|S \ S'| + |S' \ S| = O(d/\varepsilon^{7+d})$. Theorem 4.7 still applies to prove the approximation ratio of the adapted Algorithm.
References

1 Sanjeev Arora. Polynomial time approximation schemes for euclidean TSP and other geometric problems. In 37th Annual Symposium on Foundations of Computer Science, FOCS ’96, Burlington, Vermont, USA, 14-16 October, 1996, pages 2–11, 1996.

2 Sanjeev Arora. Nearly linear time approximation schemes for euclidean TSP and other geometric problems. In 38th Annual Symposium on Foundations of Computer Science, FOCS ’97, Miami Beach, Florida, USA, October 19-22, 1997, pages 554–563, 1997.

3 Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for euclidean k-medians and related problems. In Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998, pages 106–113, 1998.

4 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.

5 Jillian Beardwood, J. H. Halton, and J. M. Hammersley. The shortest path through many points. *Mathematical Proceedings of the Cambridge Philosophical Society*, 55:299–327, 10 1959.

6 Barun Chandra, Howard J. Karloff, and Craig A. Tovey. New results on the old k-opt algorithm for the TSP. In Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms. 23-25 January 1994, Arlington, Virginia., pages 150–159, 1994.

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

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

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

10 Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. *Comput. Geom.*, 28(2-3):89–112, 2004.

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

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

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

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

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

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

17 Luis A. A. Meira and Flávio Keidi Miyazawa. A continuous facility location problem and its application to a clustering problem. In *Proceedings of the 2008 ACM Symposium on Applied Computing (SAC)*, Fortaleza, Ceara, Brazil, March 16-20, 2008, pages 1826–1831, 2008.

18 Olaf Mersmann, Bernd Bischl, Jakob Bossek, Heike Trautmann, Markus Wagner, and Frank 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.
The Unreasonable Success of Local Search: Geometric Optimization

19 Joseph 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.

20 Nabil H. Mustafa and Saurabh Ray. PTAS for geometric hitting set problems via local search. In *Proceedings of the 25th ACM Symposium on Computational Geometry, Aarhus, Denmark, June 8-10, 2009*, pages 17–22, 2009.

21 Satish Rao and Warren D. Smith. Approximating geometrical graphs via 'spanners' and 'banyans'. In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 540–550, 1998.

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

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

A The Traveling Salesman Problem

![Image](image1)

**Figure 5** The tour on the right is k-optimal for any \( k = o(\sqrt{L}) \) but is \( (2 - O(1/k)) \) times longer than the tour on the left.

B Polynomial-Time Local Search Algorithms

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

**Proof.** Let Cost\((L)\) denote the cost of a locally optimum solution and Cost\((S_0)\) denotes the cost of the initial solution, then the number of steps in the algorithm is at most

\[
\log(\frac{\text{Cost}(S_0)}{\text{Cost}(L)}) \leq \log(\frac{1}{1-1/n})
\]

Since the cost of any minimal solution \( S_0 \) is at most \( O(n) \) (up to rescaling the distances) and as \( \log(n) \) and \( \log(\text{Cost}(L)) \) are polynomial in the input size, the algorithm terminates after polynomially many local search steps which are executed in polynomial time.

C The Structure Theorem

*Lemma C.1.* 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 procedure 2 separates \( v \) and \( l \) for the first time, then length\((s) \leq 5d\).
Proof of Lemma 4.2. Let $s$ be the first line that separates $l$ from $v$ in the dissection. Let $B$ be the last rectangle that contained both $v$ and $l$. Let $B'$ be the minimal rectangle that contains all the facilities of $B$ and let $B''$ be the square centered on $B'$ that have sidelength $5/3$ times the maximum sidelength of $B'$. $B''$ is thus the square that produced $s$. Since the Sub-Rectangle process focus on the intersection between $B''$ and $B$, the length of $s$ is at most the sidelength of $B''$.

Moreover $v$ is not in the rectangle and there is no facility in, at least, the first fifth of the square, $l$ is thus located on the middle part of $B''$ and it follows that length($s$)/5 $\leq d$.

\begin{lemma}
Let $L$ be a set of facilities. Let $v \in \mathbb{R}^2$, $l \in L$, $d_0 = \text{dist}(p, l)$. Suppose that $v$ and $l$ are separated first by a vertical line $s$ in the procedure 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 $d_1/2$ or (ii) smaller than $8d_0$.
\end{lemma}

Proof of Lemma 4.3. First, if $s$ was produced by the Sub-Rectangle process then by Lemma 4.2 $s$ has length at most $5d_0$.

We then assume that $s$ was produced by the Cut-Rectangle process. We consider the last rectangle $R$ (in the top-down order) that contained both $v$ and $l$ and let $r$ be the length of the largest side of $R$. Let $u$ be the closest open facility located to the left of $v$. Suppose now that length($s$) $< d_1/2$. Then $R$ does not contain $u$ (otherwise it has a size greater than $d_1$) and contain $v$. Note also that $R$ was produced by the Sub-Rectangle process and so, since there is no facility to the left of $v$, the left boundary of the rectangle $R$ is at distance at most $r/5$ of $v$.

Now, since $s$ cuts the edge $(v, l)$ and since $s$ was picked in the middle third of $R$, the edge $(v, l)$ must intersect the middle third of $R$, and so,

$$d_0 \geq r/3 - r/5$$

$$\frac{15}{2} d_0 \geq r \geq \text{length}(s)$$

Lemma C.3. Let $E_0(d, s)$ denote the event that an edge $e$ of length $d$ is separated by a cutting line of sidelength $s$ that is produced by Cut-Rectangle. Then, $\Pr[E_0(d, s)] \leq 3d/s$.

Proof of Lemma 4.4. We consider the dissection tree. Let $R_0$ be the bounding region (and so, the root of the tree). If $R_0$ has sidelength $s$ then the probability that $e$ is cut by line of sidelength $s$ is $3d/s$. Else, it does not matter if $R_0$ cuts $e$ or not and in any case, we now look at the children of $R_0$ that contain $e$; say $R_1$ and $R_2$. If $R_1$ or $R_2$ has sidelength $s$ then the probability that $e$ is cut by a line of sidelength $s$ is then at most $3d/s$. Else, we go deeper in the tree until we reach the rectangles that contain $e$ and have sidelength $s$. The probability that $e$ is cut by such a rectangle is thus at most $3d/s$. Hence, the probability that $e$ is cut by a line of sidelength $s$ is thus at most $3d/s$.

\begin{definition}
Algorithm produces a solution $L$ of cost at most $(1 + O(\varepsilon)) \cdot \text{Cost(OPT)}$.
\end{definition}
Proof. Proof of Theorem 4.5

Let OPT be a globally optimum solution and L be a locally optimum solution. By Theorem 4.4, for any p > 0 there exists an assignment E for each random dissection D_p with portals of \( L \cup \text{OPT} \), 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
\[
E = \sum_{c \in C} \max(c_L, c_G) + O(\log(p)/p \cdot \sum_{c \in C} (c_G + c_L)).
\]
This implies that there exists a dissection \( D_p \) for which \( E \) has value at most \( E \).

Throughout the proof, we consider this dissection \( D_p \) and fix \( \varepsilon := \log(p)/p \). Let \( R \) be the set of regions associated to \( D_p \). 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 a facility at each portal of \( L \) and \( G \) has a facility at each portal of \( L \cup \text{OPT} \). Moreover, 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.

Since there are \( O(\varepsilon^2(|G| + |L|) \) regions and that for each region \( R \) uses at most \( 1/\varepsilon \) extra facilities, the cost of \( G \) is at most \( \text{Cost(OPT)} + O(\varepsilon(|OPT| + |L|)) \). We now prove that the cost of \( L \) is at most \( (1 + O(\varepsilon))/(1 - O(\varepsilon)) \) times the cost of \( G \), namely
\[
|L| \cdot f + \sum_{c \in C} c_L \leq (1 + O(\varepsilon))/(1 - O(\varepsilon))(|G| \cdot f + \sum_{c \in C} c_G).
\]

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 \( |L_R \setminus G_R| + |G_R \setminus L_R| = O(\varepsilon^3) \), the locality argument applies. Namely, we have
\[
(|G_R| - |L_R|)f + \sum_{c \notin C_R \cup \Delta_R} c_L + \sum_{c \in C_R \cup \Delta_R} c_G + \sum_{c \in \Delta_R} c_E \geq (1 - 1/n)(|L|f + \sum_{c} c_L).
\]
Rearranging an summing over all region \( R \) of \( R \), we derive
\[
\sum_{R \in R} \left( (|G| - |L|)f + \sum_{c \in C_R \cup \Delta_R} (c_G - c_L) + \sum_{c \in \Delta_R} (c_E - c_L) \right) \geq -\frac{|R|}{n} \cdot \text{Cost}(L). \tag{1}
\]
We now focus on proving an upper bound on the left-hand side of the above equation. We split the sum over \( \Delta_R \) depending on whether \( c \) is in \( C_L \) or \( C_G \). By Theorem 4.4
\[
\sum_{c \in \Delta_R \cap C_G} (c_E - c_L) \leq \sum_{c \in \Delta_R \cap C_G} (c_L - c_L + O(\varepsilon \cdot (c_G + c_L))),
\]
and
\[
\sum_{c \in \Delta_R \cap C_L} (c_E - c_L) \leq \sum_{c \in \Delta_R \cap C_L} (c_G - c_L + O(\varepsilon \cdot (c_G + c_L))).
\]
Replacing in Inequality 1
\[
(|G| - |L|)f + \sum_{R \in R} \left( \sum_{c \in C_R \cup \Delta_R} (c_G - c_L) + \sum_{c \in \Delta_R \cap C_G} (c_E - c_L) \right) + \sum_{c \in C} O(\varepsilon (c_G + c_L)) \geq -\frac{|R|}{n} \cdot \text{Cost}(L).
\]
Definition of $C_R$ leads to
\[
(|G| - |L|)f + \sum_{c \in C} (CG - CL) + \mathcal{O}\left(\sum_{c \in C} \varepsilon \cdot (CG + CL)\right) \geq -\frac{|R|}{n} \text{Cost}(L).
\]

Since $|R| = \mathcal{O}(\varepsilon^2 \cdot n)$, we conclude
\[
(1 + \mathcal{O}(\varepsilon)) \left(|G|f + \sum_{c \in C} CG\right) \geq (1 - \mathcal{O}(\varepsilon)) \left(|L|f + \sum_{c \in C} CL\right)
\]
and the Theorem follows.

\section{K-Median}

\textbf{Lemma E.1 (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 property. For any cluster $C$,
\begin{itemize}
  \item $C$ contains at most $\mathcal{O}(1/\varepsilon^5)$ elements of $\mathcal{R}$, namely $|C| = \mathcal{O}(1/\varepsilon^5)$;
  \item 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 $|C|/\varepsilon$
\end{itemize}

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

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

\textbf{Proof of Lemma 4.6} We first define for each set $r_i$, $v(r_i) := |L \cap r_i| - |G \cap r_i| - 1/\varepsilon$.

The assumption on the total number of elements of $L$ and $G$ can be rewritten as $\sum_{r_i} v(r_i) \geq \varepsilon G > 0$.

Besides, the cardinality bounds on $r_i$ imply that $v(r_i)$ is an integer in the range $[-1/\varepsilon^2 - 1/\varepsilon, 1/\varepsilon^2 - 1/\varepsilon]$.

We need to construct a clustering of $\mathcal{R}$ into small clusters such that for each cluster $C$,
\[
\sum_{r_i \in C} v(r_i) \geq 0.
\]
We exhibit an algorithm that constructs such a clustering. For any set $r_i$ such that $v(r_i) = 0$, we create a new part that contains only this set. This part trivially satisfies the above property.

We now consider the remaining sets. While there exists $1 < i, j$, such that $1/\varepsilon^2 + 1/\varepsilon < |v_i|, |v_j|$, we take $i$ sets from $v_j$ and $j$ sets from $v_i$ and create a new part that contains them all. This part satisfies the property of the Lemma and contains at most $2/\varepsilon^2$ sets of $\mathcal{R}$.

We now turn to the last case, namely $\forall j \geq 0, |v_j| \leq 1/\varepsilon^2 + 1/\varepsilon$ (or symmetrically $\forall j \leq 0$, $|v_j| \leq 1/\varepsilon^2 + 1/\varepsilon$). We claim that it is possible to make on last part containing all the remaining sets and that this part satisfies the property of the Lemma and has size $\mathcal{O}(1/\varepsilon^5)$. We start by proving that, after each step $s$ of the above algorithm, the following invariant holds
\[
(1 + \frac{2\varepsilon}{1 - \varepsilon})|G_s| \leq |L_s| \leq (1 + \frac{4\varepsilon}{1 - 2\varepsilon})|G_s|,
\]
where \( L_s \) and \( G_s \) are the number of elements of type \( L \) and \( G \) respectively that are not contained in any part after step \( s \).

This is true at the beginning of the algorithm. We show that it is true all the way to the last step. Assume that it holds after step \( s \), we prove that it is true after step \( s + 1 \).

Let \( P \) be the part created at step \( s \). This part contains say \( P_G \) elements of \( G \) and so, \( P_G + |P|/\varepsilon \) elements of \( L \). By induction hypothesis, Inequality 2 holds. Hence, by expressing \( L_{s-1} \) and \( G_{s-1} \) in terms of \( L_s \) and \( G_s \), it follows that

\[
(1 + \frac{2\varepsilon}{1 - \varepsilon})(|G_s| + P_G) \leq L_s + P_G + |P|/\gamma \leq (1 + \frac{4\varepsilon}{(1 - 2\varepsilon)})(|G_s| + P_G).
\]

By definition of the \( s_i \),

\[
|P|/2\varepsilon^2 \leq 2P_G + |P|/\varepsilon \leq |P|/\varepsilon^2.
\]

Rearranging and replacing in the inequalities above, it follows

\[
\frac{2\varepsilon}{1 - \varepsilon} P_G \leq |P|/\varepsilon \leq \frac{4\varepsilon}{1 - 2\varepsilon} P_G.
\]

At final step \( f \), the upper and lower bounds on \( L_f \) induced by Inequality 2 implies that the final part has size at most \( O(1/\varepsilon^5) \) and satisfies the properties of the Lemma.

**Theorem E.2.** Algorithm 4 for K-Median produces a solution \( L \) that is a \( (1 + O(\varepsilon), 1 + O(\varepsilon)) \) bi-criteria approximation.

**Proof of Theorem 4.7.** Remark first that solution \( L \) uses \( (1 + O(\varepsilon))k \) facilities. We now show that the cost of solution \( L \) is at most \( 1 + O(\varepsilon) \) times higher than the cost of the optimal solution.

Recall that by Theorem 4.1, for any \( p > 0 \) there exists an assignment \( E \) for each random dissection \( D_p \) of \( L \cup \text{OPT} \) with portals, such that for any client \( c \) and region \( R \), if \( c(L) \in R \) and \( c(\text{OPT}) \not\in R \) 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 \( E = \sum_{c \in C} \max(c_L, c_{\text{OPT}}) + O(\log(p)/p \cdot (\sum_{c \in C} (c_{\text{OPT}} + c_L))) \).

This implies that there exists a dissection \( D_p \) for which \( E \) has value at most \( E \). Throughout the proof, we consider such a dissection \( D_p \) and fix \( \varepsilon := \log(p)/p \). Let \( R \) be the set of regions associated to \( D_p \).

We prove that the cost of \( L \) is at most \( (1 + O(\varepsilon))/(1 - O(\varepsilon)) \) times the cost of \( S \), namely

\[
\sum_{c \in C} c_L \leq \frac{1 + O(\varepsilon)}{1 - O(\varepsilon)} \sum_{c \in C} c_{\text{OPT}}.
\]

Let \( \mathcal{P} \) be a clustering of the regions satisfying the properties of Lemma 4.6 (depending on \( L \) and \( \text{OPT} \)). We start by constructing a solution \( G \) based on \( \text{OPT} \) 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 4.5. Namely, the solution \( G \) contains all the facilities of \( \text{OPT} \) plus some extra facilities: one facility at each portal of \( D_p \) 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 \( |S_P \setminus L| + |L \setminus S_P| = O(1/\varepsilon^9) \) and the locality argument applies.

We show that, by local optimality, the cost of \( S_P \) is close to the cost of \( L \). We serve the clients of \( C_R \) optimally (namely by the facilities that serve them in \( G \)) and the clients of \( \Delta_R \)
by the facilities located on the portals of $R$ or by the facilities of $L \setminus L_R$, according to the assignment $E$. By local optimality, the cost of replacing $L$ by $S_P$ is greater (up to a factor $(1 - 1/n)$) than the cost of $L$. Namely, we have

$$
\sum_{R \in P} \left( \sum_{c \in C_R \cup \Delta_R} c_L + \sum_{c \in C_R} c_G + \sum_{c \in \Delta_R} c_E \right) \geq (1 - 1/n) \text{Cost}(L).
$$

Rearranging and summing over all part $P$ of $P$,

$$
\sum_{P \in \mathcal{P}} \sum_{R \in P} \left( \sum_{c \in C_R} c_G - c_L + \sum_{c \in \Delta_R} c_E - c_L \right) \geq -\frac{|P|}{n} \cdot \text{Cost}(L).
$$

We now provide an upper bound on the left-hand side of the above equation. We separate the sum over $\Delta_R$ depending on whether $c$ is in $C_L$ or $C_G$.

By Theorem 4.1, we obtain

$$
\sum_{c \in \Delta_R} (c_E - c_L) \leq \sum_{c \in \Delta_R} (c_L - c_L + O(\varepsilon \cdot (c_G + c_L))),
$$

and

$$
\sum_{c \in \Delta_R} (c_E - c_L) \leq \sum_{c \in \Delta_R} (c_G - c_L + O(\varepsilon \cdot (c_G + c_L))).
$$

Replacing in Equation 3 it follows that

$$
\sum_{P \in \mathcal{P}} \sum_{R \in P} \left( \sum_{c \in C_R} c_G - c_L + \sum_{c \in \Delta_R} c_E - c_L \right) + \sum_{c \in \mathcal{C}} O(\varepsilon \cdot (c_G + c_L)) \geq -\frac{|P|}{n} \cdot \text{Cost}(L).
$$

By the definition of $C_R$, the left-hand side is exactly

$$
\sum_{c \in \mathcal{C}} (c_G - c_L) + \sum_{c \in \mathcal{C}} O(\varepsilon \cdot (c_G + c_L)).
$$

Since $|P| = O(\varepsilon k)$, we conclude

$$
(1 + O(\varepsilon)) \cdot \sum_{c \in \mathcal{C}} c_G \geq (1 - O(\varepsilon)) \sum_{c \in \mathcal{C}} c_L
$$

and the Theorem follows.

\[\blacksquare\]

**Proof.** We first define for each set $r_i$, $v(r_i) := |L \cap r_i| - |G \cap r_i| - 1/\varepsilon$. The assumption on the total number of elements of $L$ and $G$ can be rewritten as $\sum_{r_i} v(r_i) \geq \varepsilon G > 0$.

Besides, the cardinality bounds on $r_i$ imply that $v(r_i)$ is an integer in the range $[-1/\varepsilon^2 - 1/\varepsilon, 1/\varepsilon - 1/\varepsilon]$.

We need to construct a clustering of $R$ into small clusters such that for each cluster $C$, $\sum_{r_i \in C} v(r_i) \geq 0$. We exhibit an algorithm that constructs such a clustering. For any set $r_i$
such that \( v(r_i) = 0 \), we create a new cluster that contains only this set. This part trivially satisfies the above property.

We now consider the remaining sets. While there exists \( j < 0 < i \), such that the number of sets \( r \) that have value \( v(r) = i \) and the number of sets \( r' \) that have value \( v(r') = j \) is at least \( 1/\varepsilon^2 + 1/\varepsilon \), We create a new cluster that contains \( i \) sets of value \( j \) and \( -j \) sets of value \( i \). This cluster satisfies the property of the Lemma and contains at most \( 2/\varepsilon^2 \) sets of \( R \).

We now turn to the last case, namely for any \( i > 0 \), the number of sets \( r \) that have value \( v(r) = i \) is at most \( 1/\varepsilon^2 + 1/\varepsilon \) (or for any \( j > 0 \), the number of sets \( r \) that have value \( v(r) = j \) is at most \( 1/\varepsilon^2 + 1/\varepsilon \)).

We claim that, by the assumption on the cardinality of \( L \) and \( G \), the total number of remaining sets is at most \( \mathcal{O}(1/\varepsilon^5) \) and so it is possible to make one last cluster containing all the remaining sets and that satisfies the property of the Lemma The proof of the above claim is deferred to appendix E.\[\square\]