Smoothed Analysis of Partitioning Algorithms for Euclidean Functionals

Markus Bläser[1], Bodo Manthey[2], and B. V. Raghavendra Rao[1]

[1] Saarland University, Department of Computer Science
mblaeser/bvrr@cs.uni-saarland.de
[2] University of Twente, Department of Applied Mathematics
b.manthey@utwente.nl

Abstract. Euclidean optimization problems such as TSP and minimum-length matching admit fast partitioning algorithms that compute near-optimal solutions on typical instances.
We develop a general framework for the application of smoothed analysis to partitioning algorithms for Euclidean optimization problems. Our framework can be used to analyze both the running-time and the approximation ratio of such algorithms. We apply our framework to obtain smoothed analyses of Dyer and Frieze's partitioning algorithm for Euclidean matching, Karp's partitioning scheme for the TSP, a heuristic for Steiner trees, and a heuristic for degree-bounded minimum-length spanning trees.

1 Introduction

Euclidean optimization problems are a natural class of combinatorial optimization problems. In a Euclidean optimization problem, we are given a set X of points in $\mathbb{R}^2$. The topology used is the complete graph of all points, where the Euclidean distance $\|x - y\|$ is the length of the edge connecting the two points $x, y \in X$.

Many such problems, like the Euclidean traveling salesman problem [18] or the Euclidean Steiner tree problem [11], are NP-hard. For others, like minimum-length perfect matching, there exist polynomial-time algorithms. But still these polynomial-time algorithms are sometimes too slow to solve large instances. Thus, fast heuristics to find near-optimal solutions for Euclidean optimization problems are needed.

A generic approach to design heuristics for Euclidean optimization problems are partitioning algorithms: They divide the Euclidean plane into a number of cells such that each cell contains only a small number of points. This allows us to quickly find an optimum solution for our optimization problem for the points within each cell. Finally, the solutions of all cells are joined in order to obtain a solution to the whole set of points. This joining should be done quickly to obtain a fast algorithm.

Although this is a rather simple ad-hoc approach, it works surprisingly well and fast in practice [13, 20]. This is at stark contrast to the worst-case performance of partitioning algorithms: They can both be very slow and output

solutions that are far from the optimal solutions. Thus, as is often the case, worst-case analysis is far too pessimistic to explain the performance of partitioning algorithms. The reason for this is that worst-case analysis is dominated by artificially constructed pathological instances that often do not resemble practical instances.

Both to explain the performance of partitioning algorithms and to gain probabilistic insights into the structure and value of optimal solutions of Euclidean optimization problems, the average-case performance of partitioning algorithms has been studied a lot. In particular, Steele [27] proved complete convergence of Karp's partitioning algorithm [15] for Euclidean TSP. Also strong central limit theorems for a wide range of optimization problems are known. We refer to Steele [28] and Yukich [31] for comprehensive surveys.

However, also average-case analysis has its drawback: Random instances usually have very specific properties with overwhelming probability. This is often exploited in average-case analysis: One shows that the algorithm at hand performs very well if the input has some of these properties. But this does not mean that typical instances share these properties. Thus, although a good average-case performance can be an indicator that an algorithm performs well, it often fails to explain the performance convincingly.

In order to explain the performance of partitioning schemes for Euclidean optimization problems, we provide a smoothed analysis. Smoothed analysis has been introduced by Spielman and Teng [23] in order to explain the performance of the simplex method for linear programming. It is a hybrid of worst-case and average-case analysis: An adversary specifies an instance, and this instance is then slightly randomly perturbed. The perturbation can, for instance, model noise from measurement. Since its invention in 2001, smoothed analysis has been applied in a variety of contexts [3, 4, 9, 22]. We refer to Spielman and Teng [24] for a survey.

We develop a general framework for smoothed analysis of partitioning algorithms for optimization problems in the Euclidean plane (Section 3). We consider the most general model where the adversary specifies n density functions $f_1, \ldots, f_n : [0,1]^2 \to [0,\phi]$. The parameter ϕ controls the adversary's power: The larger ϕ, the more powerful the adversary. (See Section 2.2 for a formal explanation of the model.) We analyze the expected running-time and approximation performance of a generic partitioning algorithm under this model. The smoothed analysis of the running-time for partitioning algorithms depends crucially on the convexity of the worst-case bound of the running-time of the problem under consideration. The main tool for the analysis of the expected approximation ratio is Rhee's isoperimetric inequality [21].

We apply the general framework to obtain smoothed analyses of partitioning algorithms for Euclidean matching, Karp's partitioning scheme for the TSP, Steiner trees, and degree-bounded minimum spanning trees in the Euclidean plane. Table 1 shows an overview. To summarize, for $\phi \leq \log^{O(1)} n$, Dyer and Frieze's partitioning algorithm [7] has an almost linear running-time, namely $O(n \log^{O(1)} n)$. For $\phi \in o(\log^2 n)$, its expected approximation ratio tends to 1

problem	running-time	approximation ratio	reference
matching [7]	$O(n\phi^2 \log^4 n)$	$1 + O(\sqrt{\phi}/\log n)$	Corollary 4.2
TSP [15]	poly(n)	$1 + O(\sqrt{\phi/\log n})$	Corollary 5.1
Steiner tree [14]	poly(n)	$1 + O(\sqrt{\phi/\log n})$	Corollary 6.1
degree-bounded MST	poly(n)	$1 + O(\sqrt{\phi \log\log n/\log n})$	Corollary 7.1

Table 1. Smoothed bounds for some Euclidean optimization problems.

as n increases. The approximation ratios of the partitioning algorithms for TSP and Steiner trees tend to 1 for $\phi \in o(\log n)$. For degree-bounded spanning trees, this is the case for $\phi \in o(\log n/\log\log n)$. Our general framework is applicable to many other partitioning algorithms as well.

Due to space limitations, many proofs are omitted and will appear in the full version of the paper.

2 Preliminaries

2.1 Euclidean Functionals

A *Euclidean functional* is a function $\mathsf{F} : ([0,1]^2)^\star \to \mathbb{R}$ that maps a finite point set $X \subseteq [0,1]^2$ to a real number $\mathsf{F}(X)$. The following are examples of Euclidean functionals:

- MM maps a point set to the length of its minimum-length perfect matching (length means Euclidean distance, one point is left out if the cardinality of the point set is odd).
- TSP maps a point set to the length of its shortest Hamiltonian cycle, i.e., to the length of its optimum traveling salesman tour.
- MST maps a point sets to the length of its minimum-length spanning-tree.
- ST maps a point set to the length of its shortest Steiner tree.
- dbMST maps a point set to the length of its minimum-length spanning tree, restricted to trees of maximum degree at most b for some given bound b.

The Euclidean functionals that we consider in this paper are all associated with an underlying combinatorial optimization problem. Thus, the function value $\mathsf{F}(X)$ is associated with an optimum solution (minimum-length perfect matching, optimal TSP tour, ...) to the underlying combinatorial optimization problem. In this sense, we can design approximation algorithms for F: Compute a (near-optimal) solution (where it depends on the functional what a solution actually is; for instance, a perfect matching), and compare the objective value (for instance, the sum of the lengths of its edges) to the function value.

We follow the notation of Frieze and Yukich [10,31]. A Euclidean functional F is called *smooth* [21,31] if there is a constant C such that $\big|\mathsf{F}(X \cup Y) - \mathsf{F}(X)\big| \leq C\sqrt{|Y|}$ for all finite $X, Y \subseteq [0,1]^2$.

Let $C_1, \ldots, C_s$ be a partition of $[0,1]^2$ into rectangles. We call each C_ℓ a *cell*. Note that the cells are not necessarily of the same size. For a finite set $X \subseteq [0,1]^2$ of n points, let $X_\ell = X \cap C_\ell$ be the points of X in cell C_ℓ. Let $n_\ell = |X_\ell|$ be the number of points of X in cell C_ℓ. Let $\text{diameter}(C_\ell)$ be the diameter of cell C_ℓ.

We call F *sub-additive* if

$$\mathsf{F}(X) \leq \sum_{\ell=1}^{s} \big(\mathsf{F}(X_\ell) + \text{diameter}(C_\ell)\big).$$

F is called *super-additive* if

$$\mathsf{F}(X) \geq \sum_{\ell=1}^{s} \mathsf{F}(X_\ell).$$

The Euclidean functions TSP, MM and MST are smooth and sub-additive [27, 28, 31].

A combination of sub-additivity and super-additivity for a Euclidean functional F is a sufficient (but not a necessary) condition for the existence of a partitioning heuristic for approximating F. We will present such a generic partitioning heuristic in Section 3. Following Frieze and Yukich [10], we define a slightly weaker additivity condition that is sufficient for the performance analysis of partitioning algorithms.

Frieze and Yukich [10] call a Euclidean function F *near-additive* if, for all partitions $C_1, \ldots, C_s$ of $[0,1]^2$ into cells and for all finite $X \subseteq [0,1]^2$, we have

$$\big|\mathsf{F}(X) - \textstyle\sum_{\ell=1}^{s} \mathsf{F}(X_\ell)\big| \leq O\big(\sum_{\ell=1}^{s} \text{diameter}(C_\ell)\big). \tag{1}$$

It is not hard to see that, if F is sub-additive and super-additive, then F is also near-additive. Euclidean functionals such as TSP, MM, and MST are sub-additive but not super-additive. However, these functionals can be approximated by their corresponding canonical *boundary functionals*, which are super-additive [10, 31]. Yukich [31] has shown that this is a sufficient condition for a Euclidean functional to be near-additive.

Proposition 2.1 (Yukich [31, Lemma 5.7]). *Let F be a sub-additive Euclidean functional. Let F_{B} be a super-additive functional that well-approximates F. (This means that $|\mathsf{F}(X) - \mathsf{F}_{\mathrm{B}}(X)| = O(1)$ for all finite $X \subseteq [0,1]^2$.) Then F is near-additive.*

The functionals MM, TSP, MST, ST, and dbMST are near-additive.

Limit theorems are a useful tool for the analysis of Euclidean functionals. Rhee [21] proved the following limit theorem for smooth Euclidean functionals over $[0,1]^2$. We will mainly use it to bound the probability that F assumes a too small function value.

Theorem 2.2 (Rhee [21]). *Let X be a set of n points drawn independently according to identical distributions from $[0,1]^2$. Let F be a smooth Euclidean functional. Then there exist constants C and C' such that for all $t > 0$, we have*

$$\mathbb{P}\left[\big|\mathsf{F}(X) - \mathbb{E}\big[\mathsf{F}(X)\big]\big| > t\right] \leq C \cdot \exp\big(-C' t^4/n\big).$$

Remark 2.3. *Rhee proved Theorem 2.2 for the case that $x_1, \ldots, x_n$ are identically distributed. However, as pointed out by Rhee herself [21], the proof carries over to the case when $x_1, \ldots, x_n$ are drawn independently but their distributions are not necessarily identical.*

2.2 Smoothed Analysis

In the classical model of smoothed analysis [23], an adversary specifies a point set $\bar{X}$, and then this point set is perturbed by independent identically distributed random variables in order to obtain the input set X. A different view-point is that the adversary specifies the means of the probability distributions according to which the point set is drawn. This model has been generalized as follows [4]: Instead of only specifying the mean, the adversary can specify a density function for each point, and then we draw the points independently according to their density functions. In order to limit the power of the adversary, we have an upper bound ϕ for the densities: The adversary is allowed to specify any density function $[0,1]^2 \to [0,\phi]$. If $\phi = 1$, then this boils down to the uniform distribution on the unit square $[0,1]^2$. If ϕ gets larger, the adversary becomes more powerful and can specify the location of the points more and more precisely. The role of ϕ is the same as the role of $1/\sigma$ in classical smoothed analysis, where σ is the standard deviation of the perturbation. We summarize this model formally in the following assumption.

Assumption 2.4. *Let $\phi \geq 1$. An adversary specifies n probability density functions $f_1, \ldots, f_n : [0,1]^2 \to [0,\phi]$. We write $f = (f_1, \ldots, f_n)$ for short. Let $x_1, \ldots, x_n \in [0,1]^2$ be n random variables where x_i is drawn according to f_i, independently from the other points. Let $X = \{x_1, \ldots, x_n\}$.*

If the actual density functions f matter and are not clear from the context, we write $X \sim f$ to denote that X is drawn as described above. If we have a performance measure P for an algorithm (P will be either running-time or approximation ratio in this paper), then the smoothed performance is $\max_f\big(\mathbb{E}_{X\sim f}[P(X)]\big)$. Note that the smoothed performance is a function of the number n of points and the parameter ϕ.

Let F be a Euclidean functional. For the rest of this paper, let $\mu_{\mathsf{F}}(n,\phi)$ be a lower bound for the expected value of F if X is drawn according to the probabilistic model described above. More precisely, μ_{F} is some function that fulfills $\mu_{\mathsf{F}}(n,\phi) \leq \min_f\big(\mathbb{E}_{X\sim f}[\mathsf{F}(X)]\big)$. The function μ_{F} comes into play when we have to bound F (or: the objective value of an optimum solution) from below in order to analyze the approximation ratio.

3 Framework

In this section, we present our framework for the performance analysis of partitioning heuristics for Euclidean functionals. Let A_{opt} be an optimal algorithm for some smooth and near-additive Euclidean functional F, and let A_{join} be an

algorithm that combines solutions for each cell into a global solution. We assume that A_{join} runs in time linear in the number of cells. Then we obtain the following algorithm, which we call A.

Algorithm 3.1 (generic algorithm A). *Input: set $X \subseteq [0,1]^2$ of n points.*

1. *Divide $[0,1]^2$ into s cells $C_1, \ldots, C_s$.*
2. *Compute optimal solutions for each cell using A_{opt}.*
3. *Join the s partial solutions to a solution for X using A_{join}.*

We use the following assumptions in our analysis and mention explicitly whenever they are used.

Assumption 3.2.
1. *$\phi \in O(s)$. This basically implies that the adversary cannot concentrate all points in a too small number of cells.*
2. *$\phi \in \omega(s \log n/n)$. This provides a lower bound for the probability mass in a "full" cell, where full is defined in Section 3.1.*
3. *$\phi \in o(\sqrt{n/\log n})$. With this assumption, the tail bound of Theorem 2.2 becomes sub-polynomial.*

These assumptions are not too restrictive: For the partitioning algorithms we analyze here, we have $s = O(n/\log^{O(1)} n)$. Ignoring poly-logarithmic terms, the first and third assumption translate roughly to $\phi = O(n)$ and $\phi = o(\sqrt{n})$, respectively. But $\phi = \Theta(\sqrt{n})$ suffices for the adversary to specify an individual small square for each point, thus we can expect to approach almost worst-case behavior for $\phi = \Omega(\sqrt{n})$. The second assumption roughly says $\phi = \omega(1)$. But for $\phi = O(1)$, we can expect (almost) average-case behavior.

3.1 Smoothed Running-Time

Many of the schemes that we analyze choose the partition in such a way that we have a worst-case upper bound on the number of points in each cell. Other algorithms, like the one for matching, have a fixed partition independent of the input points. In the latter case, the running-time also depends on ϕ.

Let $T(n)$ denote the worst-case running-time of A_{opt} on n points. Then the running-time of A is bounded by $\sum_{\ell=1}^{s} T(n_\ell) + O(s)$. The expected running-time of A is thus

$$\sum_{\ell=1}^{s} \mathbb{E}[T(n_\ell)] + O(s). \tag{2}$$

For the following argument, we assume that T (the running-time of A_{opt}) is a monotonically increasing, convex function, and that the locations of the cells are fixed and all their volumes are equal. (The assumption about the cells is not fulfilled for all partitioning heuristics. For instance, Karp's partitioning scheme [15] chooses the cells not in advance but based on the actual point set. However, in Karp's scheme, the cells are chosen in such a way that there is a good worst-case upper bound for the number of points per cell, so there is no need for a smoothed analysis.) By abusing notation a bit, let $f_i(C_\ell) = \int_{C_\ell} f_i(x)\,\mathrm{d}x$ be the cumulative density of f_i in the cell C_ℓ. Since f_i is bounded from above by ϕ, we

have $f_i(C_\ell) \leq \phi/s$. Let $f(C_\ell) = \sum_{i=1}^{n} f_i(C_\ell)$. Note that $f_i(C_\ell) = \mathbb{P}(x_i \in C_\ell)$ and $f(C_\ell) = \mathbb{E}[n_\ell]$.

We call a cell C_ℓ *full* with respect to f if $f(C_\ell) = n\phi/s$. We call C_ℓ *empty* if $f(C_\ell) = 0$. Our bound (2) on the running-time depends only on the values $f_1(C_\ell), \ldots, f_n(C_\ell)$, but not on where exactly within the cells the probability mass is assumed.

Our goal is now to show that the adversary, in order to make our algorithm as slow as possible, will make as many cells as possible full. Note that there are at most $\lfloor s/\phi \rfloor$ full cells. Assume that we have $\lfloor s/\phi \rfloor$ full cells and at most one cell that is neither empty nor full. Then the number of points in any full cell is a binomially distributed random variable B with parameters n and ϕ/s. By linearity of expectation, the expected running-time is bounded by $\left(\left\lfloor \frac{s}{\phi} \right\rfloor + 1\right) \cdot \mathbb{E}\big[T(B)\big] + O(s)$. Since $\phi = O(s)$ by Assumption 3.2 (1), this is bounded by $O\big(\frac{s}{\phi} \cdot \mathbb{E}[T(B)] + s\big)$. If T is bounded by a polynomial, then this evaluates to $O\big(\frac{s}{\phi} \cdot T(n\phi/s) + s\big)$ by the following Lemma 3.3. This lemma can be viewed as "Jensen's inequality in the other direction" with $p = \phi/s$ for $\phi \in \omega(s \log n/n)$. The latter is satisfied by Assumption 3.2 (2).

Lemma 3.3 (inverse Jensen's inequality). *Let T be any convex, monotonically increasing function that is bounded by a polynomial, and let B be a binomially distributed random variable with parameters $n \in \mathbb{N}$ and $p \in [0,1]$ with $p \in \omega(\log n/n)$. Then $\mathbb{E}[T(B)] = \Theta(T(\mathbb{E}[B]))$.*

What remains to be done is to show that the adversary will indeed make as many cells as possible full. This follows essentially from the convexity of the running-time. Thus, we obtain the following theorem.

Theorem 3.4. *Assume that the running-time of A_{opt} can be bounded from above by a convex function T that is bounded by a polynomial. Then, under Assumption 2.4 as well as Assumptions 3.2 (1) and (2), the expected running-time of A on input X is bounded by $O\big(\frac{s}{\phi} \cdot T(n\phi/s) + s\big)$.*

3.2 Smoothed Approximation Ratio

The value computed by A can be bounded from above by $\mathsf{A}(X) \leq \sum_{\ell=1}^{s} \mathsf{F}(X_\ell) + J'$, where J' is an upper bound for the cost incurred by joining the solution for the cells. Since F is near-additive, $\mathsf{A}(X) \leq \mathsf{F}(X) + J$ for $J = J' + O(\sum_{\ell=1}^{s} \text{diameter}(C_\ell))$. Dividing by $\mathsf{F}(X)$ yields

$$\frac{\mathsf{A}(X)}{\mathsf{F}(X)} \leq 1 + O\big(\tfrac{J}{\mathsf{F}(X)}\big). \tag{3}$$

For estimating the expected approximation ratio $\mathbb{E}[\mathsf{A}(X)/\mathsf{F}(X)]$ for some algorithm A, the main challenge is that $\mathsf{F}(X)$ stands in the denominator. Thus, even if we know $\mathsf{A}(X)$ precisely, we are basically left with the problem of estimating $\mathbb{E}[1/\mathsf{F}(X)]$. Jensen's inequality yields $1/\mathbb{E}[\mathsf{F}(X)] \leq \mathbb{E}[1/\mathsf{F}(X)]$. But this does not help, as we need upper bounds for $\mathbb{E}[1/\mathsf{F}(X)]$. Unfortunately, such

upper bounds cannot be derived easily from $1/\mathbb{E}[\mathsf{F}(X)]$. The problem is that we need strong upper bounds for the probability that $\mathsf{F}(X)$ is close to 0. Theorem 2.2 is too weak for this. This problem of bounding the expected value of the inverse of the optimum objective value arises frequently in bounding expected approximation ratios [8, 9].

There are two ways to attack this problem: The first and easiest way is if A comes with a worst-case guarantee $\alpha(n)$ on its approximation ratio for instances of n points. Then we can apply Theorem 2.2 to bound $\mathsf{F}(X)$ from below. If $\mathsf{F}(X) \geq \mu_{\mathsf{F}}(n,\phi)/2$, then we can use (3) to obtain a ratio of $1 + O\big(\frac{J}{\mu_{\mathsf{F}}(n,\phi)}\big)$. Otherwise, we obtain a ratio of $\alpha(n)$. If $\alpha(n)$ is not too large compared to the tail bound obtained from Theorem 2.2, then this contributes only little to the expected approximation ratio. The following theorem formalizes this.

Theorem 3.5. *Assume that* A *has a worst-case approximation ratio of* $\alpha(n)$ *for instance consisting of* n *points. Then, under Assumption 2.4, the expected approximation ratio of* A *is*

$$\mathbb{E}\left[\frac{\mathsf{A}(X)}{\mathsf{F}(X)}\right] \leq 1 + O\left(\frac{J}{\mu_{\mathsf{F}}(n,\phi)}\right) + \alpha(n) \cdot \exp\left(-\frac{\mu_{\mathsf{F}}(n,\phi)^4}{Cn}\right).$$

Now we turn to the case that the worst-case approximation ratio of A cannot be bounded by some $\alpha(n)$. In order to be able to bound the expected approximation ratio, we need an upper bound on $\mathbb{E}[1/\mathsf{F}(X)]$. This upper bound is formalized in the following theorem.

Theorem 3.6. *Assume that there exists a* $\beta \leq J$ *and a function* h_n *such that* $\mathbb{P}(F(X) \leq x) \leq h_n(x)$ *for all* $x \in [0,\beta]$. *Then, under Assumption 2.4, the expected approximation ratio of* A *is*

$$\mathbb{E}\left[\frac{\mathsf{A}(X)}{\mathsf{F}(X)}\right] \leq 1 + O\left(J \cdot \left(\frac{1}{\mu_{\mathsf{F}}(n,\phi)} + \frac{\exp\big(-\frac{\mu_{\mathsf{F}}(n,\phi)^4}{Cn}\big)}{\beta} + \int_{1/\beta}^{\infty} h_n\left(\frac{1}{x}\right)\,\mathrm{d}x\right)\right).$$

4 Matching

As a first example, we apply our framework to the matching functional MM defined by the Euclidean minimum-length perfect matching problem. A partitioning algorithm, which we call DF, for approximating MM was proposed by Dyer and Frieze [7]. This algorithm divides $[0,1]^2$ into k^2 equal-sized sub-squares, computes an optimum matching within these cells and combines the solutions using the so-called strip heuristic.

Let $\mathsf{DF}(X)$ be the cost of the matching computed by the algorithm above on input $X = \{x_1, \ldots, x_n\}$, and let $\mathsf{MM}(X)$ be the cost of a perfect matching of minimum total length. Dyer and Frieze showed that $\mathsf{DF}(X)$ converges to $\mathsf{MM}(X)$ with probability 1 if the points in X are drawn according to uniform distributions on $[0,1]^2$ (this corresponds to Assumption 2.4 with $\phi = 1$) and n goes to infinity. We extend this to the case when X is drawn as described in Assumption 2.4.

4.1 Smoothed Running-Time

A minimum-length perfect matching can be found in time $O(n^3)$ [1]. By Theorem 3.4, we get the following corollary.

Corollary 4.1. *Under Assumption 2.4 as well as Assumption 3.2* (1) *and* (2)*, the expected running-time of* DF *on input* X *is at most* $O\big(\frac{n^3\phi^2}{k^4} + k^2\big)$*. If we plug in* $k = \sqrt{n}/\log n$*, we obtain an expected running-time of at most* $O(n\phi^2 \log^4 n)$*.*

4.2 Smoothed Approximation Ratio

To estimate the approximation performance, we have to specify the function $\mu_{\mathsf{MM}}(n, \phi)$. To obtain a lower bound for $\mu_{\mathsf{MM}}(n, \phi)$, let $\mathsf{NN}(X)$ denote the total edge length of the nearest-neighbor graph for the point set $X \subseteq [0, 1]^2$. This means that

$$\mathsf{NN}(X) = \textstyle\sum_{x \in X} \min_{y \in X : y \neq x} \|x - y\|.$$

We have $\mathsf{MM}(X) \geq \mathsf{NN}(X)/2$. We will use $\mathbb{E}[\mathsf{NN}(X)]$ to bound $\mathbb{E}[\mathsf{MM}(X)]$. Next, one shows the tail bound $\mathbb{P}\big(\mathsf{MM}(X) \leq c\big) \leq (2\phi\pi c)^{n/2}$. This allows us to apply Theorem 3.6

Corollary 4.2. *Under Assumption 2.4 and 3.2* (3)*, the expected approximation ratio of* DF *is* $1 + O\big(\frac{\sqrt{\phi}}{\log n}\big)$*.*

Remark 4.3. *1. There exist other partitioning schemes for Euclidean matching [2], which can be analyzed in a similar way.*

2. Instead of a standard cubic-time algorithm, we can use Varadarajan's matching algorithm [30], which has a running-time of $O(m^{1.5} \log^5 m)$ *for* m *points, for computing the optimal matchings within each cell. This improves the running-time bound to* $O(n\sqrt{\phi}\log(n)\log^5(\phi \log n))$*.*

5 Karp's Partitioning Scheme for Euclidean TSP

Karp's partitioning scheme [15] is a well-known heuristic for Euclidean TSP. For a point set $X \subseteq [0, 1]^2$, let $\mathsf{KP}(X)$ denote the cost of the tour through X computed by Karp's scheme. Steele [27] has proved complete convergence of $\mathsf{KP}(X)$ to $\mathsf{TSP}(X)$ with probability 1, if the points are chosen uniformly and independently. Using our framework developed in Section 3, we extend the analysis of KP to the case of non-uniform and non-identical distributions.

Since Karp's scheme chooses the cells adaptively, based on the point set X, our framework for the analysis of the running-time cannot be applied. However, the total running-time of the algorithm is $T(n) = 2^{O(n/k^2)}\,\mathrm{poly}(n/k^2) + O(k^2)$, which is, independent of the randomness, polynomial in n for $k^2 = n/\log n$.

The nice thing about the TSP is that every tour has a worst-case approximation guarantee of at most $\frac{n}{2} \cdot \mathsf{TSP}(X)$. Thus, we can use Theorem 3.5 with $\alpha(n) = n/2$.

Corollary 5.1. *Under Assumption 2.4 as well as Assumption 3.2* (3)*, the expected approximation ratio of* KP *is* $\mathbb{E}\big[\frac{\mathsf{KP}(X)}{\mathsf{TSP}(X)}\big] \leq 1 + O\big(\sqrt{\phi/\log n}\big)$*.*

6 Euclidean Steiner Trees

Kalpakis and Sherman [14] proposed a partitioning algorithm for the Euclidean minimum Steiner tree problem analogous to Karp's partitioning scheme for Euclidean TSP. The worst case cost of the Steiner tree computed by the algorithm, however, could be larger than optimal by a constant factor. Let $\mathsf{KS}(X)$ denote the cost of the Steiner tree computed.

The running-time of this algorithm is polynomial for the choice of $s = n/\log n$ [6]. For the same reason as for Karp's partitioning scheme, we cannot use our framework to estimate the running-time, because the choice of cells depends on the actual point set.

As for the traveling salesman problem, we have a worst-case approximation ratio of $\alpha(n) = O(n)$. The reason is that, for any two points $x, y \in X$, we have $\|x - y\| \leq \mathsf{ST}(X)$. Since Kalpakis and Sherman's partitioning algorithm outputs a tree with at most a linear number of edges, we have $\mathsf{KS}(X) \leq O\big(n \cdot \mathsf{ST}(X)\big)$. This gives us a worst-case approximation ratio of $O(n)$ and yields the following corollary of Theorem 3.5.

Corollary 6.1. *Under Assumption 2.4 as well as Assumption 3.2* (3)*, the expected approximation ratio of* KS *is* $\mathbb{E}\big[\frac{\mathsf{KS}(X)}{\mathsf{ST}(X)}\big] \leq 1 + O\big(\sqrt{\phi/\log n}\big)$.

7 Degree-Bounded Minimum Spanning Tree

A b-degree-bounded minimum spanning tree of a given set of points in $[0,1]^2$ is a spanning tree in which the degree of every point is bounded by b. For $2 \leq b \leq 4$, this problem is NP-hard, and it is solvable in polynomial time for $b \geq 5$ [19]. Let dbMST denote the Euclidean functional that maps a point set to the length of its shortest b-degree-bounded minimum spanning tree. This is a smooth, subadditive, and near-additive Euclidean functional [25].

Naturally, near-additivity implies that Karp's partitioning scheme can be extended to the b-degree-bounded minimum spanning tree problem. Let $\mathsf{P\text{-}bMST}$ be the adaptation of Karp's partitioning algorithm to dbMST with parameter $k^2 = \frac{n \log\log n}{\log n}$. With this choice of k, $\mathsf{P\text{-}bMST}$ runs in polynomial-time as a degree-bounded minimum-length spanning tree on m nodes can be found in time $2^{O(m \log m)}$ using brute-force search.

Again, we cannot use our framework for the running-time. The running-time is guaranteed to be bounded by a polynomial. But we can use Theorem 3.5 to obtain the following result.

Corollary 7.1. *Under Assumption 2.4 as well as Assumption 3.2* (3)*, the expected approximation ratio is* $\mathbb{E}\big[\frac{\mathsf{P\text{-}bMST}(X)}{\mathsf{dbMST}(X)}\big] \leq 1 + O\big(\sqrt{\phi \log\log n/\log n}\big)$.

8 Concluding Remarks

We have provided a smoothed analysis of partitioning algorithms for Euclidean optimization problems. The results can be extended to distributions over $\mathbb{R}^2$ by

scaling down the instance so that the inputs lie inside $[0,1]^2$. The analysis can also be extended to higher dimensions. However, the value of ϕ for which our results are applicable will depend on the dimension d.

Even though solutions computed by most of the partitioning algorithms achieve convergence to the corresponding optimal value with probability 1 under uniform samples, in practice they have constant approximation ratios close to 1 [13, 20]. Our results show that the expected function values computed by partitioning algorithms approach optimality not only under uniform, identical distributions, but also under non-uniform, non-identical distributions, provided that the distributions are not sharply concentrated.

One prominent open problem for which our approach does not work is the functional defined by the total edge weight of a minimum-weight triangulation in the Euclidean plane. The two main obstacles for this problem are that, first, the functional corresponding to minimum-weight triangulation is not smooth and, second, the value computed by the partitioning heuristic depends on the number of points in the convex hull of the point set [12]. Damerow and Sohler [5] provide a bound for the smoothed number of points in the convex hull, but this bound is not strong enough for this purpose.

References

1. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows.* Prentice-Hall, 1993.
2. Birgit Anthes and Ludger Rüschendorf. On the weighted Euclidean matching problem in R^d dimensions. *Applicationes Mathematicae*, 28(2):181–190, 2001.
3. David Arthur, Bodo Manthey, and Heiko Röglin. k-means has polynomial smoothed complexity. In *Proc. 50th Ann. IEEE Symp. on Foundations of Computer Science (FOCS)*, pp. 405–414. IEEE, 2009.
4. René Beier and Berthold Vöcking. Random knapsack in expected polynomial time. *J. Comput. System Sci.*, 69(3):306–329, 2004.
5. Valentina Damerow and Christian Sohler. Extreme points under random noise. In *Proc. 12th Ann. European Symp. on Algorithms (ESA)*, *LNCS* 3221, pp. 264–274. Springer, 2004.
6. S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1(3):195–207, 1971.
7. Martin E. Dyer and Alan M. Frieze. A partitioning algorithm for minimum weighted euclidean matching. *Inform. Process. Lett.*, 18(2):59–62, 1984.
8. Christian Engels and Bodo Manthey. Average-case approximation ratio of the 2-opt algorithm for the TSP. *Oper. Res. Lett.*, 37(2):83–84, 2009.
9. Matthias Englert, Heiko Röglin, and Berthold Vöcking. Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP. In *Proc. 18th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pp. 1295–1304. SIAM, 2007.
10. Alan M. Frieze and Joseph E. Yukich. Probabilistic analysis of the traveling salesman problem. In G. Gutin and A. P. Punnen, editors, *The Traveling Salesman Problem and Its Variations*, chapter 7, pp. 257–308. Kluwer, 2002.
11. Michael R. Garey, R. L. Graham, and David S. Johnson. The complexity of computing Steiner minimal trees. *SIAM J. Appl. Math.*, 32(4):835–859, 1977.

12. Mordecai J. Golin. Limit theorems for minimum-weight triangulations, other euclidean functionals, and probabilistic recurrence relations. In *Proc. 7th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pp. 252–260. SIAM, 1996.
13. David S. Johnson and Lyle A. McGeoch. Experimental analysis of heuristics for the STSP. In G. Gutin and A. P. Punnen, editors, *The Traveling Salesman Problem and Its Variations*, chapter 9, pp. 369–443. Kluwer, 2002.
14. Konstantinos Kalpakis and Alan T. Sherman. Probabilistic analysis of an enhanced partitioning algorithm for the Steiner tree problem in R^d. *Networks*, 24(3):147–159, 1994.
15. Richard M. Karp. Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane. *Math. Oper. Res.*, 2(3):209–224, 1977.
16. Carlos A. León and François Perron. Extremal properties of sums of Bernoulli random variables. *Statist. Probab. Lett.*, 62(4):345–354, 2003.
17. Michael Mitzenmacher and Eli Upfal. *Probability and Computing*. Cambridge University Press, 2005.
18. Christos H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theoret. Comput. Sci.*, 4(3):237–244, 1977.
19. Christos H. Papadimitriou and Umesh V. Vazirani. On two geometric problems related to the traveling salesman problem. *J. Algorithms*, 5(2):231–246, 1984.
20. Sivakumar Ravada and Alan T. Sherman. Experimental evaluation of a partitioning algorithm for the steiner tree problem in R^2 and R^3. *Networks*, 24(8):409–415, 1994.
21. Wansoo T. Rhee. A matching problem and subadditive euclidean functionals. *Ann. Appl. Probab.*, 3(3):794–801, 1993.
22. Heiko Röglin and Shang-Hua Teng. Smoothed analysis of multiobjective optimization. In *Proc. 50th Ann. IEEE Symp. on Foundations of Computer Science (FOCS)*, pp. 681–690. IEEE, 2009.
23. Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, 51(3):385–463, 2004.
24. Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis: An attempt to explain the behavior of algorithms in practice. *Comm. ACM*, 52(10):76–84, 2009.
25. Anand Srivastav and Sören Werth. Probabilistic analysis of the degree bounded minimum spanning tree problem. In *Proc. 27th Int. Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, *LNCS* 4855, pp. 497–507. Springer, 2007.
26. J. Michael Steele. Complete convergence of short paths in Karp's algorithm for the TSP. *Math. Oper. Res.*, 6:374–378, 1981.
27. J. Michael Steele. Subadditive Euclidean functionals and nonlinear growth in geometric probability. *Ann. Probab.*, 9(3):365–376, 1981.
28. J. Michael Steele. *Probability Theory and Combinatorial Optimization*, vol.$\tilde{6}$9 of *CBMS-NSF Regional Conf. Series in Appl. Math.*. SIAM, 1987.
29. Kenneth J. Supowit and Edward M. Reingold. Divide and conquer heuristics for minimum weighted euclidean matching. *SIAM J. Comput.*, 12(1):118–143, 1983.
30. Kasturi R. Varadarajan. A divide-and-conquer algorithm for min-cost perfect matching in the plane. In *Proc. 39th Ann. Symp. on Foundations of Computer Science (FOCS)*, pp. 320–331. IEEE, 1998.
31. Joseph E. Yukich. *Probability Theory of Classical Euclidean Optimization Problems*, vol. 1675 of *Lecture Notes in Mathematics*. Springer, 1998.