Smoothed Analysis of Partitioning Algorithms for Euclidean Functionals*

Markus Bläser[1] Bodo Manthey[2] 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

Euclidean optimization problems such as TSP and minimum-length matching admit fast partitioning algorithms that compute near-optimal solutions on typical instances.

In order to explain this performance, 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 [22] or the Euclidean Steiner tree problem [14], are NP-hard. For others, like minimum-length perfect matching, there exist polynomial-time algorithms. However, 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 compute quickly an optimal 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.

Although this is a rather simple ad-hoc approach, it works surprisingly well and fast in practice [16, 24]. This is at stark contrast to the worst-case performance of partitioning algorithms:

*A preliminary version has been presented at the 12th Algorithms and Data Structures Symposium (WADS 2011) [5]. Supported by DFG grant BL 511/7-1.

They can both be very slow and output solutions that are far from being optimal. Thus, as it is often the case, worst-case analysis is too pessimistic to explain the performance of partitioning algorithms. The reason for this is that worst-case analysis is dominated by artificially constructed 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 [31] proved complete convergence of Karp's partitioning algorithm [18] for Euclidean TSP. Also strong central limit theorems for a wide range of optimization problems are known. We refer to Steele [32] and Yukich [35] 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 [27] 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, 6, 12, 26]. We refer to two recent surveys [20, 28] for a broader picture.

We develop a general framework for smoothed analysis of partitioning algorithms for optimization problems in the Euclidean plane (Section 3). We consider a very general probabilistic model where the adversary specifies n density functions $f_1, \ldots, f_n : [0,1]^2 \to [0, \phi]$, one for each point. Then the actual point set is obtained by drawing x_i independently from the others according to f_i. 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 [25]. Let us note that, even in the average case, convergence to the optimal value for large n does not imply a bound on the expected approximation ratio. The reason is that if we compute a very bad solution with very small probability, then this allows convergence results but it deteriorates the expected approximation ratio.

We apply the general framework to obtain smoothed analyses of partitioning algorithms for Euclidean matching (Section 4), Karp's partitioning scheme for the TSP (Section 5), Steiner trees (Section 6), and degree-bounded minimum spanning trees (Section 7) in the Euclidean plane. Table 1 shows an overview. To summarize, for $\phi \leq \log^{O(1)} n$, Dyer and Frieze's partitioning algorithm for computing matchings [10] 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 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, but we focus on the aforementioned problems in this work.

problem	running-time	approximation ratio	reference
matching [10]	$O(n\phi^2 \log^4 n)$	$1 + O(\sqrt{\phi}/\log n)$	Corollaries 4.2 & 4.5
TSP [18]	$\text{poly}(n)$	$1 + O(\sqrt{\phi/\log n})$	Corollary 5.2
Steiner tree [17]	$\text{poly}(n)$	$1 + O(\sqrt{\phi/\log n})$	Corollary 6.2
degree-bounded MST	$\text{poly}(n)$	$1 + O(\sqrt{\phi \log\log n/\log n})$	Corollary 7.2

Table 1: Smoothed bounds for some Euclidean optimization problems.

2 Preliminaries

For $n \in \mathbb{N}$, let $[n] = \{1, 2, \ldots, n\}$. We denote probabilities by $\mathbb{P}$ and expected values by $\mathbb{E}$.

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 optimal traveling salesman tour.
- MST maps a point set 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 optimal 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 [13, 35]. A Euclidean functional F is called *smooth* [25, 35] if there is a constant c such that

$$\left|\mathsf{F}(X \cup Y) - \mathsf{F}(X)\right| \leq c\sqrt{|Y|}$$

for all finite $X, Y \subseteq [0,1]^2$. The constant c may depend on the function F, but not on the sets X and Y or their cardinality.

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

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

for all finite $X \subseteq [0,1]^2$ and all partitioning of the square. F is called *super-additive* if

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

for all finite $X \subseteq [0,1]^2$ and all partitioning of the square. 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 [13], we define a slightly weaker additivity condition that is sufficient for the performance analysis of partitioning algorithms. Frieze and Yukich [13] 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

$$\left| \mathsf{F}(X) - \sum_{\ell=1}^{s} \mathsf{F}(X_\ell) \right| = O\left(\sum_{\ell=1}^{s} \text{diameter}(C_\ell) \right).$$

If F is sub-additive and super-additive, then F is also near-additive.

Unfortunately, the Euclidean functionals TSP, MM and MST are smooth and sub-additive but not super-additive [31, 32, 35]. However, these functionals can be approximated by their corresponding canonical *boundary functionals*, which are super-additive [13,35]. We obtain the canonical boundary functional of a Euclidean functional by considering the boundary of the domain as a single point [35]. This means that two points can either be connected directly or via a detour along the boundary. In the latter case, only the lengths of the two edges connecting the two points to the boundary count, walking along the boundary is free of charge. Yukich [35] has shown that this is a sufficient condition for a Euclidean functional to be near-additive.

Proposition 2.1 (Yukich [35, 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 powerful tool for the analysis of Euclidean functionals. Rhee [25] 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 [25])**.** *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*

$$\mathbb{P}\left[\bigl| \mathsf{F}(X) - \mathbb{E}[\mathsf{F}(X)] \bigr| > t \right] \leq c' \cdot \exp\left(-\frac{ct^4}{n} \right).$$

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 [25], 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 [27], 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 vectors 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 the objective value of an optimal solution, i.e., $\mathsf{F}(X)$, 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}.*

The cells in the first step of Algorithm 3.1 are rectangles. They are not necessarily of the same size (in this paper, only the algorithm for matching divides the unique square into cells of exactly the same size, the other algorithms choose the division into squares depending on the actual point set). 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 $s = O(n/\log^{O(1)} n)$ (for matching, we could also use smaller s while maintaining polynomial, albeit worse, running-time; for the other problems, we even need $s = O(n/\log^{O(1)}))$. Ignoring poly-logarithmic terms, the first and third assumption translate roughly to $\phi = O(n)$ and $\phi = o(\sqrt{n})$, respectively. The second assumption roughly says $\phi = \omega(1)$. But for $\phi = O(1)$, we can expect roughly average-case behavior because the adversary has only little influence on the positions of the points.

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 [10], 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 $\sum_{\ell=1}^{s} T(n_\ell) + O(s)$, where n_ℓ is the number of points in cell C_ℓ. The expected running-time of A is thus bounded by

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

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 [18] 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 slightly abusing notation, 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 $f_i(C_\ell) \leq \phi/s$ (this requires that the cells are of equal size, thus their area is $1/s$). 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 (1) 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.

The goal of the adversary is to cause the partitioning algorithm to be slow. We will show that, in order to do this, the adversary 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

$$\left(\left\lfloor\frac{s}{\phi}\right\rfloor + 1\right) \cdot \mathbb{E}\bigl[T(B)\bigr] + O(s).$$

Since $\phi = O(s)$ by Assumption 3.2 (1), this is bounded by $O\bigl(\frac{s}{\phi} \cdot \mathbb{E}[T(B)] + s\bigr)$. If T is bounded by a polynomial, then this evaluates to $O\bigl(\frac{s}{\phi} \cdot T(n\phi/s) + s\bigr)$ 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).

Theorem 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 $\mathbb{E}[T(B)] = \Theta(T(\mathbb{E}[B]))$.*

Proof. We have $\mathbb{E}[B] = np$. Jensen's inequality yields $\mathbb{E}[T(B)] \geq T(np)$. Thus, what remains to be proved is $\mathbb{E}[T(B)] = O(T(np))$. Chernoff's bound [21, Theorem 4.4] says

$$\mathbb{P}\bigl[B > 2np\bigr] \leq \left(\frac{e}{4}\right)^{np}.$$

This allows us to bound

$$\mathbb{E}\bigl[T(B)\bigr] \leq T(2np) + \left(\frac{e}{4}\right)^{np} \cdot T(n).$$

Since T is bounded by a polynomial, we have $T(2np) = O(T(np))$. Since $p \in \omega(\log n/n)$ and T is bounded by a polynomial, we have $(e/4)^{np} \cdot T(n) \in o(1)$. Thus, $\mathbb{E}[T(B)] = O(T(np))$, which proves the lemma. □ □

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. In the following series of three lemmas, we make the argument rigorous.

The first lemma basically says that we maximize a convex function of a sum of independent 0/1 random variables if we balance the probabilities of the random variables. This is similar to a result by León and Perron [19]. But when we apply Lemma 3.4 in the proof of Lemma 3.5, we have to deal with the additional constraint $p_i \in [\varepsilon_i, 1 - \varepsilon_i]$. This makes León and Perron's result [19] inapplicable.

Theorem 3.4. *Let $p \in (0,1)$. Let X_1, X_2 be independent 0/1 random variables with $\mathbb{P}[X_1 = 1] = p - \delta$ and $\mathbb{P}[X_2 = 1] = p + \delta$. Let $X = X_1 + X_2$. Let f be any convex function, and let $g(\delta) = \mathbb{E}[f(X)]$.*

Then g is monotonically decreasing in δ for $\delta > 0$ and monotonically increasing for $\delta < 0$ and has a global maximum at $\delta = 0$.

Proof. A short calculation shows that

$$\mathbb{E}\bigl[f(X)\bigr] = (1 - 2p + p^2 - \delta^2) \cdot f(0) + (2p - 2p^2 + 2\delta^2) \cdot f(1) + (p^2 - \delta^2) \cdot f(2).$$

Abbreviating all terms that do not involve δ by z yields

$$g(\delta) = z + \bigl(-\delta^2 f(0) + 2\delta^2 f(1) - \delta^2 f(2)\bigr).$$

The lemma follows now by the convexity of f. □ □

With Lemma 3.4 above, we can show the following lemma: If we maximize a convex function of n 0/1 random variables and this function is symmetric around $n/2$, then we should make all probabilities as small as possible (or all as large as possible) in order to maximize the function.

Theorem 3.5. *Let f be an arbitrary convex function. Let $X_1, X_2, \ldots, X_n$ be independent* 0/1 *random variables with $\mathbb{P}[X_i = 1] = p_i \in [\varepsilon_i, 1 - \varepsilon_i]$, and let $X = \sum_{i=1}^{n} X_i$. Let $g(p_1, \ldots, p_n) = \mathbb{E}[f(X) + f(n - X)]$. Then g has a global maximum at $(\varepsilon_1, \ldots, \varepsilon_n)$.*

Proof. In the following, let $X' = \sum_{i=1}^{n-1} X_i$. Without loss of generality, we can assume that $\sum_{i=1}^{n} p_i \leq n/2$. Otherwise, we replace p_i by $1 - p_i$, which does not change the function value of g by symmetry.

First, we want to eliminate p_i with $p_i > 1/2$. If there is a $p_i > 1/2$, then there must be a $p_{i'} < 1/2$ since $\sum_{i=1}^{n} p_i \leq n/2$. Let $i = n$ and $i' = n - 1$ without loss of generality. Our goal is to shift "probability mass" from X_n to X_{n-1}. To do this, let $q = (p_{n-1} + p_n)/2$. We consider two new functions $\tilde{g}$ and h. The function $\tilde{g}$ is defined by

$$\tilde{g}(X_{n-1}, X_n) = \mathbb{E}_{X_1,\ldots,X_{n-2}} \left[f\left(\sum_{i=1}^{n} X_i \right) \right],$$

where the expected value is taken only over $X_1, \ldots, X_{n-2}$. The function h is defined by

$$h(\delta) = g(p_1, \ldots, p_{n-2}, q - \delta, q + \delta) = \mathbb{E}_{X_{n-1},X_n} \big[\tilde{g}(X_{n+1}, X_n) \big].$$

By definition, we have $h\big(\frac{p_n - p_{n-1}}{2}\big) = g(p_1, \ldots, p_n)$. The function h is convex and we can apply Lemma 3.4: We should choose $|\delta|$ as small as possible in order to maximize it. We decrease δ from $(p_n - p_{n-1})/2 > 0$ until $q - \delta$ or $q + \delta$ becomes $1/2$. Then we set p_{n-1} and p_n accordingly. In this way, we guarantee that $p_{n-1} \in [\varepsilon_{n-1}, 1 - \varepsilon_{n-1}]$ and $p_n \in [\varepsilon_n, 1 - \varepsilon_n]$. We iterate this process until we have $p_i \leq 1/2$ for all $i \in [n]$. This only increases F.

Now we can assume that $p_1, \ldots, p_n \leq 1/2$. We finish the proof by showing that decreasing any p_i as much as possible only increases $g(p_1, \ldots, p_n)$. Let $\Delta(x) = f(x + 1) - f(x)$. Since f is convex, Δ is non-decreasing. By symmetry, it suffices to consider p_n. We have

$$\begin{aligned}
g(p_1, \ldots, p_n) = {} & \quad p_n \cdot \mathbb{E}\big[f(X' + 1) + f(n - X' - 1)\big] \\
& + (1 - p_n) \cdot \mathbb{E}\big[f(X') + f(n - X')\big] \\
= {} & \quad p_n \cdot \mathbb{E}\big[f(X') + \Delta(X') + f(n - X' - 1)\big] \\
& + (1 - p_n) \cdot \mathbb{E}\big[f(X') + f(n - X' - 1) + \Delta(n - X' - 1)\big] \\
= {} & \quad \mathbb{E}\big[f(X') + f(n - X' - 1)\big] \\
& + \mathbb{E}\big[p_n \cdot \Delta(X') + (1 - p_n) \cdot \Delta(n - X' - 1)\big] \\
= {} & \quad \mathbb{E}\big[f(X') + f(n - X' - 1)\big] \\
& + p_n \cdot \mathbb{E}\big[\Delta(X')\big] + (1 - p_n) \cdot \mathbb{E}\big[\Delta(n - X' - 1)\big].
\end{aligned}$$

Only the term in the last line depends on p_n. Since $p_i \leq 1/2$ for all $i \in [n - 1]$, X' is stochastically dominated by $n - X' - 1$. Since Δ is non-decreasing, this yields

$$\mathbb{E}\big[\Delta(n - X' - 1)\big] \geq \mathbb{E}\big[\Delta(X')\big].$$

Hence, decreasing p_n will never decrease the value of g. □ □

Lemma 3.5 above is the main ingredient for the proof that the adversary wants as many full cells as possible. Lemma 3.6 below makes this rigorous.

Theorem 3.6. *Let $C_{\ell'}$ and $C_{\ell''}$ be any two cells. Let $f_1, \ldots, f_n : [0,1]^2 \to [0, \phi]$ be any density functions. Let $\tilde{f}_1, \ldots, \tilde{f}_n : [0,1]^2 \to [0, \phi]$ be density functions with the following properties for all $i \in [n]$:*

1. $\tilde{f}_i(C_{\ell'}) = \min\bigl(\phi/s, f_i(C_{\ell'}) + f_i(C_{\ell''})\bigr)$.
2. $\tilde{f}_i(C_{\ell''}) = \bigl(f_i(C_{\ell'}) + f_i(C_{\ell''})\bigr) - \tilde{f}_i(C_{\ell'})$.

(Note that there are densities $\tilde{f}_1, \ldots, \tilde{f}_n$ with these properties: First, all $\tilde{f}_i$ are non-negative and, second, $\int_{[0,1]^2} \tilde{f}_i(x)\, \mathrm{d}x = 1$. Furthermore, $\tilde{f}_1, \ldots, \tilde{f}_n$ can be chosen such that they are bounded by ϕ since we have $f_i(C_{\ell'}), f_i(C_{\ell''}) \leq \phi/s$ by construction.) Let n_ℓ be the (random) number of points in X_ℓ with respect to $f = (f_1, \ldots, f_n)$, and let $\tilde{n}_\ell$ be the (random) number of points in X_ℓ with respect to $\tilde{f} = (\tilde{f}_1, \ldots, \tilde{f}_n)$. Then

$$\sum_{\ell=1}^{s} \mathbb{E}\bigl[T(n_\ell)\bigr] \leq \sum_{\ell=1}^{s} \mathbb{E}\bigl[T(\tilde{n}_\ell)\bigr].$$

Proof. First, we note that $\mathbb{E}[T(n_\ell)] = \mathbb{E}[T(\tilde{n}_\ell)]$ for $\ell \neq \ell', \ell''$. Without loss of generality, let $\ell' = 1$ and $\ell'' = 2$. Thus, we have to prove

$$\mathbb{E}\bigl[T(n_1)\bigr] + \mathbb{E}\bigl[T(n_2)\bigr] \leq \mathbb{E}\bigl[T(\tilde{n}_1)\bigr] + \mathbb{E}\bigl[T(\tilde{n}_2)\bigr].$$

Let $M = \{i \mid x_i \in C_1 \cup C_2\}$ be the (random) set of indices of points in the two cells. To prove this, we prove the inequality

$$\mathbb{E}\bigl[T(n_1) + T(n_2) \mid M = I\bigr] \leq \mathbb{E}\bigl[T(\tilde{n}_1) + T(\tilde{n}_2) \mid M = I\bigr]$$

for any set $I \subseteq [n]$. This is equivalent to

$$\mathbb{E}\bigl[T(n_1) + T(|M| - n_1) \mid M = I\bigr] \leq \mathbb{E}\bigl[T(\tilde{n}_1) + T(|M| - \tilde{n}_1) \mid M = I\bigr].$$

Without loss of generality, we restrict ourselves to the case $I = [n]$. This gives us the following setting: Any point x_i is either in C_1 or in C_2. Under this condition, the probability that x_i is in C_1 is $p_i = \frac{f_i(C_1)}{f_i(C_1 \cup C_2)}$, and the probability that x_i is in C_2 is $1 - p_i = \frac{f_i(C_2)}{f_i(C_1 \cup C_2)}$. We can choose p_i arbitrarily such that $p_i \leq \min\bigl\{1, \frac{\phi/s}{f_i(C_1)+f_i(C_2)}\bigr\} = 1 - \varepsilon_i$ and $p_i \geq \max\bigl\{0, 1 - \frac{\phi/s}{f_i(C_1)+f_i(C_2)}\bigr\} = \varepsilon_i$. This is precisely the setting that we need to apply Lemma 3.5. □ □

Let $f_1, \ldots, f_n : [0,1]^2 \to [0, \phi]$ be the given distributions. By applying Lemma 3.6 repeatedly for pairs of non-full, non-empty cells $C_{\ell'}$ and $C_{\ell''}$, we obtain distributions $\tilde{f}_1, \ldots, \tilde{f}_n$ with the following properties:

1. $\tilde{f}_1, \ldots, \tilde{f}_n$ have $\lfloor s/\phi \rfloor$ full cells and at most one cell that is neither full nor empty.
2. The expected value of T on X sampled according to $\tilde{f}_1, \ldots \tilde{f}_n$ is not smaller than the expected value of T on X sampled according to $f_1, \ldots, f_n$.

This shows that the adversary, in order to slow down our algorithm, will concentrate the probability in as few cells as possible. Thus, we obtain the following theorem.

Theorem 3.7. *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 Assumptions 2.4, 3.2* (1)*, and 3.2* (2)*, the expected running-time of* A *on input* X *is bounded from above by*

$$O\left(\frac{s}{\phi} \cdot T\left(\frac{n\phi}{s}\right) + s\right).$$

Proof. The expected running-time is maximized if we have $\lfloor s/\phi \rfloor$ cells that are full plus possibly one cell containing all the remaining probability mass. The expected running-time for each such cell is $O(T(n\phi/s))$ by Lemma 3.3 and because of Assumption 3.2 (2). Thus, the expected running-time of A is bounded from above by

$$\lceil \frac{s}{\phi} \rceil \cdot O\left(T\left(\frac{n\phi}{s}\right)\right) + O(s).$$

The theorem follows as $\phi = O(s)$ by Assumption 3.2 (1). □ □

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 a near-additive Euclidean functional, we have $\mathsf{A}(X) \leq \mathsf{F}(X) + J$ for

$$J = J' + O\left(\sum_{\ell=1}^{s} \text{diameter}(C_\ell)\right).$$

Dividing by $\mathsf{F}(X)$ yields

$$\frac{\mathsf{A}(X)}{\mathsf{F}(X)} \leq 1 + O\left(\frac{J}{\mathsf{F}(X)}\right). \tag{2}$$

Together with $\mathbb{E}[\mathsf{F}(X)] \geq \mu_{\mathsf{F}}(n, \phi)$, we obtain a generic upper bound of

$$\frac{\mathbb{E}[\mathsf{A}(X)]}{\mathbb{E}[\mathsf{F}(X)]} \leq 1 + O\left(\frac{J}{\mu_{\mathsf{F}}(n, \phi)}\right)$$

for the ratio of expected output of A and expected function value of F. While this provides some guarantee on the approximation performance, it does not provide a bound on the expected approximation ratio, which is in fact our goal.

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 have a good (deterministic) upper bound for $\mathsf{A}(X)$ that we can plug into the expected ratio in order to get an upper bound for the ratio that only depends on $\mathsf{F}(X)$, we are basically left with the problem of estimating $\mathbb{E}[1/\mathsf{F}(X)]$. Jensen's inequality yields $\mathbb{E}[1/\mathsf{F}(X)] \geq 1/\mathbb{E}[\mathsf{F}(X)]$. 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 optimal objective value arises frequently in bounding expected approximation ratios [11, 12].

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 (2) 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 3.8. *Assume that* A *has a worst-case approximation ratio of* $\alpha(n)$ *for any instance consisting of* n *points. Then, under Assumption 2.4, the expected approximation ratio of* A *is*

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

for some positive constant $c > 0$.

Proof. We have

$$\frac{\mathsf{A}(X)}{\mathsf{F}(X)} \leq \min\left\{1 + O\left(\frac{J}{\mathsf{F}(X)}\right), \alpha(n)\right\}. \tag{3}$$

By Theorem 2.2 and Remark 2.3, we have

$$\mathbb{P}\left[\mathsf{F}(X) < \frac{\mu_{\mathsf{F}}(n,\phi)}{2}\right] \leq c' \exp\left(-\frac{c\mu_{\mathsf{F}}(n,\phi)^4}{n}\right)$$

for some constants $c, c' > 0$. Together with (3), this allows us to bound the expected approximation ratio as

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

which completes the proof. □ □

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 $\mathbb{E}[1/\mathsf{F}(X)]$. Note that we do not explicitly provide an upper bound for $\mathbb{E}[1/\mathsf{F}(X)]$, but only a sufficiently strong tail bound h_n for $1/\mathsf{F}(X)$.

Theorem 3.9. *Assume that there exists a* $\beta \leq J$ *and a function* h_n *such that* $\mathbb{P}[\mathsf{F}(X) \leq x] \leq h_n(x)$ *for all* $x \in [0, \beta]$. *Then, under Assumption 2.4, the expected approximation ratio of* A *is*

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

Proof. If $\mathsf{F}(X) \geq \mu_{\mathsf{F}}(n,\phi)/2$, then the approximation ratio is

$$1 + O\left(\frac{J}{\mu_{\mathsf{F}}(n,\phi)}\right),$$

which is good. By Theorem 2.2, the probability that this does not hold is bounded from above by $\exp\bigl(-\frac{\mu_{\mathsf{F}}(n,\phi)^4}{Cn}\bigr)$ for some constant $C > 0$. If we still have $\mathsf{F}(X) \geq \beta$, then we can bound the ratio from above by

$$1 + O\left(\frac{J}{\beta}\right).$$

This contributes

$$\exp\left(-\frac{\mu_{\mathsf{F}}(n,\phi)^4}{Cn}\right) \cdot \left(1 + O\left(\frac{J}{\beta}\right)\right) \leq \exp\left(-\frac{\mu_{\mathsf{F}}(n,\phi)^4}{Cn}\right) \cdot O\left(\frac{J}{\beta}\right)$$

to the expected value, where the inequality follows from $\beta \leq J$. We are left with the case that $\mathsf{F}(X) \leq \beta$. This case contributes

$$J \cdot \int_{1/\beta}^{\infty} \mathbb{P}\left[\frac{1}{\mathsf{F}(X)} \geq x\right] \mathrm{d}x.$$

to the expected value. By definition, we have

$$\mathbb{P}\left[\frac{1}{\mathsf{F}(X)} \geq x\right] = \mathbb{P}\left[\mathsf{F}(X) \leq \frac{1}{x}\right] \leq h_n\left(\frac{1}{x}\right),$$

which completes the proof. □ □

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 for approximating MM was proposed by Dyer and Frieze [10]. For completeness, let us describe their algorithm.

Algorithm 4.1 (DF; Dyer, Frieze [10])**.** *Input: set $X \subseteq [0,1]^2$ of n points, n is even.*

1. *Partition $[0,1]^2$ into $s = k^2$ equal-sized sub-squares $C_1, \ldots, C_{k^2}$, each of side length $1/k$, where $k = \frac{\sqrt{n}}{\log n}$.*

2. *Compute minimum-length perfect matchings for X_ℓ for each $\ell \in [k^2]$.*

3. *Compute a matching for the unmatched points from the previous step using the strip heuristic [33].*

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 the uniform distribution on $[0,1]^2$ (this corresponds to Assumption 2.4 with $\phi = 1$). 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 3.7, we get the following corollary.

Theorem 4.2. *Under Assumptions 2.4, 3.2* (1)*, and 3.2* (2)*, the expected running-time of* DF *on input X is at most*

$$O\left(\frac{n^3\phi^2}{k^4}+k^2\right).$$

If we plug in $k=\sqrt{n}/\log n$, we obtain an expected running-time of at most

$$O\bigl(n\phi^2\log^4 n\bigr).$$

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

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

We use NN to bound MM from below: First, we have $\mathsf{MM}(X)\geq\mathsf{NN}(X)/2$. Second, $\mathbb{E}\bigl[\mathsf{NN}(X)\bigr]$ is easier to analyze than $\mathbb{E}\bigl[\mathsf{MM}(X)\bigr]$. Thus, according to the following lemma, we can choose $\mu_{\mathsf{MM}}(n,\phi)=\Omega\bigl(\sqrt{n/\phi}\bigr)$.

Theorem 4.3. *Under Assumption 2.4, we have*

$$\mathbb{E}\bigl[\mathsf{NN}(X)\bigr]=\Omega\left(\sqrt{\frac{n}{\phi}}\right).$$

Proof. By linearity of expectation, we have $\mathbb{E}\bigl[\mathsf{NN}(X)\bigr]=n\cdot\mathbb{E}\bigl[\min_{i\geq2}\|x_1-x_i\|\bigr]$. Thus, we have to prove $\mathbb{E}\bigl[\min_{i\geq2}\|x_1-x_i\|\bigr]=\Omega\bigl(1/\sqrt{n\phi}\bigr)$. To bound this quantity from below, we assume that x_1 is fixed by an adversary and that only $x_2,\ldots,x_n$ are drawn independently according to their density functions. Then we obtain

$$\begin{aligned}\mathbb{E}\bigl[\min_{i\geq2}\|x_1-x_i\|\bigr]&=\int_0^\infty\mathbb{P}\left[\min_{i\geq2}\|x_1-x_i\|\geq r\right]\mathrm{d}r\\&=\int_0^\infty\prod_{i=2}^n\bigl(1-\mathbb{P}\bigl[\|x_1-x_i\|\leq r\bigr]\bigr)\,\mathrm{d}r\\&\geq\int_0^{1/\sqrt{\phi\pi n}}\prod_{i=2}^n\bigl(1-\mathbb{P}\bigl[\|x_1-x_i\|\leq r\bigr]\bigr)\,\mathrm{d}r.\end{aligned}$$

The probability that $\|x_1-x_i\|\leq r$ can be bounded from above by ϕ times the area of a circle of radius r, which is $\phi\pi r^2$. Thus,

$$\begin{aligned}\mathbb{E}\bigl[\min_{i\geq2}\|x_1-x_i\|\bigr]&\geq\int_0^{1/\sqrt{\phi\pi n}}(1-\phi\pi r^2)^{n-1}\,\mathrm{d}r\\&\geq\int_0^{1/\sqrt{\phi\pi n}}\left(1-\frac{1}{n}\right)^{n-1}\mathrm{d}r\geq\frac{1}{e\sqrt{\phi\pi n}}.\end{aligned}$$

The second inequality holds because $1-\phi\pi r^2 \geq 1-\frac{1}{n}$ for $r \in [0, 1/\sqrt{\phi\pi n}]$. The third inequality exploits $\left(1-\frac{1}{n}\right)^{n-1} \geq 1/e$. □ □

Since MM is near-additive and the diameter of each cell is $O(1/k)$, we can use

$$J = O\left(\sum_{\ell=1}^{k^2} \text{diameter}(C_\ell)\right) = O(k) = O\left(\frac{\sqrt{n}}{\log n}\right). \tag{4}$$

Unfortunately, we cannot bound the worst-case approximation ratio of Dyer and Frieze's partitioning algorithm. Thus, we cannot apply Theorem 3.8, but we have to use Theorem 3.9. Thus, we first need a tail bound for $1/\mathsf{MM}(X)$. The bound in the following lemma suffices for our purposes.

Theorem 4.4. *Under Assumption 2.4, we have*

$$\mathbb{P}\big[\mathsf{MM}(X) \leq c\big] \leq (2\phi c)^{n/2}$$

for all $c \leq \frac{1}{2\pi}$.

Proof. Let us first analyze the probability that a specific fixed matching M has a length of at most c. We let an adversary fix one end-point of each edge. Then the probability that a specific edge of M has a length of at most c is bounded from above by $\phi\pi c^2$. Thus, the density of the length of a particular edge is bounded from above by $2\phi\pi c \leq \phi$ as $c \leq \frac{1}{2\pi}$. Furthermore, the lengths of the edges of M are independent random variables. Thus, the probability that the sum of the edge lengths of all $n/2$ edges of M is bounded from above by c is at most $\frac{(\phi\pi c)^{n/2}}{(n/2)!}$, which can be proved by the following induction: Let $m = n/2$, and let $a_1, \ldots, a_m$ be the (random) edge lengths of the edge of M. For $m = 1$, the statement follows from $\mathbb{P}[a_1 \leq c] \leq \phi c$. For larger m, assume that the claim holds for $m-1$, and let h be the density of a_m. This density is bounded by ϕ as argued above. Thus,

$$\begin{aligned}\mathbb{P}\big[a_1 + \ldots + a_m \leq c\big] &\leq \int_0^c h(a_m)\,\mathbb{P}\big[a_1 + \ldots + a_{m-1} \leq c - a_m\big]\,\mathrm{d}a_m \\ &\leq \int_0^c \phi \cdot \frac{(\phi(c-a_m))^{m-1}}{(m-1)!}\,\mathrm{d}a_m = \frac{(\phi c)^m}{m!}.\end{aligned}$$

The number of perfect matchings of a complete graph on n vertices is $(n-1)!! = (n-1)\cdot(n-3)\cdot(n-5)\cdot\ldots$ ("!!" denotes the double factorial). A union bound over all matchings yields

$$\mathbb{P}\big[\mathsf{MM}(X) \leq c\big] \leq \frac{(n-1)!!\cdot(\phi c)^{n/2}}{(n/2)!} \leq \frac{n!!}{(n/2)!}\cdot(\phi c)^{n/2} = (2\phi c)^{n/2},$$

which completes the proof. □ □

With this tail bound for $1/\mathsf{MM}(X)$, we can prove the following bound on the smoothed approximation ratio.

Theorem 4.5. *Under Assumptions 2.4 and 3.2* (3), *the expected approximation ratio of* DF *is* $1 + O\big(\frac{\sqrt{\phi}}{\log n}\big)$.

Proof. We apply Theorem 3.9. To do this, let $\beta = \frac{1}{2\pi\phi}$ (this is exactly the value at which Lemma 4.4 becomes non-trivial). Lemma 4.4 allows us to choose $h_n(x) = (2\phi\pi x)^{n/2}$ and yields

$$\int_{1/\beta}^{\infty} h_n\left(\frac{1}{x}\right)\,\mathrm{d}x = \int_{1/\beta}^{\infty}\left(\frac{2\phi\pi}{x}\right)^{n/2}\mathrm{d}x = \frac{2(2\pi\phi)^{\frac{n}{2}}\beta^{\frac{n}{2}-1}}{n-2} = \frac{4\pi\phi}{(n-2)}.$$

Assumption 3.2 (3) with (4) yields

$$J \cdot \frac{4\pi\phi}{(n-2)} = O\left(\frac{\phi}{\sqrt{n}\cdot\log n}\right) = o\left(\frac{\sqrt{\phi}}{\log n}\right)$$

by Assumption 3.2 (3).

We can choose $\mu_{\mathsf{MM}}(n,\phi) = \Omega(\sqrt{n/\phi})$ as $\mathsf{MM}(X) \geq \mathsf{NN}(X)/2 = \Omega(\sqrt{n/\phi})$ by Lemma 4.3. Theorem 2.2 together with Assumption 3.2 (3) thus yields that the probability that $\mathsf{MM}(X) < \mu_{\mathsf{MM}}(n,\phi)/2$ is bounded from above by

$$\exp\left(-\frac{(\mu_{\mathsf{MM}}(n,\phi))^4}{Cn}\right) = \exp\left(-\Omega\left(\frac{n}{\phi^2}\right)\right) = \exp\big(-\omega(\log n)\big).$$

This bound decreases faster than any polynomial in n. Thus, also by Assumption 3.2 (3),

$$J \cdot \frac{\exp\big(-\frac{(\mu_{\mathsf{MM}}(n,\phi))^4}{Cn}\big)}{\beta} = O\left(\frac{\phi\sqrt{n}}{\log n}\cdot\exp\left(-\frac{(\mu_{\mathsf{MM}}(n,\phi))^4}{Cn}\right)\right)$$

decreases faster than any polynomial in n.

Altogether, Theorem 3.9 yields a bound of

$$1 + O\left(\frac{J}{\mu_{\mathsf{MM}}(n,\phi)}\right) + o\left(\frac{\sqrt{\phi}}{\log n}\right) = 1 + O\left(\frac{\sqrt{\phi}}{\log n}\right)$$

for the expected approximation ratio. □ □

Remark 4.6. *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 [34] for computing the optimal matchings within each cell. This algorithm has a running-time of $O(m^{1.5}\log^5 m)$ for m points, which improves the running-time bound to $O\big(n\sqrt{\phi}\log(n)\log^5(\phi\log n)\big)$.

5 Karp's Partitioning Scheme for Euclidean TSP

Karp's partitioning scheme [18] is a heuristic for Euclidean TSP that computes near-optimal solutions on average. It proceeds as follows:

Algorithm 5.1 (KP, Karp's partitioning scheme)**.** *Input: set $X \subseteq [0,1]^2$ of n points.*

1. Partition $[0,1]^2$ into $k = \sqrt{n/\log n}$ stripes such that each stripe contains exactly $n/k = \sqrt{n\log n}$ points.

2. Partition each stripe into k cells such that each cell contains exactly $n/k^2 = \log n$ points.

3. Compute optimal TSP tours for each cell.

4. Join the tours to obtain a TSP tour for X.

We remark that the choice of k in Karp's partitioning scheme is optimal in the following sense: On the one hand, more that $\Theta(\log n)$ points per cell would yield a super-polynomial running-time as the running-time is exponential in the number of points per cell. On the other hand, less than $\Theta(\log n)$ point per cell would yield a worse approximation ratio as the approximation ratio gets worse with increasing k.

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 [31] 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 $T(n) = 2^{n/k^2} \operatorname{poly}(n/k^2) + O(k^2)$, which is, independent of the randomness, polynomial in n for $k^2 = n/\log n$.

The nearest-neighbor functional NN is a lower bound for TSP. Thus, we can use Lemma 4.3 to obtain $\mu_{\mathsf{TSP}}(n,\phi) = \Omega(\sqrt{n/\phi})$. We can use the bound [18, 30]

$$\mathsf{KP}(X) \leq \mathsf{TSP}(X) + 6k = \mathsf{TSP}(X) + 6\sqrt{n/\log n}$$

to obtain $J = O(\sqrt{n/\log n})$.

The nice thing about the TSP is that every tour has a worst-case approximation guarantee: Consider any two points $x, y \in X$. Since any tour must visit both x and y, its length is at least $2\|x-y\|$ by the triangle inequality. Since a tour consists of n edges, any tour has a length of at most $\frac{n}{2} \cdot \mathsf{TSP}(X)$. Thus, we can use Theorem 3.8 together with $\alpha(n) = n/2$ and obtain the following result.

Theorem 5.2. *Under Assumptions 2.4 and 3.2* (3)*, the expected approximation ratio of* KP *is* $\mathbb{E}\big[\frac{\mathsf{KP}(X)}{\mathsf{TSP}(X)}\big] \leq 1 + O\big(\sqrt{\phi/\log n}\big)$.

Proof. We plug $J = O(\sqrt{n \log n})$ and $\mu_{\mathsf{TSP}}(n,\phi) = \Theta(\sqrt{n/\phi})$ and $\alpha(n) = n/2$ into the bound of Theorem 3.8 and obtain an upper bound of

$$1 + O\left(\sqrt{\frac{\phi}{\log n}}\right) + O\left(n \cdot \exp\left(-\Omega\left(\frac{n}{\phi^2}\right)\right)\right)$$

for the expected approximation ratio. By Assumption 3.2 (3), the exponential term decreases faster than any polynomial. Thus, $O(\sqrt{\phi/\log n})$ is an upper bound for the last term. □ □

6 Euclidean Steiner Trees

Kalpakis and Sherman [17] proposed a partitioning algorithm for the Euclidean minimum Steiner tree problem analogous to Karp's partitioning scheme for Euclidean TSP. The solution produced by their algorithm converges to the optimal value with probability $1 - o(1)$. Also, their algorithm [17] is known to produce near-optimal solutions in practice too [24]. Let us now describe Kalpakis and Sherman's algorithm [17].

Algorithm 6.1 (KS, Kalpakis, Sherman [17])**.** *Input: set $X \subseteq [0,1]^2$ of n points.*

1. *Let $s = n/\log n$. Partition $[0,1]^2$ into $\Theta(s)$ cells such that each cell contains at most $n/s = \log n$ points.*
2. *Solve the Steiner tree problem optimally within each cell.*
3. *Compute a minimum-length spanning tree to connect the forest thus obtained.*

The running-time of this algorithm is polynomial for the choice of $s = n/\log n$ [8]. 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.

Let $\mathsf{KS}(X)$ denote the cost of the Steiner tree computed Kalpakis and Sherman's algorithm [17]. For the analysis of the approximation performance, let $\mathsf{ST}(X)$ denote the cost of a minimum Steiner tree for the point set X, and let $\mathsf{MST}(X)$ denote the cost of a minimum-length spanning tree of X. Kalpakis and Sherman [17] have shown that

$$\mathsf{KS}(X) \leq \mathsf{ST}(X) + O\left(\sqrt{n/\log n}\right).$$

Thus, $J = O(\sqrt{n/\log n})$.

Since minimum spanning trees are $2/\sqrt{3}$ approximations for Euclidean Steiner trees [9], we have $\mathsf{ST}(X) \geq \frac{\sqrt{3}}{2} \cdot \mathsf{MST}(X)$. Furthermore, we have $\mathsf{MST}(X) \geq \frac{1}{2} \cdot \mathsf{NN}(X)$. Thus, we can choose $\mu_{\mathsf{ST}}(n, \phi) = \Theta\big(\sqrt{n/\phi}\big)$ by Lemma 4.3.

As KP for the traveling salesman problem, KS comes with 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 [17] outputs 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 3.8.

Theorem 6.2. *Under Assumptions 2.4 and 3.2* (3)*, the expected approximation ratio of* KS *is*

$$\mathbb{E}\left[\frac{\mathsf{KS}(X)}{\mathsf{ST}(X)}\right] \leq 1 + O\left(\sqrt{\frac{\phi}{\log n}}\right).$$

Proof. The proof is almost identical to the proof of Corollary 5.2. □ □

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 $b \geq 5$ [23]. Let dbMST denote the Euclidean functional that maps a point set to the length of its shortest b-degree-bounded minimum spanning tree.

Proposition 7.1. dbMST *is a smooth, sub-additive and near-additive Euclidean functional.*

Proof. The smoothness and sub-additivity have been proved by Srivastav and Werth [29]. They have also defined a canonical super-additive boundary functional that well-approximates dbMST [29, Lemmas 3 and 4]. This, together with Proposition 2.1 proves that dbMST is near-additive. □ □

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. Then, for any X, we have

$$\mathsf{P\text{-}bMST}(X) \leq \mathsf{dbMST}(X) + O\left(\sqrt{\frac{n \log \log n}{\log n}}\right),$$

which yields $J = O(\sqrt{n \log \log n / \log n})$.

Again, we have $\|x - y\| \leq \mathsf{dbMST}(X)$ for all X and $x, y \in X$, which implies that any possible tree is at most a factor n worse than the optimal tree. This implies in particular that the worst-case approximation ratio of $\mathsf{P\text{-}bMST}$ is $O(n)$: $\mathsf{P\text{-}bMST}(X) = O(n \cdot \mathsf{dbMST}(X))$. Furthermore, we can use $\mu_{\mathsf{dbMST}}(n, \phi) = \Omega(\sqrt{n/\phi})$ by Lemma 4.3 as $\mathsf{dbMST}(X) = \Omega(\mathsf{NN}(X))$.

We can apply Theorem 3.8 to obtain the following result.

Theorem 7.2. *Under Assumptions 2.4 and 3.2* (3)*, the expected approximation ratio is*

$$\mathbb{E}\left[\frac{\mathsf{P\text{-}bMST}(X)}{\mathsf{dbMST}(X)}\right] \leq 1 + O\left(\sqrt{\frac{\phi \log \log n}{\log n}}\right).$$

Proof. The proof is almost identical to the proof of Corollary 5.2. The only difference is we now have to use $J = O(\sqrt{n \log \log n / \log n})$, which leads to the slightly worse bound for the approximation ratio. □ □

Again, we cannot use our framework for the running-time, but the running-time is guaranteed to be bounded by a polynomial.

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 [16,24]. 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 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 [15]. Damerow and Sohler [7] provide a bound for the smoothed number of points in the convex hull. However, their bound is not strong enough for analyzing triangulations.

References

[1] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, 1993.

[2] Birgit Anthes and Ludger Rüschendorf. On the weighted Euclidean matching problem in $\mathbb{R}^d$. *Applicationes Mathematicae*, 28(2):181–190, 2001.

[3] David Arthur, Bodo Manthey, and Heiko Röglin. Smoothed analysis of the k-means method. *Journal of the ACM*, 58(5), 2011.

[4] René Beier and Berthold Vöcking. Random knapsack in expected polynomial time. *Journal of Computer and System Sciences*, 69(3):306–329, 2004.

[5] Markus Bläser, Bodo Manthey, and B. V. Raghavendra Rao. Smoothed analysis of partitioning algorithms for Euclidean functionals. In Frank Dehne, John Iacono, and Jörg-Rüdiger Sack, editors, *Proc. of the 12th Algorithms and Data Structures Symposium (WADS)*, volume 6844 of *Lecture Notes in Computer Science*, pages 110–121. Springer, 2011.

[6] Valentina Damerow, Bodo Manthey, Friedhelm Meyer auf der Heide, Harald Räcke, Christian Scheideler, Christian Sohler, and Till Tantau. Smoothed analysis of left-to-right maxima with applications. *ACM Transactions on Algorithms*, to appear.

[7] Valentina Damerow and Christian Sohler. Extreme points under random noise. In Susanne Albers and Tomasz Radzik, editors, *Proc. of the 12th Ann. European Symp. on Algorithms (ESA)*, volume 3221 of *Lecture Notes in Computer Science*, pages 264–274. Springer, 2004.

[8] S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1(3):195–207, 1971.

[9] Ding-Zhu Du and Frank K. Hwang. A proof of the Gilbert-Pollak conjecture on the Steiner ratio. *Algorithmica*, 7(2&3):121–135, 1992.

[10] Martin E. Dyer and Alan M. Frieze. A partitioning algorithm for minimum weighted euclidean matching. *Information Processing Letters*, 18(2):59–62, 1984.

[11] Christian Engels and Bodo Manthey. Average-case approximation ratio of the 2-opt algorithm for the TSP. *Operations Research Letters*, 37(2):83–84, 2009.

[12] Matthias Englert, Heiko Röglin, and Berthold Vöcking. Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP. In *Proc. of the 18th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1295–1304. SIAM, 2007.

[13] Alan M. Frieze and Joseph E. Yukich. Probabilistic analysis of the traveling salesman problem. In Gregory Gutin and Abraham P. Punnen, editors, *The Traveling Salesman Problem and Its Variations*, chapter 7, pages 257–308. Kluwer Academic Publishers, 2002.

[14] Michael R. Garey, R. L. Graham, and David S. Johnson. The complexity of computing Steiner minimal trees. *SIAM Journal of Applied Mathematics*, 32(4):835–859, 1977.

[15] Mordecai J. Golin. Limit theorems for minimum-weight triangulations, other euclidean functionals, and probabilistic recurrence relations. In *Proc. of the 7th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 252–260. SIAM, 1996.

[16] David S. Johnson and Lyle A. McGeoch. Experimental analysis of heuristics for the STSP. In Gregory Gutin and Abraham P. Punnen, editors, *The Traveling Salesman Problem and Its Variations*, chapter 9, pages 369–443. Kluwer Academic Publishers, 2002.

[17] 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.

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

[19] Carlos A. León and François Perron. Extremal properties of sums of Bernoulli random variables. *Statistics & Probability Letters*, 62(4):345–354, 2003.

[20] Bodo Manthey and Heiko Röglin. Smoothed analysis: Analysis of algorithms beyond worst case. *it – Information Technology*, 53(6):280–286, 2011.

[21] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis.* Cambridge University Press, 2005.

[22] Christos H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theoretical Computer Science*, 4(3):237–244, 1977.

[23] Christos H. Papadimitriou and Umesh V. Vazirani. On two geometric problems related to the traveling salesman problem. *Journal of Algorithms*, 5(2):231–246, 1984.

[24] 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.

[25] Wansoo T. Rhee. A matching problem and subadditive euclidean functionals. *Annals of Applied Probability*, 3(3):794–801, 1993.

[26] Heiko Röglin and Shang-Hua Teng. Smoothed analysis of multiobjective optimization. In *Proc. of the 50th Ann. IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 681–690. IEEE, 2009.

[27] Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463, 2004.

[28] Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis: An attempt to explain the behavior of algorithms in practice. *Communications of the ACM*, 52(10):76–84, 2009.

[29] Anand Srivastav and Sören Werth. Probabilistic analysis of the degree bounded minimum spanning tree problem. In Vikraman Arvind and Sanjiva Prasad, editors, *Proc. of the 27th Int. Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 4855 of *Lecture Notes in Computer Science*, pages 497–507. Springer, 2007.

[30] J. Michael Steele. Complete convergence of short paths in Karp's algorithm for the TSP. *Mathematics of Operations Research*, 6:374–378, 1981.

[31] J. Michael Steele. Subadditive Euclidean functionals and nonlinear growth in geometric probability. *Annals of Probability*, 9(3):365–376, 1981.

[32] J. Michael Steele. *Probability Theory and Combinatorial Optimization*, volume 69 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, 1987.

[33] Kenneth J. Supowit and Edward M. Reingold. Divide and conquer heuristics for minimum weighted euclidean matching. *SIAM Journal on Computing*, 12(1):118–143, 1983.

[34] Kasturi R. Varadarajan. A divide-and-conquer algorithm for min-cost perfect matching in the plane. In *Proc. of the 39th Ann. Symp. on Foundations of Computer Science (FOCS)*, pages 320–331. IEEE, 1998.

[35] Joseph E. Yukich. *Probability Theory of Classical Euclidean Optimization Problems*, volume 1675 of *Lecture Notes in Mathematics*. Springer, 1998.