2013 IEEE 54th Annual Symposium on Foundations of Computer Science

Approximation Schemes for Maximum Weight Independent Set of Rectangles

Anna Adamaszek and Andreas Wiese
Max-Planck-Institut für Informatik
Saarbrücken, Germany
Email: {anna,awiese}@mpi-inf.mpg.de

***Abstract*—In the Maximum Weight Independent Set of Rectangles (MWISR) problem we are given a set of n axis-parallel rectangles in the 2D-plane, and the goal is to select a maximum weight subset of pairwise non-overlapping rectangles. Due to many applications, e.g. in data mining, map labeling and admission control, the problem has received a lot of attention by various research communities. We present the first $(1+\varepsilon)$-approximation algorithm for the MWISR problem with quasi-polynomial running time $2^{\mathrm{poly}(\log n/\varepsilon)}$. In contrast, the best known polynomial time approximation algorithms for the problem achieve superconstant approximation ratios of $O(\log\log n)$ (unweighted case) and $O(\log n/\log\log n)$ (weighted case).**

Key to our results is a new geometric dynamic program which recursively subdivides the plane into polygons of bounded complexity. We provide the technical tools that are needed to analyze its performance. In particular, we present a method of partitioning the plane into small and simple areas such that the rectangles of an optimal solution are intersected in a very controlled manner. Together with a novel application of the weighted planar graph separator theorem due to Arora et al. [4] this allows us to upper bound our approximation ratio by $1+\varepsilon$.

Our dynamic program is very general and we believe that it will be useful for other settings. In particular, we show that, when parametrized properly, it provides a *polynomial time* $(1+\varepsilon)$-approximation for the special case of the MWISR problem when each rectangle is relatively large in at least one dimension. Key to this analysis is a method to tile the plane in order to approximately describe the topology of these rectangles in an optimal solution. This technique might be a useful insight to design better polynomial time approximation algorithms or even a PTAS for the MWISR problem. In particular, note that our results imply that the MWISR problem is not APX-hard, unless $\mathsf{NP} \subseteq \mathsf{DTIME}(2^{\mathrm{polylog}(n)})$.

I. Introduction

One of the most fundamental problems in combinatorial optimization is the Independent Set problem: given an undirected graph, find a set of pairwise non-adjacent vertices with maximum total weight. While the general problem is essentially intractable (it is NP-hard to approximate with a factor $n^{1-\varepsilon}$ for any $\varepsilon > 0$ [22]), many special cases allow much better approximation ratios.

One extensively studied setting are graphs which stem from geometric shapes in the 2D-plane. Given a set of geometric objects in the plane, the goal is to find a set of pairwise non-overlapping objects with maximum total weight. Depending on the complexity of these shapes, the approximation factors of the best known polynomial time algorithms range from $1+\varepsilon$ for fat objects [12], to n^{ε} for arbitrary shapes [14]. Observe that the latter is still much better than the complexity lower bound of $n^{1-\varepsilon}$ for arbitrary Independent Set instances.

Interestingly, there is a very large gap between the best known approximation factors when the considered objects are squares of arbitrary sizes and when they are rectangles. For squares, a $(1+\varepsilon)$-approximation has been known for several years [12]. For the rectangles, the best known approximation factors are $O(\log n/\log\log n)$ for the general case [10], and $O(\log\log n)$ for the cardinality case [7]. Importantly, no constant factor approximation algorithms are known for rectangles, while the best known hardness result is NP-hardness [13], [16]. These gaps remain despite a lot of research on the problem [2], [5], [7], [9], [10], [13], [16], [17], [19], [20], which is particularly motivated by its many applications in areas such as channel admission control [19], map labeling [2], [11], and data mining [15], [17], [18].

Since even for arbitrary shapes the best known hardness result is NP-hardness, it seems that more sophisticated algorithmic techniques and/or complexity results are needed to fully understand the Independent Set problem in the geometric setting.

A. Related Work

The maximum weight independent set of rectangles problem has been widely studied. There are several $O(\log n)$ approximation algorithms known [2], [17], [20], and in fact the hidden constant can be made arbitrarily small since for any k there is a $\lceil \log_k n \rceil$-approximation algorithm due to Berman et al. [5]. Eventually, a $O(\log n/\log\log n)$-approximation algorithm has been presented by Chan and Har-Peled [10]. Some algorithms have been studied which perform better for special cases of MWISR. There is a $4q$-approximation algorithm due to Lewin-Eytan, Naor, and Orda [19] where q denotes the size of the largest clique in the given instance. In case that the optimal independent set has size βn for some $\beta \leq 1$, Agarwal and Mustafa present an algorithm which computes an independent set of size $\Omega(\beta^2 n)$ [3].

In a break-through result, Chalermsook and Chuzhoy give a $O(\log\log n)$-approximation algorithm for the cardinality case [7], which is based on the natural LP-relaxation of the

0272-5428/13 $26.00 © 2013 IEEE
DOI 10.1109/FOCS.2013.50

problem. In fact, it is a challenging open problem to determine the exact integrality gap of the LP. Currently the best known upper bounds for it are $O(\log n/\log\log n)$ [10] for the weighted case, and $O(\log\log n)$ [7] for the cardinality case. The best known lower bounds on the integrality gap are $3/2$ [7] and 2 [21], both already for the cardinality case. There is a strong connection between the integrality gap of the LP and the maximum ratio between the coloring- and the clique-number of a set of rectangles, see [8] and references therein.

Interestingly, for the special case when all given rectangles are squares of arbitrary sizes, the problem is much better understood. There is a polynomial time $(1+\varepsilon)$-approximation algorithm by Erlebach, Jansen and Seidel [12], which works even for the more general case of arbitrary fat objects. For the unweighted squares, and also for the more general setting of unweighted pseudo-disks, even a simple local search algorithm gives a PTAS [10].

Although the complexity is well-understood in the setting of squares, for rectangles it is still widely open. In particular, the techniques of the above approximation schemes for squares do not carry over to rectangles. The PTAS from [12] requires that every horizontal or vertical line intersects only a bounded number of objects of the optimal solution that are relatively large in at least one dimension. For rectangles, this number can be up to $\Theta(n)$ which is too much. For local search, one can easily construct examples showing that for any size of the local search neighborhood (which gives quasi-polynomial running time) the optimum is missed by an arbitrarily large (superconstant) factor.

For line segments in the plane Agarwal and Mustafa [3] give an algorithm which finds an independent set of size $\sqrt{OPT/\log(2n/OPT)}$ which yields a worst case approximation factor of $n^{1/2+o(1)}$. This was improved by Fox and Pach to n^{ε} for any $\varepsilon > 0$ and their algorithm even works for k-intersecting curves with constant k [14]. Note that already for lines with at most one bend (i.e., lines forming an "L") the natural LP-relaxation has an integrality gap of $\Omega(n)$.

To the best of our knowledge, no inapproximability result is known for MWISR (and not even for arbitrary shapes in the 2D-plane). In particular, an important open problem is to construct a polynomial time constant factor approximation algorithm for MWISR.

B. Our Contribution and Techniques

We present the first $(1+\varepsilon)$-approximation algorithm for the Maximum Weight Independent Set of Rectangles problem with a quasi-polynomial running time of $2^{\mathrm{poly}(\log n/\varepsilon)}$. In contrast, the best known polynomial time approximation algorithms achieve approximation ratios of $O(\log n/\log\log n)$ for the weighted case [10], and $O(\log\log n)$ for the cardinality case [7]. We are not aware of any previous algorithms for the problem with quasi-polynomial running time which would give better bounds than the above mentioned polynomial time algorithms. Our quasi-PTAS rules out the possibility that the problem is APX-hard, assuming that $\mathsf{NP} \not\subseteq \mathsf{DTIME}(2^{\mathrm{polylog}\,(n)})$, and thus it suggests that it should be possible to obtain significantly better polynomial time approximation algorithms for the problem. In addition, we present a PTAS for the case that each rectangle is δ-large in at least one dimension, i.e., if at least one of its edges has length at least δN for some constant $\delta > 0$, assuming that in the input only integer coordinates within $\{0,...,N\}$ occur.

Key to our results is a new geometric dynamic program *GEO-DP* whose DP-table has one entry for each axis-parallel polygon P with at most k edges, where k is a fixed parameter. Such a cell corresponds to a subproblem where the input consists only of the input rectangles contained in P. The algorithm solves each such subproblem by trying every possible subdivision of P into at most k polygons with again at most k edges each, and selects the partition with maximum weight according to the DP-cells of all subproblems.

For analyzing this algorithm, we show that there is a recursive sequence of partitions such that the rectangles of OPT intersected within that sequence have a total weight of at most $\varepsilon \cdot OPT$. For our QPTAS, we first provide a method to tile the plane into polygons such that each rectangle of the optimal solution is intersected only $O(1)$ times. Using a new stretching method for the input area, we can guarantee that each face of our partition either contains only rectangles of relatively small total weight, or contains at most one rectangle. With a planar separator theorem from [4] we can find a cut in the partition such that the intersected rectangles have only marginal weight and both sides of the cut contain rectangles whose total weight is upper bounded by $\frac{2}{3}OPT$. When using these cuts in every iteration, the recursion terminates after $O(\log n/\varepsilon)$ levels and we show that by setting $k := (\frac{\log n}{\varepsilon})^{O(1)}$ we obtain an approximation ratio of $1+\varepsilon$ in quasi-polynomial time.

We demonstrate the potential of our new algorithm by proving that it yields a *polynomial* time $(1+\varepsilon)$-approximation algorithm for the special case when each rectangle is large in at least one dimension, as defined above. For this result, we employ a finer partition of the plane which ensures that in the initial partition only large rectangles with small total weight are intersected. Even more, each face of the subdivision is either a path or a cycle of a small width (strictly smaller than the longer edge of each large rectangle). Using this, we show that GEO-DP solves each resulting subproblem within an accuracy of $1+\varepsilon$ by using only subpolygons with at most a constant number of edges. Therefore, we prove that GEO-DP parametrized by $k := (1/\varepsilon)(1/\delta)^{O(1)}$ gives a polynomial time $(1+\varepsilon)$-approximation for δ-large rectangles (for any constant $\delta > 0$). In fact, this yields a PTAS for the case that the lengths of the longer edges of the rectangles differ by

at most a constant factor (when the parameter k is chosen appropriately). We would like to point out that the initial partition might be a useful ingredient for constructing a PTAS for the general problem since it sparsely describes the topology of the large rectangles while losing only an ε-fraction of their total weight.

We can well imagine that our algorithmic approach finds applications for solving the INDEPENDENT SET problem for more general geometric shapes. Given the large gaps in terms of approximation and hardness results for the INDEPENDENT SET problem in such settings we hope that our new techniques will help to bridge these gaps. Finally, we would like to note that our DP might well yield a constant factor approximation or even a PTAS for MWISR when parametrized by a sufficiently large parameter k independent of n, e.g., $k = (1/\varepsilon)^{O(1)}$. We leave this as an open question. Due to space constraints some of the proofs were omitted in this extended abstract, but can be found in the full version of this paper [1].

C. Problem Definition

We are given a set of n axis-parallel rectangles $\mathcal{R} = \{R_1, ..., R_n\}$ in the 2-dimensional plane. Each rectangle R_i is specified by two opposite corners $(x_i^{(1)}, y_i^{(1)}) \in \mathbb{N}^2$ and $(x_i^{(2)}, y_i^{(2)}) \in \mathbb{N}^2$, with $x_i^{(1)} < x_i^{(2)}$ and $y_i^{(1)} < y_i^{(2)}$, and a weight $w(R_i) \in \mathbb{R}^+$. We define the area of a rectangle as the open set $R_i := \{(x, y) | x_i^{(1)} < x < x_i^{(2)} \wedge y_i^{(1)} < y < y_i^{(2)}\}$. The goal is to select a subset of rectangles $\mathcal{R}' \subseteq \mathcal{R}$ such that for any two rectangles $R, R' \in \mathcal{R}'$ we have $R \cap R' = \emptyset$. Our objective is to maximize the total weight of the selected rectangles $w(\mathcal{R}') := \sum_{R \in \mathcal{R}'} w(R)$. For each rectangle R_i we define its *width* g_i by $g_i := x_i^{(2)} - x_i^{(1)}$ and its *height* h_i by $h_i = y_i^{(2)} - y_i^{(1)}$.

By losing at most a (multiplicative) factor of $(1 - \varepsilon)^{-1}$ in the objective, we assume that $1 \leq w(R) \leq n/\varepsilon$ for each rectangle $R \in \mathcal{R}$. First, we scale the weights of all rectangles such that $\max_{R \in \mathcal{R}} w(R) = n/\varepsilon$. Since then $OPT \geq n/\varepsilon$, all rectangles R' with $w(R') < 1$ can contribute a total weight of at most $n \cdot 1 = \varepsilon \cdot \frac{n}{\varepsilon} \leq \varepsilon \cdot OPT$. We remove them from the instance which reduces the weight of the optimal solution by at most $\varepsilon \cdot OPT$.

II. THE ALGORITHM GEO-DP

Our results are achieved by using a new geometric dynamic programming algorithm which we call GEO-DP and which we define in this section. The algorithm is parametrized by a value $k \in \mathbb{N}$ which affects both the running time and the achieved approximation ratio. In brief, the algorithm has a DP-cell for each axis-parallel polygon P with at most k edges, which represents the subproblem consisting of all rectangles contained in P. When computing a (near-optimal) solution for this subproblem, GEO-DP tries all possibilities to subdivide P into at most k polygons with at most k edges each and recurses.

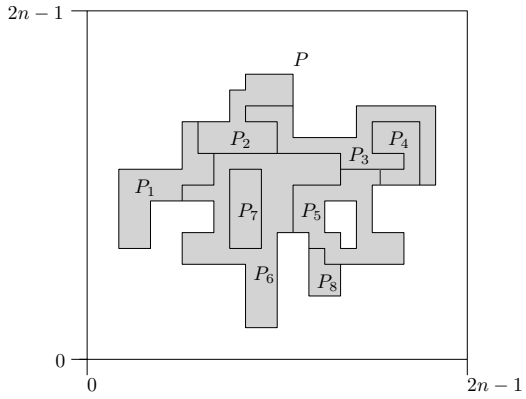

Figure 1. The partition of a polygon P (gray area) into at most k smaller polygons, each with at most k edges.

Without loss of generality we assume that $x_i^{(1)}, y_i^{(1)}, x_i^{(2)}, y_i^{(2)} \in \{0, ..., 2n - 1\}$ for each $R_i \in \mathcal{R}$. If this is not the case, we transform the instance in polynomial time into a combinatorially equivalent instance with the latter property.

Fix a parameter $k \in \mathbb{N}$. Let $\mathcal{P}$ denote the set of all polygons within the $[0, 2n - 1] \times [0, 2n - 1]$ input square whose corners have only integer coordinates and which have at most k axis-parallel edges each. Whenever we speak of a polygon, we allow it to have holes and we do not require it to be simple. In particular, by the edges of a polygon with holes we mean both the outer edges and the edges bounding the holes. We introduce a DP-cell for each polygon $P \in \mathcal{P}$, where a cell corresponding to P stores a near-optimal solution $sol(P) \subseteq \mathcal{R}_P$ where $\mathcal{R}_P$ denotes the set of all rectangles from $\mathcal{R}$ which are contained in P.

Proposition 1. *The number of DP-cells is at most* $n^{O(k)}$.

To compute the solution $sol(P)$ for some polygon $P \in \mathcal{P}$ we use the following procedure. If $\mathcal{R}_P = \emptyset$ then we set $sol(P) := \mathcal{R}_P$ and terminate. Otherwise, we enumerate all possibilities to partition P into k' polygons $P_1, ..., P_{k'} \in \mathcal{P}$ such that $k' \leq k$. See Figure 1 for a sketch. Since by Proposition 1 we have $|\mathcal{P}| \leq n^{O(k)}$, the number of potential partitions we need to consider is upper bounded by $\binom{n^{O(k)}}{k} = n^{O(k^2)}$. Let $P_1, ..., P_{k'}$, where $k' \leq k$, be a feasible partition (for any enumerated set $\{P_1, ..., P_{k'}\} \subseteq \mathcal{P}$ this can be verified efficiently since all polygons have axis-parallel edges with integer coordinates in $\{0, ..., 2n - 1\}$). For each polygon $P_i \in \{P_1, ..., P_{k'}\}$ we look up the DP-table value $sol(P_i)$ and compute $\sum_{i=1}^{k'} w(sol(P_i))$. We set $sol'(P) := \cup_{i=1}^{k'} sol(P_i)$ for the partition $\{P_1, ..., P_{k'}\}$ which yields the maximum profit. Now we define $sol(P) := sol'(P)$ if $w(sol'(P)) > \max_{R \in \mathcal{R}_P} w(R)$, and otherwise $sol(P) := \{R_{\max}\}$ for a rectangle $R_{\max} \in \mathcal{R}_P$ with maximum profit. At the end, the algorithm outputs the value in the DP-cell which corresponds to the polygon containing

the entire input region $[0, 2n-1] \times [0, 2n-1]$.

Since $|\mathcal{P}| \leq n^{O(k)}$ we get the following upper bound on the running time of GEO-DP.

Proposition 2. *When parametrized by k the running time of GEO-DP is upper bounded by $n^{O(k^2)}$.*

For bounding the approximation ratio of GEO-DP for any parameter k, it is sufficient to consider only the special case that the input set $\mathcal{R}$ is already a feasible (optimal) solution. Therefore, we will assume this from now on.

We note that in a sense GEO-DP is a generalization of an algorithm presented in [6] for a special case of MWISR.

III. Quasi-Polynomial Time Approximation Scheme

In this section we prove that GEO-DP achieves an approximation ratio of $1+\varepsilon$ when parametrized by $k = (\frac{\log n}{\varepsilon})^{O(1)}$ and is thus a QPTAS for the MWISR problem (using Proposition 2).

Key ingredient for our analysis is to show that for any set of feasible rectangles there is a *balanced cheap cut*, i.e., a polygon which consists of only few edges, which intersects rectangles from $\mathcal{R}$ of marginal total weight, and which separates the rectangles from $\mathcal{R}$ into two parts of similar size. By applying such cuts recursively for $O(\log n/\varepsilon)$ levels, we eventually obtain trivial subproblems. For proving that such good cuts always exist, we partition the plane into polygons in such a way that each rectangle is intersected only $O(1)$ times and each face of the partition consists either of exactly one rectangle or intersects rectangles of only small total weight. To ensure the latter, we apply a stretching procedure to the input before actually defining the partition. On the constructed partition we apply the weighted planar graph separator theorem from [4] to obtain the cut.

A. Balanced Cheap Cuts

We introduce *balanced α-cheap ℓ-cuts*, where α is a small positive value. Intuitively, given any set of non-overlapping rectangles $\bar{\mathcal{R}}$, such a cut is given by a polygon P with at most ℓ axis-parallel edges whose boundary intersects rectangles with weight at most $\alpha \cdot w(\bar{\mathcal{R}})$ such that the interior and the exterior of P each contain only rectangles whose weight is at most $2/3 \cdot w(\bar{\mathcal{R}})$.

Definition 3. Let $\ell \in \mathbb{N}$ and $\alpha \in \mathbb{R}$ with $0 < \alpha < 1$. Let $\bar{\mathcal{R}}$ be a set of pairwise non-overlapping rectangles. A polygon P with axis-parallel edges is a *balanced α-cheap ℓ-cut* if:

- P has at most ℓ edges,
- for the set of all rectangles $\mathcal{R}' \subseteq \bar{\mathcal{R}}$ intersecting the boundary of P we have $w(\mathcal{R}') \leq \alpha \cdot w(\bar{\mathcal{R}})$,
- for the set of all rectangles $\mathcal{R}_{\text{in}} \subseteq \bar{\mathcal{R}}$ contained in P it holds that $w(\mathcal{R}_{\text{in}}) \leq 2/3 \cdot w(\bar{\mathcal{R}})$, and
- for the set of all rectangles $\mathcal{R}_{\text{out}} \subseteq \bar{\mathcal{R}}$ contained in the complement of P, i.e., in $\mathbb{R}^2 \setminus P$, it holds that $w(\mathcal{R}_{\text{out}}) \leq 2/3 \cdot w(\bar{\mathcal{R}})$.

As we will show in the next lemma, GEO-DP performs well if for any set of rectangles there exists a good cut.

Lemma 4. *Let $\varepsilon > 0$. Let $\alpha > 0$ with $\alpha < 1/2$ and $\ell \geq 4$ be values such that for any set $\bar{\mathcal{R}}$ of pairwise non-overlapping rectangles there exists a balanced α-cheap ℓ-cut, or there is a rectangle $R \in \bar{\mathcal{R}}$ such that $w(R) \geq \frac{1}{3} \cdot w(\bar{\mathcal{R}})$. Then GEO-DP has approximation ratio $(1+\alpha)^{O(\log(n/\varepsilon))}$ when parametrized by $k = \ell^2 \cdot O(\log^2(n/\varepsilon))$.*

Proof (sketch): Starting with the $[0, 2n-1] \times [0, 2n-1]$ input square, we either find a rectangle R with $w(R) \geq \frac{1}{3} \cdot w(\bar{\mathcal{R}})$, or a balanced α-cheap ℓ-cut. In either case we get a decomposition of the problem into subproblems, where each subproblem is defined by a polygon which contains rectangles whose total weight is at most a $2/3$-fraction of $w(\mathcal{R})$. Continuing for $O(\log n/\varepsilon)$ recursion levels, we obtain subproblems consisting of at most one rectangle each (as $1 \leq w(R) \leq n/\varepsilon$ for each $R \in \mathcal{R}$). Each appearing subproblem can be expressed as the intersection of $O(\log n/\varepsilon)$ polygons with at most $\ell+4$ edges each (we have to add 4 since the boundary of the input square can become the outer boundary of a polygon). Thus, the boundary of each considered subproblem consists of $O(\ell^2 \cdot \log^2(n/\varepsilon))$ edges. Additionally, we can show that each subproblem gives rise to $O(\ell^2 \cdot \log^2(n/\varepsilon))$ subproblems in the next recursion level. As GEO-DP tries all partitions of a polygon into at most k polygons, each with at most k edges, at each recursion level it will consider the partition corresponding to the cut. Since at each level we lose rectangles whose weight is at most an α-fraction of the total weight of the rectangles from the current recursion level, we obtain the claimed approximation ratio. ∎

In the remainder of this section we prove that for any set $\bar{\mathcal{R}}$ and any $\delta > 0$ there is a balanced $O(\delta)$-cheap $\left(\frac{1}{\delta}\right)^{O(1)}$-cut or there is a rectangle $R \in \bar{\mathcal{R}}$ with $w(R) \geq \frac{1}{3} \cdot w(\bar{\mathcal{R}})$. This implies our main result when choosing $\delta := \Theta(\varepsilon/\log(n/\varepsilon))$. Note that from now on our reasoning does not need to be (algorithmically) constructive.

B. Stretching the Rectangles

For the purpose of finding a good cut, we are free to stretch or squeeze the rectangles of $\mathcal{R}$. We do this as a preprocessing step in order to make them well-distributed.

Definition 5. A set of rectangles $\bar{\mathcal{R}}$ with integer coordinates in $\{0, ..., N\}$ is *well-distributed* if for any $\gamma > 0$ and for any $t \in \{0, ..., N\}$ we have that all rectangles contained in the area $[0, N] \times [t, t + \gamma \cdot N]$ have a total weight of at most $2\gamma \cdot w(\mathcal{R})$. We require the same for the rectangles contained in the area $[t, t + \gamma \cdot N] \times [0, N]$.

We say that two sets of rectangles $\mathcal{R}, \bar{\mathcal{R}}$ are *combinatorially equivalent* if we can obtain one from the other by stretching or squeezing the input area, possibly non-uniformly.

Lemma 6. *Let $\mathcal{R}$ be a set of rectangles with arbitrary weights. There is a combinatorially equivalent set $\bar{\mathcal{R}}$ using only integer coordinates in $\{0, ..., 4 \cdot |\mathcal{R}|\}$ which is well-distributed.*

Proof (sketch): Without loss of generality we assume that $x_i^{(1)}, y_i^{(1)}, x_i^{(2)}, y_i^{(2)} \in \{0, ..., 2n-1\}$ for each $R_i \in \mathcal{R}$, where $n = |\mathcal{R}|$. We stretch the original input square in such a way that the lengths of its sides double (i.e., increase by another $2|\mathcal{R}|$), and the distance between two original consecutive x-coordinates (y-coordinates) x_i and $x_i + 1$ increases proportionally to the weight of all rectangles "starting" at the x-coordinate (y-coordinate) x_i. We then need to introduce some rounding, as we want the new coordinates of all the rectangles to be integral. The new set of rectangles is clearly equivalent to the original one.

Consider any vertical stripe S in the modified input square with left- and rightmost x-coordinates x'_a and x'_b, respectively. There is a set of x-coordinates $x_1, x_2, ..., x_m$ from the original instance which were mapped to values $x'_1, x'_2, ..., x'_m \in [x'_a, x'_b]$. All rectangles contained in S must have their respective leftmost x-coordinates in $\{x'_1, x'_2, ..., x'_{m-1}\}$. The total weight of rectangles with leftmost coordinate x'_i is proportional to $x'_{i+1} - x'_i$, for each i. Hence, the total weight of rectangles contained in S is proportional to $x'_m - x'_1 \leq x'_b - x'_a$. The same is true for horizontal stripes, and so the modified input instance is well-distributed. ∎

Observe that there is a balanced α-cheap ℓ-cut for any values α and ℓ in the stretched instance if and only if there is such a cut in the original instance. Thus, suppose from now on that we have a well-distributed set of pairwise non-intersecting rectangles $\mathcal{R}$, using integer coordinates in $\{0, ..., N\}$ for some integer N, and a value $\delta > 0$ such that $1/\delta \in \mathbb{N}$. As we do not require any special bound on the value of N, we can scale up all coordinates of the rectangles by a factor of $(1/\delta)^2$, and therefore we can assume that $\delta^2 N$ is an integer.

C. Partitioning the Plane

We define a procedure to partition the input square $I := [0, N] \times [0, N]$. This partition is defined by only $(1/\delta)^{O(1)}$ lines, and it has the properties that each rectangle in $\mathcal{R}$ is intersected only $O(1)$ times and each face either surrounds exactly one rectangle or it has non-empty intersection with rectangles with small total weight of at most $O(\delta^2 w(\mathcal{R}))$.

We call a rectangle $R_i \in \mathcal{R}$ *large* if $h_i > \delta^2 N$ or $g_i > \delta^2 N$, and *small* if $h_i \leq \delta^2 N$ and $g_i \leq \delta^2 N$. We denote the subsets of $\mathcal{R}$ consisting of large and small rectangles by $\mathcal{R}_L$ and $\mathcal{R}_S$, respectively. We call a rectangle $\mathcal{R}_i \in \mathcal{R}$ *vertical* if $h_i > g_i$, and *horizontal* if $h_i \leq g_i$. We say that a line L *cuts* a rectangle $R \in \mathcal{R}$, if $R \setminus L$ has two connected components.

We now present the construction of the partition. It will consist of a set of straight axis-parallel lines $\mathcal{L}$ contained in the input square I, and containing the boundary of I. A connected component of $I \setminus \mathcal{L}$ is called a *face*, and the set of faces is denoted by $\mathcal{F}(\mathcal{L})$. Note that the faces are open polygons and for any $F \in \mathcal{F}(\mathcal{L})$ and $L \in \mathcal{L}$ we have $F \cap L = \emptyset$.

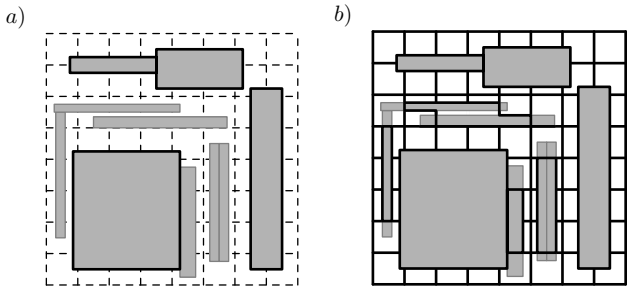

Figure 2. Creating the partition of the input square. The large rectangles $\mathcal{R}_L$ are depicted in gray. The bold edges represent the lines from $\mathcal{L}$.

Grid: We subdivide the input square I into $\frac{1}{\delta^2} \times \frac{1}{\delta^2}$ *grid cells*, where each grid cell is a square of size $\delta^2 N \times \delta^2 N$. Formally, for each $i, j \in \{0, ..., 1/\delta^2 - 1\}$ we have a grid cell $[i \cdot \delta^2 N, (i+1) \cdot \delta^2 N] \times [j \cdot \delta^2 N, (j+1) \cdot \delta^2 N]$. As $\delta^2 N$ is an integer, the corners of the grid cells have integer coordinates. The lines subdividing the input square into the grid cells are called *grid lines*.

We say that a rectangle $R \in \mathcal{R}$ *intersects* a grid cell Q, if $R \cap Q \neq \emptyset$ (recall that rectangles have been defined as open sets). Each rectangle $R \in \mathcal{R}_L$ intersects at least two grid cells, and each rectangle $R \in \mathcal{R}_S$ intersects at most four grid cells. We say that a rectangle $R \in \mathcal{R}$ *crosses* a grid cell Q, if R intersects Q and R has non-empty intersection with two opposite edges of Q. Notice that small rectangles do not cross any grid cells.

Rectangle faces: For each large vertical rectangle which is cut by a vertical grid line, and for each large horizontal rectangle which is cut by a horizontal grid line, we add the edges of the rectangle to the set of lines $\mathcal{L}$ (see Figure 2a). The added edges are called *rectangle edges*, and the faces corresponding to such rectangles are called *rectangle faces*.

Lemma 7. *The number of rectangle faces is at most $2(1/\delta)^4$.*

Notice that if a large vertical rectangle is not contained in a rectangle face then it is contained in a single column of grid cells. Similarly, if a large horizontal rectangle is not contained in a rectangle face then it is contained in a single row of grid cells.

Lines within the grid cells.: We now consider each grid cell Q separately, and proceed as follows. Let $\mathcal{R}_Q$ denote the set of rectangles from $\mathcal{R}$ which are not contained in the rectangle faces, and which cross Q. Notice that, as the rectangles from $\mathcal{R}$ are pairwise non-overlapping, $\mathcal{R}_Q$ cannot contain both vertical and horizontal rectangles.

- If $\mathcal{R}_Q = \emptyset$, we add to $\mathcal{L}$ the whole boundary of Q, with the exception of the fragments which are in the interior of the rectangle faces (see Figure 3a).
  Notice that the boundaries of the rectangle faces are in $\mathcal{L}$, so in this case $\mathcal{L} \cap Q$ is connected.
- If $\mathcal{R}_Q$ consists of vertical rectangles, let L_ℓ and L_r be the leftmost and the rightmost vertical edge of a rectangle from $\mathcal{R}_Q$. We add to $\mathcal{L}$ the lines $L_\ell \cap Q$ and $L_r \cap Q$, and the boundary of Q with the exception of the fragments which are between L_ℓ and L_r, or in the interior of the rectangle faces (see Figure 3b).
  Notice that the lines added to $\mathcal{L}$ while considering the grid cell Q do not intersect any rectangles from $\mathcal{R}_Q$.
- If $\mathcal{R}_Q$ consists of horizontal rectangles, we proceed as before, considering horizontal lines instead of vertical (see Figure 3c).

Notice that the lines from $\mathcal{L}$ can *overlap* (i.e., we can have $L_1, L_2 \in \mathcal{L}$ s.t. $L_1 \cap L_2$ is an interval), but they do not *intersect properly* (i.e., if for $L_1, L_2 \in \mathcal{L}$ we have $L_1 \cap L_2 = \{p\}$, then p is an endpoint of at least one of the lines L_1, L_2). The lines from $\mathcal{L}$ cover the boundary of the input square I. An example of the partition can be seen in Figure 2b.

Graph $G(\mathcal{L})$: We now construct a graph $G(\mathcal{L}) = (V, E)$ embedded in the input square I, representing the partition $\mathcal{L}$. Any point $p \in I$ becomes a vertex of $G(\mathcal{L})$ if and only if there is at least one line $L \in \mathcal{L}$ with an endpoint in p. For any pair of vertices $v, w \in V$ for which there is a line $L \in \mathcal{L}$ such that $\{v, w\} \in L$, and for which no vertex lies on the straight line strictly between v and w, we add an edge vw to $G(\mathcal{L})$ (i.e., edges of $G(\mathcal{L})$ represent subdivisions of lines in $\mathcal{L}$). As $\bigcup_{L\in\mathcal{L}} L = \bigcup_{e\in E} e$, the faces of $G(\mathcal{L})$ are exactly $\mathcal{F}(\mathcal{L})$. The claim of the next lemma is directly implied by the construction.

Lemma 8. *The graph $G(\mathcal{L})$ is planar and has $O((1/\delta)^4)$ vertices and $O((1/\delta)^4)$ edges.*

The following lemmas will be needed to show the existence of a balanced cut in $G(\mathcal{L})$.

Lemma 9. *Each rectangle from $\mathcal{R}$ can be intersected by at most four edges of the graph $G(\mathcal{L})$.*

Lemma 10. *Let $F \in \mathcal{F}(\mathcal{L})$. The boundary of F intersects rectangles from $\mathcal{R}$ of total weight at most $8\delta^2 w(\mathcal{R})$. If F is not a rectangle face, then F has non-empty intersection with rectangles from $\mathcal{R}$ of total weight at most $8\delta^2 w(\mathcal{R})$.*

Proof (sketch): The lemma clearly holds for rectangle faces. Let $F \in \mathcal{F}(\mathcal{L})$ be face which is not a rectangle face. If F is contained in one grid cell Q then one can show that all rectangles intersecting F must be contained in the area defined by grid column and grid row containing Q together with the two adjacent grid rows. On the other hand, if F spans several grid cells, one can show that all rectangles intersecting it must be contained in a single grid row or column. In both cases, the claim follows since $\mathcal{R}$ is well-distributed. ■

D. Defining the Cut

For obtaining our desired cut, we apply the following theorem from [4] for the graph $G(\mathcal{L})$. A *V-cycle* C is a Jordan curve in the embedding of a given planar graph G which might go along the edges of G and also might cross faces of G. The parts of C crossing an entire face of G are called *face edges.*

Theorem 11 ([4])**.** *Let G denote a planar, embedded graph with weights on the vertices and faces and with costs on the edges. Let W denote the total weight, and M the total cost of the graph. Then, for any parameter $\bar{k}$, we can find in polynomial time a separating V-cycle C such that the interior and exterior of C each has weight at most $2W/3$, C uses at most $\bar{k}$ face edges, and C uses ordinary edges of total cost $O(M/\bar{k})$.*

First, we need to assign costs to the edges of $G(\mathcal{L})$ and weights to the vertices and faces of $G(\mathcal{L})$. For each edge $e \in E$ we define its cost c_e to be the total weight of rectangles intersecting e. The weights of all vertices are zero. For each face F we define its weight w_F to be the total weight of all rectangles contained in F, plus a fraction of the weight of the rectangles which intersect the boundary of F. If a rectangle $R \in \mathcal{R}$ has non-empty intersection with m faces, each of these faces obtains a $1/m$-fraction of the weight of R. From Lemmas 9 and 10 we obtain the following bounds.

Lemma 12. *The total cost of edges in $G(\mathcal{L})$ is at most $4w(\mathcal{R})$. The weight of each non-rectangle face F is at most $8\delta^2 \cdot w(\mathcal{R})$. The total weight of the faces equals $w(\mathcal{R})$.*

For constructing the cut we apply Theorem 11 with parameter $\bar{k} = 1/\delta$ to the graph $G(\mathcal{L})$. The obtained V-cycle C yields a cut in the plane. We replace each face edge crossing some face $F \in \mathcal{F}(\mathcal{L})$ by the edges going along the boundary of F. If $w(R) \leq w(\mathcal{R})/3$ for each rectangle $R \in \mathcal{R}$ and if $\delta < 1/5$, then, from Lemma 10, each face has weight at most $w(\mathcal{R})/3$. We can then ensure that each side of the modified cut contains rectangles of total weight at most $2w(\mathcal{R})/3$. Using the upper bound on the number of edges of $G(\mathcal{L})$ from Lemma 8, and upper bounding the total weight of rectangles intersected by the modified cut (using Lemmas 10 and 12) implies the following result.

Lemma 13. *Assume that $1/5 > \delta > 0$. For any set $\mathcal{R}$ of pairwise non-overlapping rectangles not containing a rectangle of weight at least $w(\mathcal{R})/3$ there exists a balanced $O(\delta)$-cheap $O((1/\delta)^4)$-cut.*

When choosing $\delta := \Theta(\varepsilon/\log(n/\varepsilon))$, from Lemma 4 and Lemma 13 we obtain that GEO-DP is a QPTAS.

Theorem 14. *The algorithm GEO-DP parametrized by $k = (\frac{\log n}{\varepsilon})^{O(1)}$ yields a quasi-polynomial time approxi-*

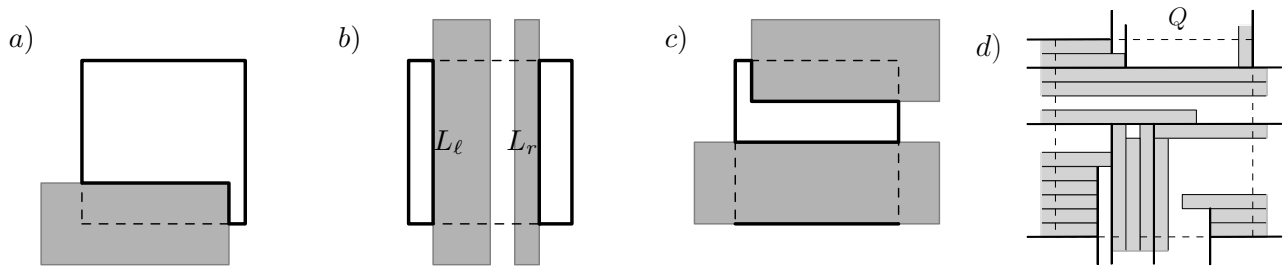

Figure 3. a)-c) Constructing the lines within a single grid cell Q. The depicted gray rectangles are either in $\mathcal{R}_Q$ or they are rectangles from $\mathcal{R}_L$ which have their own respective rectangle face. The bold lines correspond to the lines of $\mathcal{L}$ within the grid cell. d) The thick lines denote the lines in $\mathcal{L}_0$ added for the grid cell Q. The blocks of the considered instance are depicted in gray.

mation scheme for the maximum weight independent set of rectangles problem.

IV. A PTAS for Large Rectangles

In this section we show that GEO-DP yields a polynomial time approximation scheme for input instances which contain only large rectangles, i.e., in which every rectangle has width or height greater than a δ-fraction of the length of the edges of the input square, for some constant $\delta > 0$. As a corollary, we obtain a PTAS for the special case of the MWISR problem when the lengths of the longer edges of the rectangles differ only by a constant factor, i.e., for some constant $\delta > 0$ we have $\max\{h_i, g_i\} \leq 1/\delta \cdot \max\{h_{i'}, g_{i'}\}$ for all rectangles $R_i, R_{i'}$.

Let $\varepsilon > 0$ and $\delta > 0$. Let $\mathcal{R}$ be a set of rectangles, and let N be an integer such that for each rectangle $R_i \in \mathcal{R}$ we have $x_i^{(1)}, x_i^{(2)}, y_i^{(1)}, y_i^{(2)} \in \{0, ..., N\}$. We call a rectangle $R_i \in \mathcal{R}$ *δ-large* if $h_i > \delta N$ or $g_i > \delta N$. In this section we assume that the input consists of a set $\mathcal{R}$ of δ-large rectangles for some constant $\delta > 0$. Assume w.l.o.g. that $1/\delta \in \mathbb{N}$ and $\delta N \in \mathbb{N}$. As in the previous section, for the analysis of GEO-DP we can assume that $\mathcal{R}$ itself is the optimal solution, i.e., no two rectangles in $\mathcal{R}$ overlap.

Overview: First, we show that there is a way to partition the plane using a set of at most $\frac{1}{\varepsilon} \cdot (\frac{1}{\delta})^{O(1)}$ lines, such that the intersected rectangles have small total weight and each face of the partition is a path or a cycle of "width" at most δN. Note that the latter bound is strictly smaller than the length of the longer edge of each rectangle. In a sense, this partition sparsely describes the topology of the (large) rectangles while losing only rectangles of negligible weight. Then, we show that for each face GEO-DP can solve the resulting subproblem within a $(1+\varepsilon)$-accuracy, without an increase in the complexity of the subproblems during the recursion.

When given an input instance, GEO-DP first preprocesses it so that all rectangles have coordinates which are integers in $\{0, ..., 2n-1\}$. Note, however, that this routine might cause that some rectangles are not δ-large anymore. Therefore, in the analysis in this section, we show that a good recursive subdivision of the input square exists for the *original* input with coordinates in $\{0, ..., N\}$ for some integer N, and where all the rectangles are δ-large. As the preprocessing consists essentially of stretching and squeezing of the input area, there is a corresponding recursive subdivision of the preprocessed input instance, whose polygons have the same complexity, and which will be considered by GEO-DP.

A. Constructing the Partition for Large Rectangles

We define a set of lines $\mathcal{L}$ forming a partition in the input square $[0, N] \times [0, N]$. The lines in $\mathcal{L}$ will have the properties that

- $|\mathcal{L}| \leq \frac{1}{\varepsilon} \cdot (\frac{1}{\delta})^{O(1)}$,
- the rectangles intersected by a line in $\mathcal{L}$ have a total weight of at most $\varepsilon \cdot w(\mathcal{R})$, and
- each face in the partition obtained by $\mathcal{L}$ which contains rectangles from $\mathcal{R}$ is either a path or a cycle with "width" at most δN.

Without saying explicitly, from now on each considered line is either horizontal or vertical and its endpoints have integral coordinates.

Grid and blocks: We construct a grid consisting of $\frac{1}{\delta} \times \frac{1}{\delta}$ grid cells in the input square $[0, N] \times [0, N]$, i.e., for each $i, j \in \{0, ..., 1/\delta - 1\}$ there is a grid cell with coordinates $[i \cdot \delta N, (i+1) \cdot \delta N] \times [j \cdot \delta N, (j+1) \cdot \delta N]$.

We slice all rectangles parallel to their longer edge into *blocks*, i.e., rectangles of unit width or height. Formally, we cut each rectangle $R_i \in \mathcal{R}$ with $h_i > g_i$ into $x_i^{(2)} - x_i^{(1)}$ *vertical blocks*, with the corners $(j, y_i^{(1)})$ and $(j+1, y_i^{(2)})$ for $j = \{x_i^{(1)}, x_i^{(1)} + 1, \ldots, x_i^{(2)} - 1\}$. With a symmetric operation we generate *horizontal blocks* for each rectangle $R_i \in \mathcal{R}$ with $h_i \leq g_i$. We denote by $\mathcal{B}$ the set of all generated blocks and observe that also they have integer coordinates. Like the rectangles, we define the blocks as open sets. We will first find a partition for the blocks, which essentially means that in the first version of the partition we can cut the rectangles arbitrarily parallel to their longer edges. Later we will show how to adjust the partition—by introducing some detours—so that these cuts will be eliminated.

We use the following notation. A line L *touches* a rectangle R if $L \cap (R \cup \partial R) \neq \emptyset$. A line L *intersects* a rectangle R if $L \cap R \neq \emptyset$. A line L *hits* a rectangle R if L touches R, L does not intersect R, but extending L would result in L intersecting R. A line L *cuts* a rectangle R if $R \setminus L$ has two connected components. We say that a rectangle R (a block B, a line L) *intersects* a grid cell Q if $R \cap \mathrm{int}(Q) \neq \emptyset$ ($B \cap \mathrm{int}(Q) \neq \emptyset$, $L \cap \mathrm{int}(Q) \neq \emptyset$). Each rectangle $R \in \mathcal{R}$, as well as each block $B \in \mathcal{B}$, intersects at least two grid cells. We say that a block $B \in \mathcal{B}$ *ends* in a grid cell Q, if B intersects Q, and for a short edge e (i.e., an edge with unit length) of B we have $e \subseteq Q$.

Initial set of lines: We start by introducing an initial set of lines $\mathcal{L}_0$ as follows. First, we add to $\mathcal{L}_0$ four lines which form the boundary of the input square $[0, N] \times [0, N]$.

Consider a grid cell Q and its bottom edge e. If possible, we add to $\mathcal{L}_0$ the following maximal lines which do not intersect any block or any line previously added to $\mathcal{L}_0$, which touch e, and which are strictly longer than δN:

- a vertical line with the smallest possible x-coordinate,
- a vertical line with the largest possible x-coordinate,
- a vertical line L which maximizes the length of $L \cap Q$. If there are several such lines, we add two: one with the smallest and one with the largest x-coordinate. Lines maximizing $|L \cap Q|$ are called *sticking-in lines* for e in Q.

We do the same operation for the top, left and right edges of Q, where for the left and right edges we take horizontal lines, considering the y-coordinates instead of the x-coordinates. We do this in a fixed order, e.g., first we add all vertical lines, and then all horizontal lines. See Figure 3d) for an example. We do not want $\mathcal{L}_0$ to be a multi-set and thus we add each line at most once. Note that for any two lines $L_1, L_2 \in \mathcal{L}_0$ the intersection $L_1 \cap L_2$ is either empty or consists of one single point (which is the endpoint of one of the lines). All lines in $\mathcal{L}_0$ are maximal, which means that they cannot be extended without intersecting any perpendicular block or a perpendicular line in $\mathcal{L}_0$.

Proposition 15. *The set $\mathcal{L}_0$ consists of at most $16(\frac{1}{\delta})^2 + 4$ lines.*

Extending lines: A line in $\mathcal{L}_0$ might have *loose ends* which are endpoints which are not contained in some other line in $\mathcal{L}_0$. We fix this by adding a set of lines $\mathcal{L}_{\mathrm{ext}}$. We extend each loose end p of a line in $\mathcal{L}_0$ by a path connecting p either to a line in $\mathcal{L}_0$ or to a line in the so far computed set $\mathcal{L}_{\mathrm{ext}}$. Such a path will contain $O(1/(\varepsilon\delta^2))$ horizontal or vertical line segments, and will cut only rectangles of total weight $O(\varepsilon\delta^2 \cdot w(\mathcal{R}))$ parallel to their shorter edges. We add the lines of this path to $\mathcal{L}_{\mathrm{ext}}$ and continue with the next loose end of a line in $\mathcal{L}_0$.

Due to space constraints we present just the idea of the construction. As all lines from $\mathcal{L}_0$ are maximal, if an endpoint of a line $L \in \mathcal{L}_0$ does not hit a line from $\mathcal{L}_0$ then it must hit a perpendicular block $B \in \mathcal{B}$. The first line L_1 on the path goes along the boundary of B such that it crosses the boundary of a grid cell (as blocks are not contained in a single grid cell, their longer edges always cross the boundary of a grid cell). We extend L_1 so that it either touches a line from $\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}}$ (and the construction of the path is finished), or hits some perpendicular block. In the latter case we proceed with the construction of the path, considering a loose end of L_1 instead of L. Notice that in this part of the construction the path does not intersect any rectangles parallel to their shorter edges (i.e., it does not intersect any blocks). After $O(1/(\varepsilon\delta^2))$ steps either the path ends by touching a line from $\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}}$, or we can make a shortcut by adding a line L' at the end of the path such that L' goes along a grid cell boundary, connects the path with a line from $\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}}$, and cuts rectangles of total weight $O(\varepsilon\delta^2 \cdot w(\mathcal{R}))$ parallel to their shorter edges. See Figure 4 for an example of the construction.

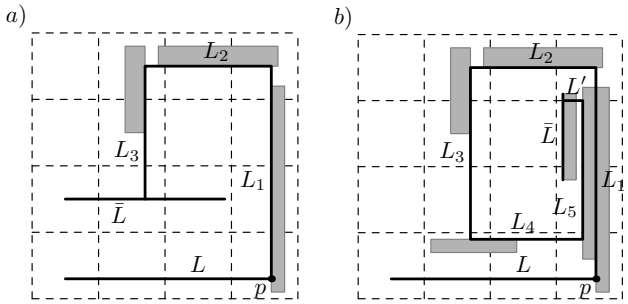

Figure 4. The construction of the lines $\mathcal{L}_{\mathrm{ext}}$. A new path is constructed, starting at a loose end p of a line $L \in \mathcal{L}_0$. In case a) the construction of the path ends when L_3 hits a line $\bar{L} \in \mathcal{L}_0$. In case b) we make a "cheap shortcut", by ending the path $L_1, \ldots, L_5$ with a line L', which connects L_5 with a line $\bar{L} \in \mathcal{L}_0$. L' cuts rectangles of small total weight.

We say that a set of lines $\mathcal{L}$ is *nicely connected* if no two lines $L, L' \in \mathcal{L}$ overlap (i.e., share more than one point) or intersect properly (i.e., such that $L \cup L' \setminus \{L \cap L'\}$ has four connected components) and for any endpoint p of a line $L \in \mathcal{L}$ there is a line $L' \in \mathcal{L}$, perpendicular to L, such that $L \cap L' = \{p\}$.

Lemma 16. *The set of lines $\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}}$ is nicely connected, $|\mathcal{L}_{\mathrm{ext}}| \leq \frac{1}{\varepsilon} \cdot (\frac{1}{\delta})^{O(1)}$, and the total weight of rectangles in $\mathcal{R}$ cut by some line in $\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}}$ parallel to their shorter edge is upper bounded by $\varepsilon \cdot w(\mathcal{R})$. Also, all lines in $\mathcal{L}_{\mathrm{ext}}$ cutting rectangles in $\mathcal{R}$ lie on some grid line and for each line $L \in \mathcal{L}_{\mathrm{ext}}$ there exists no cell Q such that $L \subseteq \mathrm{int}(Q)$.*

Faces of the partition: The lines $\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}}$ subdivide the input square into a set of faces which are the connected components of $I \setminus (\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}})$ (so in particular, the faces are open sets). Denote by $\mathcal{F}(\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}})$ the set of all faces of this partition, and by $\mathcal{F}_+(\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}})$ the set of all faces which contain at least one block from $\mathcal{B}$. As the next lemma shows, inside of each grid cell each face from the set $\mathcal{F}_+(\mathcal{L}_0 \cup \mathcal{L}_{\mathrm{ext}})$

has a simple structure. We say that a polygon P is an *L-shape* if its boundary has exactly six axis-parallel edges.

Lemma 17. *Consider a face $F \in \mathcal{F}_+(\mathcal{L}_0 \cup \mathcal{L}_{\text{ext}})$ and let Q be a grid cell with $F \cap Q \neq \emptyset$. Consider one connected component C of $F \cap Q$. Then $\text{int}(C)$ is the interior of a rectangle or the interior of an L-shape. Also, $C \cap \partial Q$ consists of one or two disjoint lines.*

Now we study the structure of the faces in $\mathcal{F}_+(\mathcal{L}_0 \cup \mathcal{L}_{\text{ext}})$ at the boundary of the grid cells. In the following lemma we show that multiple connected components of a face inside one grid cell Q' cannot merge into one component in a neighboring grid cell Q.

Lemma 18. *Let Q and Q' be grid cells such that $Q \cap Q' = \{e\}$ for an edge e. Consider a face $F \in \mathcal{F}_+(\mathcal{L}_0 \cup \mathcal{L}_{\text{ext}})$ such that $F \cap Q \neq \emptyset$, and let C be a connected component of $F \cap Q$ such that $C \cap e \neq \emptyset$. Then there is exactly one connected component C' of $F \cap Q'$ such that $C \cap C' \neq \emptyset$.*

Circumventing some rectangles: As the last step of the construction of the partition, we want to ensure that if a line L in our construction intersects a rectangle $R \in \mathcal{R}$, then it cuts R parallel to its short edge. We achieve this as follows: whenever a line $L \in \mathcal{L}_0 \cup \mathcal{L}_{\text{ext}}$ intersects a rectangle $R \in \mathcal{R}$ such that $R \setminus L$ has only one connected component or $R \cap L$ is longer than δN, then we add the four edges of R as new lines and remove all parts of lines from $\mathcal{L}_0 \cup \mathcal{L}_{\text{ext}}$ which are inside R. For any line in $\mathcal{L}_0 \cup \mathcal{L}_{\text{ext}}$ this operation adds at most $O(1/\delta)$ new lines. Denote by $\mathcal{L}$ the resulting final set of lines. Similarly as above, the set $\mathcal{F}(\mathcal{L})$ denotes all faces, and the set $\mathcal{F}_+(\mathcal{L})$ denotes all faces which contain at least one rectangle.

Using the bounds on the number of edges in $\mathcal{L}_0 \cup \mathcal{L}_{\text{ext}}$ from Proposition 15 and Lemma 16, and the upper bound on the total weight of rectangles in $\mathcal{R}$ cut by a line from $\mathcal{L}_0 \cup \mathcal{L}_{\text{ext}}$ parallel to its shorter edge (Lemma 16), we can show the following result.

Lemma 19. *The set of lines $\mathcal{L}$ has the properties that $|\mathcal{L}| \leq \frac{1}{\varepsilon} \cdot \left(\frac{1}{\delta}\right)^{O(1)}$ and the total weight of intersected rectangles is upper bounded by $\varepsilon \cdot w(\mathcal{R})$.*

B. Solving the Subproblems for the Faces

We transform the set of lines $\mathcal{L}$ into a graph $G(\mathcal{L}) = (V, E)$ in the same way as in Section III. From Lemma 19 we get that $|V| \leq (1/\varepsilon)(1/\delta)^{O(1)}$ and $|E| \leq (1/\varepsilon)(1/\delta)^{O(1)}$. The algorithm GEO-DP parametrized with $k \geq (1/\varepsilon)(1/\delta)^{O(1)}$ tries to subdivide the input square into the faces $\mathcal{F}(\mathcal{L})$ and then recurses on the subproblems given by the faces. Observe that each face in $\mathcal{F}(\mathcal{L}) \setminus \mathcal{F}_+(\mathcal{L})$ does not contain any rectangle from $\mathcal{R}$ and thus we ignore those faces from now on. We distinguish two types of faces in $\mathcal{F}_+(\mathcal{L})$: faces which are homeomorphic to a straight line, and those homeomorphic to a cycle. Note that due to Lemma 17 and Lemma 18 no more complex shapes can arise.

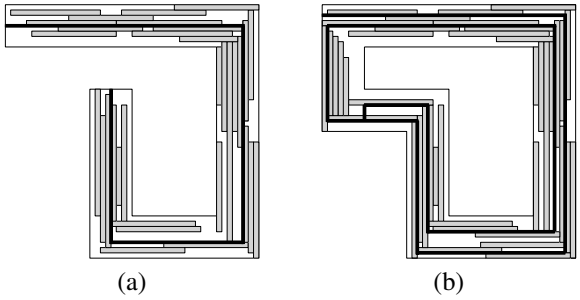

Figure 5. The thick lines show a cut of a face (a) into two paths with the same complexity as the original path and (b) into a path and a cycle with the same complexity as the original cycle.

Let $F \in \mathcal{F}_+(\mathcal{L})$ be homeomorphic to a straight line. We claim that GEO-DP finds an optimal solution for F. To get some intuition, let us pretend that F is the union of a set of complete grid cells and that all rectangles inside F are blocks, i.e., $g_i = 1$ or $h_i = 1$ for each $R_i \in \mathcal{R}$ with $R_i \subseteq F$. Then there exists a cut through F which splits F into two sub-faces without intersecting any rectangle (see Figure 5a). Moreover, the boundary of each sub-face will not be more complex than the boundary of F itself. Due to this, the complexity of the subproblems does not increase during the recursion process, and the algorithm GEO-DP finds an optimal solution for F. While for arbitrary faces homeomorphic to a straight line and arbitrary rectangles instead of blocks the analysis is more technical, and in particular requires circumventing rectangles within F, the key concept is the same.

Lemma 20. *Consider a face $F \in \mathcal{F}_+(\mathcal{L})$ which is homeomorphic to a straight line. Then GEO-DP parametrized by a value $k \geq (1/\varepsilon)(1/\delta)^{O(1)}$ computes an optimal solution for the DP-cell corresponding to F.*

Now consider a face $F \in \mathcal{F}_+(\mathcal{L})$ which forms a cycle, i.e., which is homeomorphic to S^1. Let us pretend again that F is the union of some complete grid cells and all rectangles in F are blocks. Then we can split F into a path-face F_1 and a smaller cycle F_2 while ensuring that the boundary of the faces F_1 and F_2 consists of at most $(1/\varepsilon)(1/\delta)^{O(1)}$ edges each (see Figure 5b). The recursion terminates when at some recursion level $F_2 = \emptyset$. When doing this operation repeatedly, we ensure that the total weight of intersected rectangles is only an ε-fraction of the total weight of the rectangles in the paths that we detached from the cycle.

Using this construction we can show that GEO-DP parametrized by sufficiently large k computes a $(1+\varepsilon)$-approximation for F, using that it solves the subproblems for path-faces optimally. Again, for arbitrary rectangles and more general cycle-faces F the reasoning is more technical while the core idea stays the same.

Lemma 21. *Consider a face $F \in \mathcal{F}_+(\mathcal{L})$ which is homeomorphic to S^1. Then GEO-DP parametrized by a value*

$k \geq \frac{1}{\varepsilon}(\frac{1}{\delta})^{O(1)}$ computes a $(1+\varepsilon)$-approximative solution for the DP-cell corresponding to F.

When constructing the partition given by the lines $\mathcal{L}$ we intersect (and thus lose) rectangles of total weight at most $\varepsilon \cdot w(\mathcal{R})$. When solving the subproblems given by the faces of the partition we again lose rectangles of total weight at most $\varepsilon \cdot w(\mathcal{R})$. Thus, by choosing $k := (1/\varepsilon)(1/\delta)^{O(1)}$, GEO-DP yields a PTAS.

Theorem 22. *Let $\varepsilon > 0$ and $\delta > 0$ be constants. The algorithm GEO-DP parametrized by $k = \frac{1}{\varepsilon}(\frac{1}{\delta})^{O(1)}$ is a polynomial time $(1+\varepsilon)$-approximation algorithm for the maximum weight independent set of rectangles problem for instances with only δ-large rectangles.*

Using standard shifting technique arguments we obtain the following corollary.

Corollary 23. *Let $\varepsilon > 0$ and $\delta > 0$ be constants. The algorithm GEO-DP parametrized by $k = (\frac{1}{\varepsilon \cdot \delta})^{O(1)}$ is a polynomial time $(1+\varepsilon)$-approximation algorithm for instances of MWISR where for all rectangles $R_i, R_{i'}$ it holds that $\max\{h_i, g_i\} \leq (1/\delta) \cdot \max\{h_{i'}, g_{i'}\}$.*

Acknowledgements: We would like to thank Thomas Erlebach, Danny Hermelin, and Erik Jan van Leeuwen for helpful discussions.

REFERENCES

[1] A. Adamaszek and A. Wiese, "Approximation schemes for maximum weight independent set of rectangles," *ArXiv e-prints*, vol. abs/1307.1774, 2013.

[2] P. K. Agarwal, M. van Kreveld, and S. Suri, "Label placement by maximum independent set in rectangles," *Computational Geometry*, vol. 11, pp. 209–218, 1998.

[3] P. K. Agarwal and N. H. Mustafa, "Independent set of intersection graphs of convex objects in 2D," *Computational Geometry*, vol. 34, no. 2, pp. 83–95, 2006.

[4] S. Arora, M. Grigni, D. Karger, P. Klein, and A. Woloszyn, "A polynomial-time approximation scheme for weighted planar graph TSP," in *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '98)*. SIAM, 1998, pp. 33–41.

[5] P. Berman, B. DasGupta, S. Muthukrishnan, and S. Ramaswami, "Efficient approximation algorithms for tiling and packing problems with rectangles," *Journal of Algorithms*, vol. 41, no. 2, pp. 443 – 470, 2001.

[6] P. Bonsma, J. Schulz, and A. Wiese, "A constant factor approximation algorithm for unsplittable flow on paths," in *Proceedings of the 52th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2011)*, 2011, pp. 47–56.

[7] P. Chalermsook and J. Chuzhoy, "Maximum independent set of rectangles," in *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '09)*. SIAM, 2009, pp. 892–901.

[8] P. Chalermsook, "Coloring and maximum independent set of rectangles," *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pp. 123–134, 2011.

[9] T. M. Chan, "A note on maximum independent sets in rectangle intersection graphs," *Information Processing Letters*, vol. 89, no. 1, pp. 19–23, 2004.

[10] T. M. Chan and S. Har-Peled, "Approximation algorithms for maximum independent set of pseudo-disks," *Discrete & Computational Geometry*, vol. 48, no. 2, pp. 373–392, 2012.

[11] J. S. Doerschler and H. Freeman, "A rule-based system for dense-map name placement," *Communications of the ACM*, vol. 35, no. 1, pp. 68–79, 1992.

[12] T. Erlebach, K. Jansen, and E. Seidel, "Polynomial-time approximation schemes for geometric intersection graphs," *SIAM Journal on Computing*, vol. 34, no. 6, pp. 1302–1323, 2005.

[13] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto, "Optimal packing and covering in the plane are np-complete," *Information processing letters*, vol. 12, no. 3, pp. 133–137, 1981.

[14] J. Fox and J. Pach, "Computing the independence number of intersection graphs," in *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '11)*. SIAM, 2011, pp. 1161–1165.

[15] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama, "Data mining with optimized two-dimensional association rules," *ACM Transactions on Database Systems (TODS)*, vol. 26, no. 2, pp. 179–213, 2001.

[16] H. Imai and T. Asano, "Finding the connected components and a maximum clique of an intersection graph of rectangles in the plane," *Journal of algorithms*, vol. 4, no. 4, pp. 310–323, 1983.

[17] S. Khanna, S. Muthukrishnan, and M. Paterson, "On approximating rectangle tiling and packing," in *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '98)*. SIAM, 1998, pp. 384–393.

[18] B. Lent, A. Swami, and J. Widom, "Clustering association rules," in *Proceedings of the 13th International Conference on Data Engineering*. IEEE, 1997, pp. 220–231.

[19] L. Lewin-Eytan, J. Naor, and A. Orda, "Routing and admission control in networks with advance reservations," *Approximation Algorithms for Combinatorial Optimization*, pp. 215–228, 2002.

[20] F. Nielsen, "Fast stabbing of boxes in high dimensions," *Theor. Comp. Sc.*, vol. 246, pp. 53 – 72, 2000.

[21] J. Soto, personal communication.

[22] D. Zuckerman, "Linear degree extractors and the inapproximability of max clique and chromatic number," *Theory of Computing*, vol. 3, pp. 103–128, 2007.