Maximum Independent Set of Rectangles

Hugo Jacob

Supervised by William Lochet and Christophe Paul

General context

The Maximum Independent Set (MIS) problem is amongst the most classical graph problems, it consists in finding a set of vertices that are pairwise nonadjacent of maximum size. This problem has long been known to be hard: it is one of Karp's 21 NP-hard problems, it is also W[1]-complete which roughly means that even if we search for a small set, there should not be any algorithm that is significantly better than trying all sets of such size. It is also hard to approximate in polynomial time: there should not be any algorithm that approximates the optimal solution within a factor of $n^{1-\varepsilon}$ for any $\varepsilon > 0$.

Due to this hardness, it is natural to look for settings in which the problems becomes more tractable. We focus on a setting where our graph is the intersection graph of geometric objects (i.e. there is a vertex for each object and vertices are adjacent when their corresponding objects intersect). Even in geometric settings, the Maximum Independent Set (MIS) problem remains NP-hard. This motivates the study of approximation algorithms. For planar graphs, efficient $(1+\varepsilon)$-approximation algorithms are known [Bak94]. For intersection graphs of so-called 'fat' objects, we also know $(1+\varepsilon)$-approximation algorithms [EJS05]. However, in the case of (axis-aligned) rectangles, even constant approximation in polynomial time was open until recently. Still, it is believed that polynomial $(1+\varepsilon)$-approximation algorithms should exist for rectangles since quasi-polynomial algorithms are known [AW13].

A related open problem is whether rectangle intersection graphs are linearly χ-bounded, i.e. whether than can be properly colored using $O(\omega)$ colors where ω is the clique number. A positive answer would imply a constant factor approximation algorithm for the weighted setting (i.e. where rectangles have weights and we want to maximize the total weight of a set of non intersecting rectangles), see [CW21].

Studied problem

The recent constant approximation algorithm of Mitchell [Mit21] and the following $(2+\varepsilon)$-approximation algorithm of Gálvez et al. [GKM+21] have made significant progress on the question. However, for the special case of axis-aligned segments, nothing is known besides the following straightforward 2-approximation: solve optimally when restricting to segments with the same direction for each direction, and pick the best direction. This suggests studying this subproblem as a first step towards making progress for rectangles.

A common way of obtaining approximation algorithms is to use Linear Programming (or Semi-Definite Programming) relaxations, which can both be solved in polynomial time, and

then to round the solution to obtain a reasonable integral solution. The standard way of formulating MIS with an LP is to bound the total weight in each maximal clique by 1. This uses the fact that there is a polynomial amount of maximal cliques in rectangle intersection and that they can easily be computed.

For approximation algorithms via relaxations, it is natural to look at the so-called integrality gap, the worst case ratio between the value of an optimal solution to the relaxation and an optimal integral solution. It was recently shown that the integrality gap for segment intersection graphs is in fact exactly 2, even if the graph is triangle-free [CCPW22]. This hints that a simple rounding strategy will most likely not be enough to improve on the 2-approximation.

Contribution

We first give a short survey of cases that can be dealt with using known techniques.

We improve slightly on known parameterized algorithms for MIS in segment intersection graphs.

We explore some properties of MIS and its LP relaxation on triangle-free axis-aligned segment intersection graphs, in the hope of finding techniques that allow us to improve on the approximation ratio of 2.

Perspectives

This internship was an opportunity to discover techniques used in approximation algorithms for geometric problems and get a good understanding of the challenges to overcome in order to find a polynomial $(1 + \varepsilon)$-approximation for MISR. The problem of finding such an algorithm remains wide open, although some ideas presented in this report might lead to some improvements on the approximation ratio. These ideas would need to be further investigated.

Meta-information

During the internship, I gave a seminar on the complexity of computing a tree-partition. I participated to two week-long workshops: one on twin-width, and one on directed graphs. A significant amount of time was spent on unrelated projects such as investigating the structure of graphs of twin-width 1.

1 Introduction

Computational geometry is an important field of theoretical computer science which is devoted to the study of algorithms that solve problems from geometry. The topic of this internship is part of the combinatorial branch of this field. Since we are considering an NP-hard combinatorial problem, we consider two standard approaches for such problems: approximation algorithms, i.e. algorithms that give a solution which is provably close to the optimum solution, and parameterized algorithms, most of the time algorithms whose exponential (or worse!) depency is not on the size of the input but another smaller parameter. See Section 2 for definitions.

The maximum independent set problem (MIS) is a standard NP-complete problem (one of Karp's 21 problems). It remains NP-complete even on bounded degree planar graphs. On the other hand, while a constant factor approximation for MIS in the general setting would imply P=NP, approximation is easy in planar graphs.

Consider the following greedy strategy: always pick a vertex of minimum degree and remove its closed neighbourhood. We recall that planar graphs are 5-degenerate since the average degree is less than 6 (corollary of Euler's formula relating the number of faces, edges and vertices of a planar graph), which means that all vertices we picked had at most degree 5. This gives an independent set of size at least $n/6$, a constant factor approximation, and can be implemented in linear time.

In fact, we can obtain $(1+\varepsilon)$-approximation algorithms by Baker's shifting technique [Bak94]. Pick an arbitrary vertex v, and partition vertices according to their distance to $v \mod \ell$. For each part, try removing it from the graph, the remaining graph has a nice structure that allows for efficient computations. More precisely, it has treewidth $O(\ell)$, hence MIS can be solved exactly by dynamic programming in time $f(\ell) \cdot n$. We keep the best solution among all computed solutions. In the worst case, we removed $1/\ell$ from the solution. Setting ℓ appropriately, we obtain the claimed approximation ratio. This technique can be generalized to minor-free graph classes [DHK11].

Planar graphs are sparse, which motivates studying other geometric graphs that allow for denser structures such as intersection graphs of simple objects. For 'fat' objects including squares and disks, a variant of the shifting technique can also be applied [EJS05].

For rectangles, a QPTAS is known [AW13] which suggests that the problem should admit a PTAS, and a $(2+\varepsilon)$-approximation algorithm was recently found [GKM+21]. For unit height rectangles, a PTAS is known [AvKS98], it can be roughly described as applying Baker's shifting technique on one dimension, and then solving the problem by dynamic programming on thin fractions of the plane (we give a more detailed description of a slightly more general result in Section 3.2). Another case which is known to admit a PTAS is the case of rectangles for which all larger sides are within a constant factor of each other [AW13], a brief explanation is given in Section 3.3. For the case of segments (note that this is a particular case of rectangle), a 2-approximation algorithm is easy to derive as we can solve the instance optimally when restricted to only vertical segments or only horizontal segments. Indeed, such a subinstance is an interval graph for which we can solve both the unweighted case and the weighted case in polynomial time. The unweighted case can be solved by iteratively choosing the first interval by nondecreasing order of their end and removing all intersected intervals. This is sometimes referred to as the 'Earliest Deadline First' algorithm. In the weighted case, we can solve it

by dynamic programming as follows: Maximize the total weight of a partial solution that fits in a prefix of the positions, the transitions consist in extending the prefix with or without adding an interval. Both algorithms can be implemented in quasi-linear time.

The approximation ratios of $2+\varepsilon$ for rectangles and 2 for segments (in both cases we assume our objects to be axis-parallel) suggest focusing on segments before trying to improve on the algorithm for rectangles. In fact, the approximation algorithm for rectangles is also based on some form of enumeration of the two directions. After a section introducing terminology, we briefly explain cases where a known technique can yield a PTAS. We observe that the presence of large bicliques is essentially the main technical obstacle to improving approximations. We then try to make new observations in the case of segments, in the hope of finding new techniques. We slightly improve on state of the art for parameterized algorithms by improving the kernel for segment intersection graphs (note that there is no kernel for rectangles unless W[1]=FPT because the case of squares is already W[1]-hard [Mar05]). Following the idea that the LP relaxation could be a good indicator of which direction to pick in large bicliques, we investigate the structure of LP solutions especially in the triangle-free case.

2 Preliminaries

2.1 Approximation

Definition 1. For a maximization problem, we call *ρ-approximation* a solution such that $x^* \leq \rho \cdot x$ where x is the value of our solution and x^* is the optimal value of a solution.

For a minimization problem, we call *ρ-approximation* a solution such that $x \leq \rho x^*$ where x is the value of our solution and x^* is the optimal value of a solution.

A *ρ-approximation algorithm* is an algorithm that produces a ρ-approximation. ρ is referred to as the *approximation ratio* of the algorithm.

Note that with these definitions, we always have $\rho > 1$ (if the ratio were exactly 1 then the algorithm would always give an exact solution).

Some problems can be approximated to any fixed precision in polynomial time but most likely cannot be solved in polynomial time. We now introduce terminology related to algorithms that can be designed in this case.

Definition 2. An algorithm that provides a $(1+\varepsilon)$-approximation for any fixed positive ε is called an *approximation scheme*.

If it has a running time of $n^{f(\varepsilon)}$ for some function f, we call it a *polynomial time approximation scheme* (PTAS).

If it has a running time of $f(\varepsilon) \cdot n^c$ for some function f and some constant c, we call it an *efficient polynomial time approximation scheme* (EPTAS). If f is polynomial in $1/\varepsilon$, then we call it a *fully polynomial time approximation scheme* (FPTAS).

If it has a running time of $2^{\log(n)^{f(\varepsilon)}}$ for some function f, we call it a *quasi-polynomial time approximation scheme* (QPTAS).

Note that in the terminology of parameterized complexity, a PTAS is an XP algorithm with respect to the parameter ε, and an EPTAS is an FPT algorithm with respect to ε.

Unless NP-hard problems can be solved in quasi-polynomial time, the existence of a QPTAS implies the existence of a PTAS. However this reasoning does not provide an algorithm.

2.2 Graphs

We will only consider simple graphs, i.e. undirected graphs with no loops nor multiple edges. We use standard notations but recall them for clarity. For a graph G, its vertex set is denoted $V(G)$ and its edge set is denoted $E(G)$. We denote by $N(v)$ the open neighbourhood of a vertex, and by $N[v]$ the closed neighbourhood of a vertex v, similarly, $N(X)$ and $N[X]$ correspond to the open and closed neighbourhoods of a vertex set X. Given a graph G and a subset X of its vertices, $G[X]$ denotes the subgraph induced by X, and $G - X$ denotes the subgraph induced by $V(G) \setminus X$.

The complete graph on n vertices is denoted by K_n, and the complete bipartite graph with n vertices in one class and m in the other is denoted by $K_{n,m}$. A clique is a complete (induced) subgraph and a biclique is a complete bipartite induced subgraph. An independent set is an anticomplete induced subgraph. $\alpha(G)$ denotes the size of the largest independent set in G and $\omega(G)$ denotes the size of the largest clique in G. K_3 is usually called *triangle*. A graph G is H-free if it does not contain H as an induced subgraph. A vertex cover is a subset of vertices such that each edge is incident to some vertex of the cover.

A *planar graph* is a graph that can be embedded in the plane. A *string graph* is an intersection graph of arbitrary curves in the plane. A *d-DIR (graph)* is an intersection graph of line segments with d possible directions.

Note that axis-parallel line segment intersection graphs are exactly 2-DIRs since the direction of the axes has no combinatorial effect (as soon as they are not parallel). Furthermore, 1-DIRs are exactly interval graphs.

In a connected graph, a *separator* is a subset of vertices whose removal increases the number of connected components. A *balanced* separator is a separator X of a graph G such that none of the connected components of $G-X$ is of size more than a constant fraction $c < 1$ of $|V(G)|$.

A natural way of decomposing a graph is to recursively find small balanced separators, the *treewidth* of a graph roughly corresponds to a minimized largest separator in such a decomposition. On graphs of bounded treewidth many problems can be solved in polynomial time by dynamic progamming along the decomposition.

2.3 Parameterized complexity

A *parameterized problem* is a language over inputs with an additional parameter. The parameter is usually some form of measure of the input, which ideally is correlated with the hardness of the input. The most common parameter when applicable is the solution size. When the parameter of a problem is not explicitly mentionned, usually the parameter is the solution size.

Similarly to classical complexity, parameterized complexity is based on reductions, with the only additional requirement that the new parameter resulting from a reduction is bounded by a function of the original parameter. Such a reduction is called a *parameterized reduction*.

A common example of the fundamental difference with classical reductions is the relation-

ship between Vertex Cover and Independent Set. In the classical setting, both are equivalent since the complement of a vertex cover is an independent set. However notice that in this case we cannot bound the size of the independent set by the size of the vertex cover. In fact, from a parameterized perspective, the two problems are far from being similarly difficult.

We call *fixed parameter tractable* (FPT) a parameterized algorithm running in time $f(k) \cdot n^c$ where n is the input size, k is the parameter, f is a function, and c a constant. The class of problems that can be solved by FPT algorithms is called FPT. It is stable by FPT reductions (i.e. reductions that run in FPT time).

We call *slicewise polynomial* (XP) a parameterized algorithm running in time $n^{f(k)}$, where n is the input size, k is the parameter, and f is some function. The class of problems that can be solved by XP algorithms is called XP. It is stable by XP reductions.

While Vertex Cover belongs to the class FPT, Independent Set is complete for the class W[1] which is closed by FPT reductions. It is widely believed that FPT $\neq$ W[1] (this assumption implies P $\neq$ NP). Hence, the common way of showing that a problem is not expected to be in FPT is to show W[1]-hardness.

A *kernel* is an algorithm running in polynomial time that produces an equivalent instance of size bounded by the parameter. The class FPT is exactly the class of problems admitting a kernel. However, under reasonable assumptions, not all problems of the class FPT admit a *polynomial kernel* i.e. a kernel producing an equivalent instance of size polynomially bounded by the parameter. Vertex Cover is an example of problem admitting a polynomial kernel.

We use the notation O^* to hide polynomial factors, e.g. for an algorithm running in time $O(2^k n^c)$, we may write that it runs in time $O^*(2^k)$.

2.4 Linear Programming

MISR can be formulated as follows: assign $x \in \{0,1\}^{V(G)}$ such that for every maximal clique C in G, $\sum_{v \in C} x_v \leq 1$ in order to maximize $\sum_{v \in V} x_v$

Indeed, every edge of G is contained in at least one maximal clique (possibly the edge itself) so we can never assign 1 to two endpoints of the same edge.

There are only polynomially many maximal cliques in an axis-parallel rectangle intersection graph because each maximal clique corresponds to some rectangle which is exactly the intersection of the rectangles in the clique. Consequently, there are at most n^4 maximal cliques, where n is the number of rectangles. Unlike in the general case, this ILP formulation can be computed in polynomial time given the rectangle representation.

We denote by α and α^* the integral and fractional maximum independent sets, respectively.

The dual of our LP is the following problem: find a set of points in the plane of minimum size such that each rectangle is hit by at least one point. Formally, it should be a set of cliques. However, note that a clique can always be hit by a single point and that there is always an optimal solution with all points on maximal cliques.

We denote by h and h^* the integral and fractional hitting sets, respectively.

3 Designing a PTAS with known techniques

3.1 Sparse case

If the graph contains no large biclique, we can recursively find sublinear separators. From this we can deduce a PTAS that tries to improve greedily by replacing only a subset of size $f(\varepsilon)$ in the solution.

From Theorems 1.1, 1.2 and 5.6 of [Lee17], we deduce the following:

Theorem 1. For every $t > 0$, every $K_{t,t}$-free string graph has a balanced separator of size $O(\sqrt{nt \log t})$.

From [Fre87], we use the following application of having balanced (weighted) separators of sublinear size. The initial proof is for planar graphs but closer inspection shows that the proof works for any hereditary[1] class of graphs with sublinear separators.

Lemma 1. Assume that we can recursively find balanced separators of size $O(n^{1-\delta})$ for some fixed $\delta > 0$, then we can find a vertex set X of size $O(n/r^{\delta})$ such that connected components of $G - X$ have size $O(r)$ and for each such component C, $|N(C)| = O(r^{1-\delta})$.

Proof. We begin by recursively adding balanced separators to a set X_1, until we obtain an connected component of size $O(r)$. The size of X_1 can be upperbounded as follows:

$$s(n) = O(n^{1-\delta}) + s\left(\alpha n + O(n^{1-\delta)}\right) + s\left((1-\alpha)n + O(n^{1-\delta})\right)$$

This recurrence is dominated by the leaves and solves to $s(n) = O(n/r^{\delta})$

In fact, this is even an upper bound on the number of neighbouring components of $G - X_1$ summed over all vertices of X_1.

We will now recursively add separators to a set X_2 that are balanced with respect to the number of neighbours in $X = X_1 \cup X_2$ until we reach components with $O(r^{1-\delta})$ neighbours. For a component with ℓ neighbours, this process will add at most $O(r^{1-\delta} \times \ell / r^{1-\delta}) = O(\ell)$ vertices to X_2.

Summing over components of $G - X_1$ with $\ell = \Omega(r^{1-\delta})$ neighbours, we get $|X_2| = O(n/r^{\delta})$ because the sum of the number of neighbours over all components of $G - X_1$ is $O(n/r^{\delta})$.

We can deduce $|X| = O(n/r^{\delta})$, and all components of $G - X$ have the desired properties.

□

Corollary 1. For any $K_{t,t}$-free string graph G, we can find a vertex set X of size $O(n/\sqrt{r})$ such that connected components of $G - X$ have size $O(r)$ and for each such component C, $|N(C)| = O(\sqrt{r})$.

Corollary 2. There is an $(1+\varepsilon)$-approximation algorithm for MIS in $K_{t,t}$-free string graphs running in time $n^{O(1/\varepsilon^2)}$.

[1] A class is hereditary if it closed by taking induced subgraphs.

Proof. We consider the local search algorithm that tries to replace any subset of $b = O(1/\varepsilon^2)$ vertices by $b+1$ other vertices. This runs in time $n^{O(1/\varepsilon^2)}$.

Consider the solution L of the local search and an optimal solution S.

If the original graph is $K_{t,t}$-free, the subgraph H induced by $L \cup S$ is also $K_{t,t}$-free.

By applying the previous result, we obtain a set X of vertices of H that has size $O(\varepsilon|S|)$ and disconnects H into $O(n/b)$ connected components of size at most b that have neighbourhoods of size $O(1/\varepsilon)$.

Consider a connected component C of $H - X$.

We have $|S \cap C| \leq |L \cap C| + |L \cap N(C)| \leq |L \cap C| + |N(C)|$.

$|S| \leq |X| + \sum_C |S \cap C| \leq |X| + \sum_C |L \cap C| + |N(C)| \leq |X| + |L| + \sum_C |N(C)|$

$\sum_C |N(C)| = O(n/b * \sqrt{b}) = O(\varepsilon|S|)$ since $n = |L| + |S| \leq 2|S|$

Hence, $|S| \leq |L] + O(\varepsilon|S|)$

□

Since rectangle intersection graphs are a special case of string graphs, we obtain a PTAS for the $K_{t,t}$-free case.

One way of obtaining a PTAS for a more general case could be to prove the existence of separators with much stronger properties than simply being balanced.

3.2 Bounded height ratio

The PTAS of [AvKS98] can easily be generalized from unit height to bounded height ratio. Note however that this case is included in the $K_{t,t}$-free case. We briefly explain it since it has a better dependence on ε.

First, we will cover the plane with a set $\mathcal{L}$ of horizontal lines that hit all the rectangles as follows. We add a line that hits the rectangle with maximal ordinate of its horizontal side with the smaller ordinate, remove all rectangles that are hit by the line and iterate the process until all rectangles are removed. We denote by $\ell_1, \ldots, \ell_p$ the lines (in the order in which they were cosntructed), and denote by R_i the rectangles that were first hit by ℓ_i.

Let δ denote the maximum ratio of height between any two rectangles and let h denote its minimum height, then the height of all rectangles is in the range $[h, \delta h]$. By construction the minimum distance between two lines is h, and a rectangle that is hit by a line $\ell \in \mathcal{L}$ cannot be hit by the $\lceil\delta\rceil$-th line below ℓ not the $\lceil\delta\rceil$-th line above ℓ.

Let $k = \lceil\delta\rceil \cdot \lceil 1/\varepsilon\rceil$. We now merge the sets R_i by the class of their index modulo k. For each $j \in [k]$, we will solve MISR exactly by dynamic programming on the instance where we removed the rectangles from $R_j, \ldots, R_{(j+\lceil\delta\rceil-1) \mod k}$. Then there will exist a $j \in [k]$ such that at most $1/\varepsilon$ rectangles of an optimal solution are part of $R_j, \ldots, R_{(j+\lceil\delta\rceil-1) \mod k}$. Hence, we will obtain a solution at least as good as $(1 - 1/\varepsilon)\alpha(G)$.

By construction, the subinstances on which we compute an MIS are made of several disconnected parts, each of which is covered by at most $k - \lceil\delta\rceil$ lines of $\mathcal{L}$. Each disconnected part can be solved by the following dynamic programming: for a given rectilinear polyline with at most $2k$ points, compute the maximum solution that fits to the left of the polyline.

Since there are only n relevant abscissae, this can be done in time $O(n^{2k+1})$.

3.3 Bounded longest side ratio

In the case of an instance where the length of the longest side of each rectangle is within a constant ratio, Adamaszek and Wiese provide a PTAS in [AW13] that we briefly describe in this section. It might be more intuitive to remember that there is a PTAS for unit segments which can then be generalised by allowing different lengths within a bounded ratio and by giving the segments some width smaller than their length.

We can first use a shifting technique if the typical long side of rectangles is small relatively to the size of the instance. Using this we are reduced to the case where all rectangles have their longest side of the same magnitude as the dimension of the instance. Intuitively this means that all rectangles are long. Let k be some constant depending on the maximum ratio between longest sides of rectangles. We can divide the plane into a $k \times k$ grid such that no two endpoints of a longer side of some rectangle are in the same cell. It is then possible to find an approximate solution by dynamic programming on polygonal regions of bounded complexity. Intuitively, this is because if the border of our partial solution crosses some cell too many times it must be possible to make the border shorter by crossing a small enough amount of rectangles in this cell.

4 Segment graphs

4.1 Improved kernel

We improve on the results of Kára and Kratochvil [KK06] who give kernels for MIS on segment intersection graphs and deduce FPT algorithms running in time $O^*(k^{6k})$ for 2-DIR, and $O^*(d^{2k}k^{9dk})$ for d-DIR. However, their algorithms do not require the graph representation as segments unlike ours.

Lemma 2. Given a set $\mathcal{S}$ of line segments with d directions, we can obtain a kernel of size $d(k-1)(1+(d-1)(k-1))$ for k-IS.

Proof. First note that if the restriction of $\mathcal{S}$ to some direction already gives a k-IS we can already answer in $n \log n$ time. Otherwise, we conclude that for each direction there are at most $k-1$ lines containing segments and that $\sum_i \ell_{i,j} \leq k-1$ where $\ell_{i,j}$ is the size of a maximum independent set on the i-th line in direction j.

For each direction, each of its lines is intersected by at most $(d-1)(k-1)$ other lines. We can thus compute greedy independent sets on the line, starting from the beginning of the line and from each of the intersection points. By property of the greedy algorithm, all its prefixes are also optimal. In particular, for any open interval between intersection points or between an intersection point and one end of the line, the union of our greedy independent sets contains an optimal solution.

Summing over all lines and directions, we have that the number of kept segments at most:

$$\sum_{i,j} \ell_{i,j}(1+(d-1)(k-1)) \leq d(k-1)(1+(d-1)(k-1))$$

□

Corollary 3. We can solve k-IS for 2-DIR in time $O^*\left((2\sqrt{ek})^k\right)$, and in time $O^*((ed^2k)^k)$ for d-DIR.

Proof. We first guess the direction containing the most segments of the solution, then guess the segments of the solution that are in other directions (at most $\lfloor (d-1)k/d \rfloor$). We can then solve optimally on the segments of the guessed direction that do not intersect the guessed segments.

This can be done by simple enumeration in time $O^*\left((2\sqrt{ek})^k\right)$ for 2-DIR, and in time $O^*\left(\left(\frac{ed^3k}{d-1}\right)^{\frac{d-1}{d}k}\right) = O^*((ed^2k)^k)$ for d-DIR. The previous expressions are simplified using the upper bound $\binom{n}{k} < (ne/k)^k$. □

It seems unclear whether one can obtain a single exponential algorithm or even a linear kernel. Some time was spent looking for a potential $2^{\Omega(k \log k)}$ lower bound under the ETH by reducing from some of the basic problems introduced in [LMS18] such as $k \times k$ Permutation Independent Set.

4.2 Greedy algorithm and fractional solutions

Observation 1. Given a solution of the LP relaxation, we can always find an equivalent solution that gives positive weights only to segments of our kernel.

Proof. (Sketch) Easy to prove by induction on a sweeping order compatible with the greedy algorithm used for the kernel.

Intuitively, in a solution with weights pushed towards the beginning of lines, each segment with some positive weight cannot have its weight pushed. This means that it is the first segment after some point on the line with total weight 1. Either this point is on a line intersection which guarantees that our segment is in the kernel, or this point is at the end of some other segment of positive weight on the same line which must also be in the kernel and we deduce that our segment must be in the kernel by construction. □

When the integrality gap is high, the naive 2-approximation algorithm becomes good due to the following observation.

Observation 2. If we keep only segments from the direction which is given the most weight by the LP relaxation, we can achieve an integral solution which has at least the same weight.

Since the naive algorithm gives an optimal solution when restricted to a direction, it will always give a solution of size at least $\lceil \alpha^*(G)/2 \rceil$.

In fact, we can be more precise. Let α_g denote the size of the independent set produced by the greedy 'Earliest Deadline First' algorithm applied on the best direction, and h_g denote the hitting set produced by the same algorithm applied to both directions. There is the following sequence of inequalities.

Observation 3.

$$h_g/2 \leq \alpha_g \leq \alpha \leq \alpha^* = h* \leq h \leq h_g$$

As soon as there is a gap of at least a constant fraction between α and h_g, the greedy algorithm performs better than a 2-approximation.

If we can find an approximation algorithm that performs well in the case where the gap is small enough, then we should obtain an approximation algorithm of ratio less than 2 simply by picking the best solution after trying both this algorithm and the greedy algorithm.

We stress the fact that the construction of [CCPW22] giving a lower bound for the integrality gap can be exactly solved by the greedy algorithm.

This suggests looking at the structure of instances that have a small gap.

4.3 Exploring the triangle-free case

Lemma 3 (Half-integrality)**.** On triangle-free instances, there is always an optimal fractional solution with weights in $\{0, 1/2, 1\}$.

Proof. We consider an optimal fractional solution that does not satisfy the property and show how to obtain one satisfying the property.

Let v be some vertex with weight $x \notin \{0, 1/2, 1\}$. We grow a connected component C starting from v by exploring edges with no slack (i.e. the sum of weights of the incident vertices is 1). Let C^+ denote the vertices with weight above $1/2$ and C^- denote the vertices with weight below $1/2$ (note that it partitions C). If $|C^+| \neq |C^-|$, optimality is contradicted since we can find some positive δ such that increasing by δ the weight of vertices in the largest of C^+ and C^- and decreasing by δ the weight of vertices in the smallest of C^+ and C^- does not violate any constraint. If $|C^+| = |C^-|$, we can increase the weights in C^+ by the smallest positive slack ε of constraints applied to C^+ and decrease the weights in C^- by ε, which cannot violate any constraint.

By iteratively applying this transformation to a vertex of the smallest weight above $1/2$, we either strictly decrease the multiplicity of this value as a weight or decrease by 1 the number of weights above 1.

Note that if there is no weight in the range $]1/2, 1[$, all weights must be $0, 1/2$ or 1, otherwise one weight can be increased to $1/2$ contradicting the optimality.

□

Lemma 4. In triangle-free instances, given an optimal fractional solution, there is always an optimal integral solution that respects the integral values of the fractional solution.

Proof. We consider an optimal fractional solution with values in $\{0, 1/2, 1\}$. Let $X_0, X_{1/2}, X_1$ denote the set of vertices with weight $0, 1/2$ and 1 respectively.

Observe that for every $A \subseteq X_0$, $|A| \leq |N(A) \cap X_1|$. Indeed, otherwise setting A and $N(A) \cap X_1$ to $1/2$ would increase total weight without violating constraints.

If S is an optimal integral solution, we replace it by $S' = S \cup X_1 - X_0$.

S' is an independent set because $N(X_1) = X_0$.

Furthermore, $|S \cap X_0| \leq |N(S \cap X_0) \cap X_1|$ by the earlier observation and $S \cap N(S \cap X_0) = \varnothing$ because S is an independent set. We conclude that $|S'| - |S| \geq |N(S \cap X_0) \cap X_1| - |S \cap X_0| \geq 0$.

□

Lemma 5. $\alpha^* - \alpha \leq \tau/2$ where τ is the odd cycle transversal number.

Proof. Consider an optimal fractional solution. Without loss of generality, we can assume that all weights are equal to $1/2$ by the previous lemma.

By removing an odd cycle transversal, we obtain a fractional solution to a bipartite subinstance of total weight $\alpha^* - \tau/2$. We can deduce an integral solution that achieves at least this weight, hence $\alpha^* - \tau/2 \leq \alpha$. □

Unfortunately, it seems unclear how to obtain a lower bound on $\alpha^* - \alpha$. An interesting candidate for a lower bound could be the odd cycle packing number ν due to the following conjecture:

Conjecture 1. The class of triangle-free 2-DIR graphs satisfies a linear Erdős-Pósa property for odd cycles, i.e. for such a graph G, $\tau(G) = \Theta(\nu(G))$.

This result is already known on planar graphs for which the proof heavily relies on reasoning on the faces. We can also reason on faces for triangle-free 2-DIRs (connected components of the plane in the geometric representation).

Observation 4. For a 2-connected triangle-free 2-DIR, there is always an odd face inside an odd cycle.

Observation 5. For a 2-connected triangle-free 2-DIR, the vertex-face incidence graph is the union of two planar graphs.

Note that Conjecture 1 would imply a linear Erdős-Pósa property for odd cycles on the class of 2-DIR without the triangle-free constraint.

Claim 1. Assuming Conjecture 1, we can conclude that the class of 2-DIR graphs satisfies a linear Erdős-Pósa property for odd cycles.

Proof. Let G be a 2-DIR graph.

We always have $\nu(G) \leq \tau(G)$ since all cycles of the packing must be hit.

We now give an upper bound on $\tau(G)$. One way of hitting all odd cycles is to first hit the triangles then hit longer cycles. Let $A \subseteq V(G)$ be a minimum triangle transversal, let $H = G - A$, note that H is a triangle-free induced subgraph of G, and let $B \subseteq V(H)$ be a minimum odd cycle transversal of H, then $A \cup B$ is an odd cycle transversal of G. We deduce the following inequality:

$$\tau(G) \leq \tau_{\triangle}(G) + \tau(H)$$

For triangles, $\tau_\triangle(G) \leq 3\nu_\triangle(G)$, since the vertices of a maximum triangle packing are a triangle transversal. We also have $\tau(H) = O(\nu(H))$ by assumption, and ν is subgraph monotone, so $\tau(H) = O(\nu(G))$. Furthermore, $\nu_\triangle(G) \leq \nu(G)$.

Combining our inequalities:

$$\tau(G) \leq \tau_\triangle(G) + \tau(H) = O(\nu(G))$$

□

For our purposes, Conjecture 1 could be used to have constant factor approximations for OCT in polynomial time, see [GW98] for the case of planar graphs. If, in addition, we had $\alpha^* - \alpha = \Omega(\nu)$, we could obtain a $(2-\varepsilon)$-approximation by using the following win-win argument. If the minimum OCT is of size $O(\varepsilon n)$, we simply find an approximate OCT and deduce an independent set of size $n(1-O(\varepsilon))/2$ which is a good approximation for small enough ε. Otherwise, we simply apply the greedy algorithm but we know that $\alpha^* - \alpha = \Omega(\varepsilon n)$ and we can deduce that it produces a $(2-\varepsilon)$-approximation.

Another interesting perspective is to add constraints to the LP relaxation corresponding to odd cycles. Indeed, we know that the total weight on some odd cycle C_n should be at most $\lfloor n/2 \rfloor$. In the triangle-free case, it seems unlikely that the integrality gap would still be 2 when bounding by 2 the weight on any C_5. Adding constraints on all cycles of length up to $\ell = f(\varepsilon)$ might be a way to make the integrality gap at most $1+\varepsilon$. However, it would still remain unclear whether this can be used to find good integral solutions.

References

[AvKS98] Pankaj K. Agarwal, Marc J. van Kreveld, and Subhash Suri. Label placement by maximum independent set in rectangles. *Comput. Geom.*, 11(3-4):209–218, 1998.

[AW13] Anna Adamaszek and Andreas Wiese. Approximation schemes for maximum weight independent set of rectangles. *CoRR*, abs/1307.1774, 2013.

[Bak94] Brenda S. Baker. Approximation algorithms for np-complete problems on planar graphs. *J. ACM*, 41(1):153–180, 1994.

[CCPW22] Marco Caoduro, Jana Cslovjecsek, Michal Pilipczuk, and Karol Wegrzycki. Independence number of intersection graphs of axis-parallel segments. *CoRR*, abs/2205.15189, 2022.

[CW21] Parinya Chalermsook and Bartosz Walczak. Coloring and maximum weight independent set of rectangles. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 860–868. SIAM, 2021.

[DHK11] Erik D. Demaine, MohammadTaghi Hajiaghayi, and Ken-ichi Kawarabayashi. Contraction decomposition in h-minor-free graphs and algorithmic applications. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 441–450. ACM, 2011.

[EJS05] Thomas Erlebach, Klaus Jansen, and Eike Seidel. Polynomial-time approximation schemes for geometric intersection graphs. *SIAM J. Comput.*, 34(6):1302–1323, 2005.

[Fre87] Greg N. Frederickson. Fast algorithms for shortest paths in planar graphs, with applications. *SIAM J. Comput.*, 16(6):1004–1022, 1987.

[GKM+21] Waldo Gálvez, Arindam Khan, Mathieu Mari, Tobias Mömke, Madhusudhan Reddy, and Andreas Wiese. A $(2 + \epsilon)$-approximation algorithm for maximum independent set of rectangles, 2021.

[GW98] Michel X. Goemans and David P. Williamson. Primal-dual approximation algorithms for feedback problems in planar graphs. *Comb.*, 18(1):37–59, 1998.

[KK06] Jan Kára and Jan Kratochvíl. Fixed parameter tractability of independent set in segment intersection graphs. In Hans L. Bodlaender and Michael A. Langston, editors, *Parameterized and Exact Computation, Second International Workshop, IWPEC 2006, Zürich, Switzerland, September 13-15, 2006, Proceedings*, volume 4169 of *Lecture Notes in Computer Science*, pages 166–174. Springer, 2006.

[Lee17] James R. Lee. Separators in region intersection graphs. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 1:1–1:8. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[LMS18] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Slightly superexponential parameterized problems. *SIAM J. Comput.*, 47(3):675–702, 2018.

[Mar05] Dániel Marx. Efficient approximation schemes for geometric problems? In Gerth Stølting Brodal and Stefano Leonardi, editors, *Algorithms - ESA 2005, 13th Annual European Symposium, Palma de Mallorca, Spain, October 3-6, 2005, Proceedings*, volume 3669 of *Lecture Notes in Computer Science*, pages 448–459. Springer, 2005.

[Mit21] Joseph S. B. Mitchell. Approximating maximum independent set for rectangles in the plane. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 339–350. IEEE, 2021.