A $(2+\varepsilon)$-approximation for the Maximum independent set of Rectangles.

Mathieu Mari

University of Warsaw

IDEAS NCBR

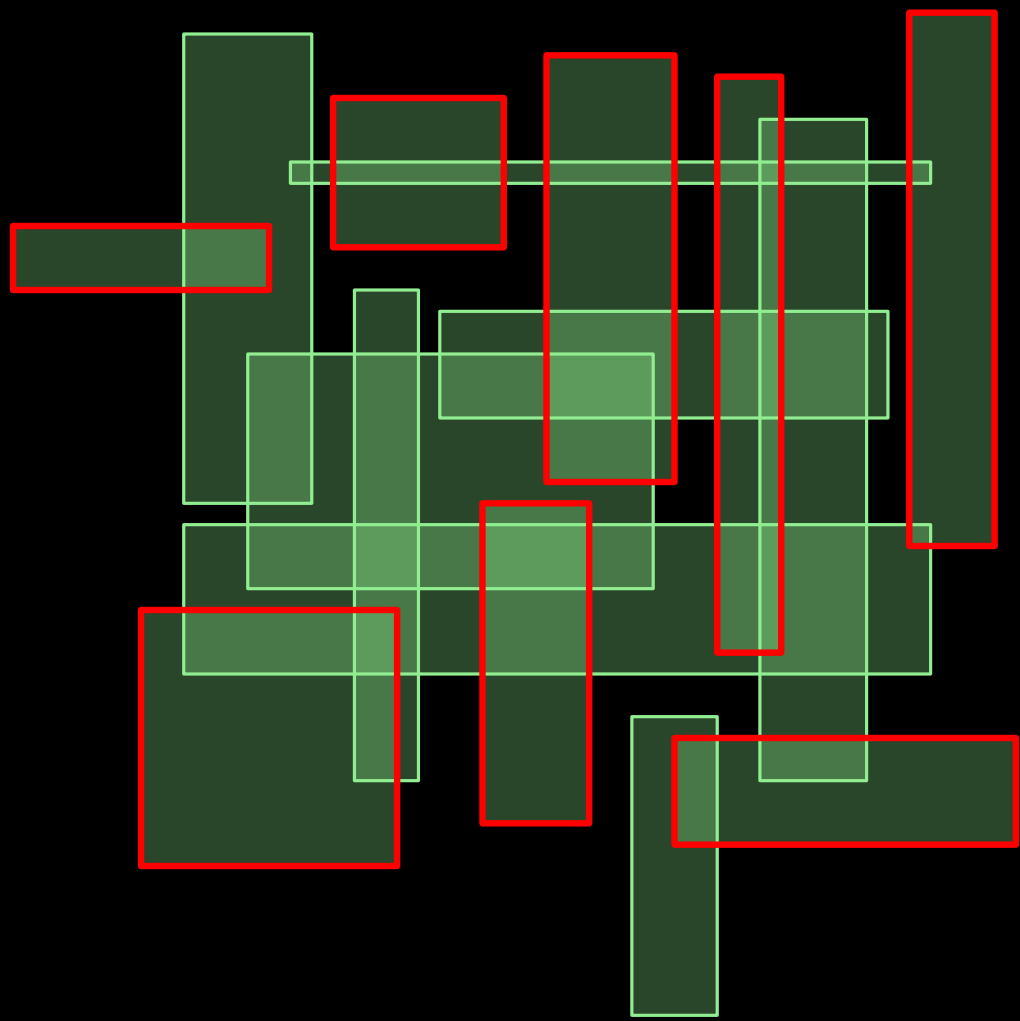
W. Galvez TU Munich

A. Khan IISc Bangalore

M. Reddy IIT KGP $\rightarrow$ CMU

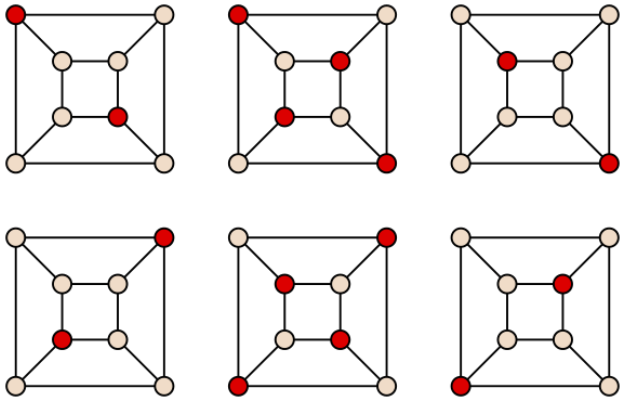
T. Mömke U of Augsburg

A. Weise U Chile

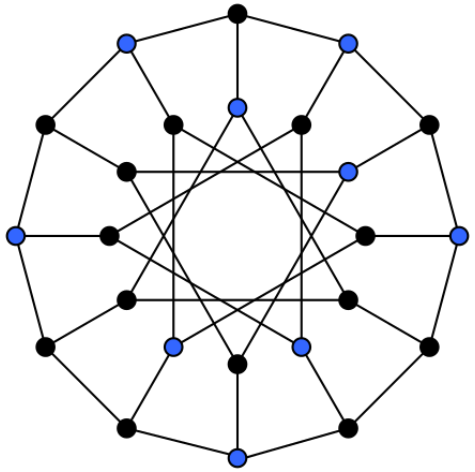


Maximum Independent Set (MIS)

- Independent set is a set S of vertices in a graph such that no two vertices in S are adjacent
- MIS: Find the Maximum cardinality independent set.



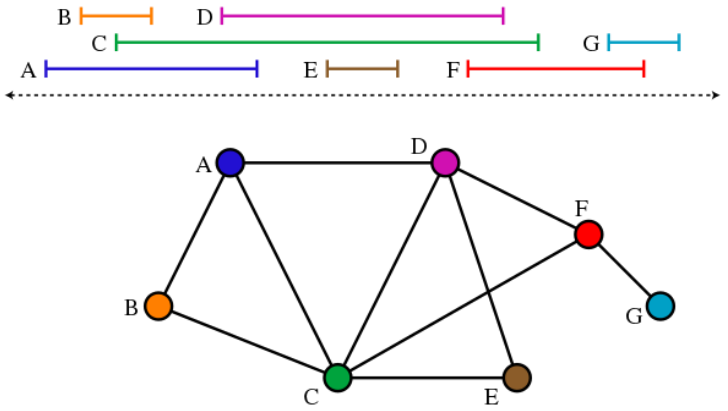
- Classical NP-hard problem
  - Trivial to get n-approximation
  - NP-hard to get $n^{1-\varepsilon}$ approximation. [Håstad, 1999]



Independent sets shown in red and blue

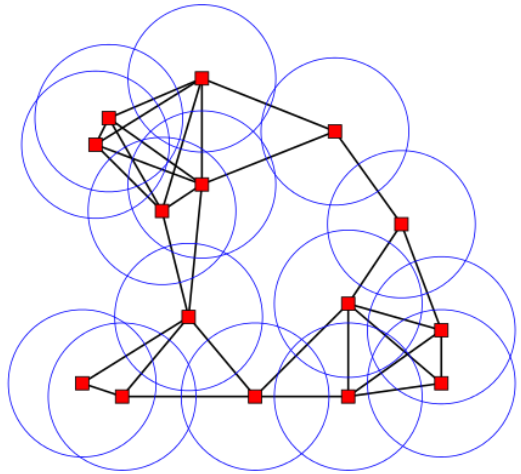
Special Case: Geometric Intersection Graphs

- Nodes correspond to geometric objects like polygons, spheres, … etc.
- There is edge (u, v) iff objects corresponding to u and v overlap



$1D$- Interval Graph

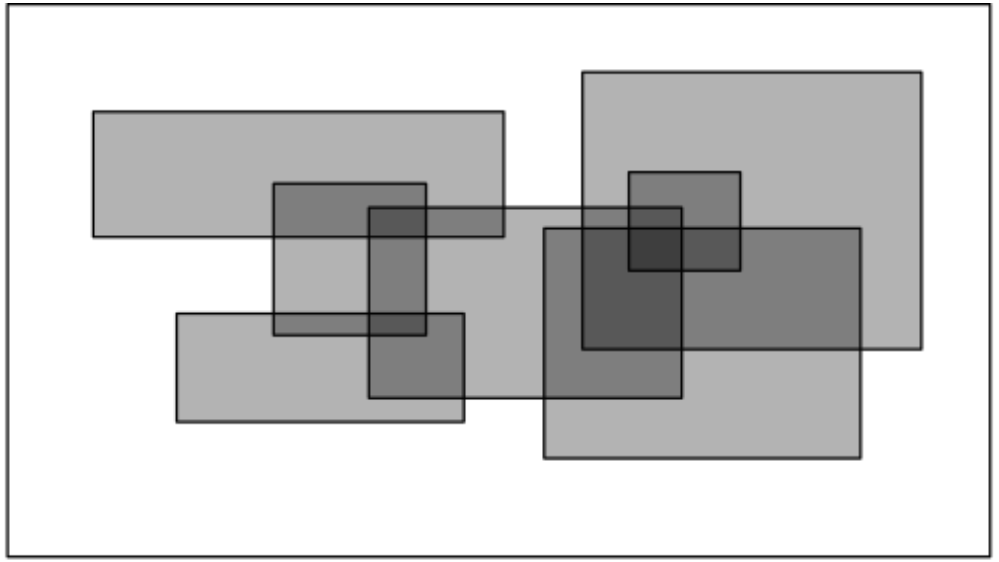
- Polynomial time solvable for intersection graphs of intervals on a line
- PTAS for Squares and Spheres. [Erlebach et al. , SODA '01]
- PTAS for Pseudo-Disks. [Chan, Har-Peled, SOCG '09]

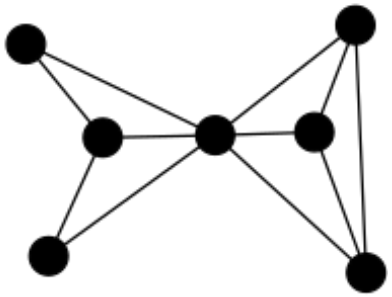


Unit Disk Graph

Our focus: Maximum Independent Set of Rectangles (MISR)

- Input: A set of n axis-parallel rectangles on a plane.
- Goal: Output a set of non-overlapping rectangles of maximum cardinality.

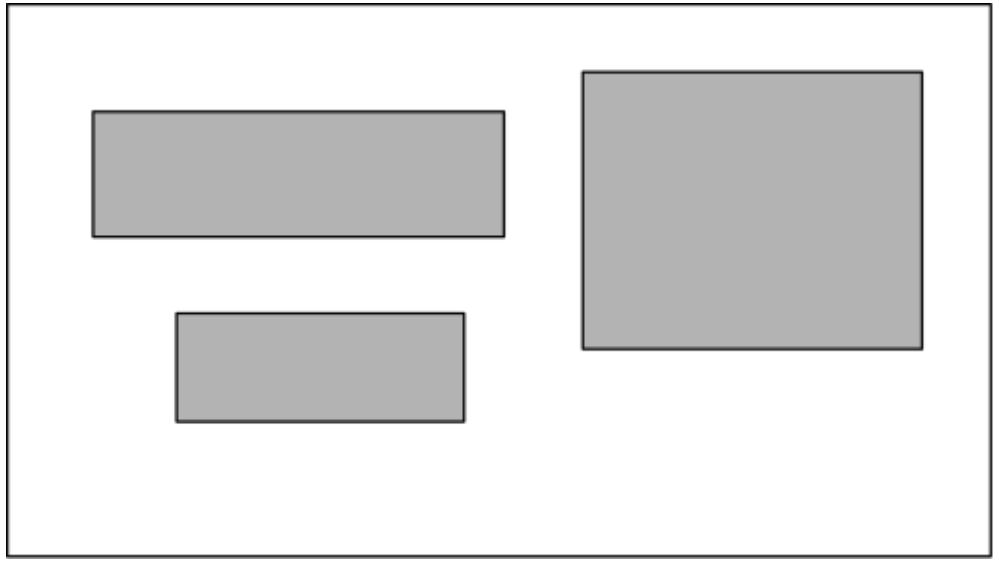


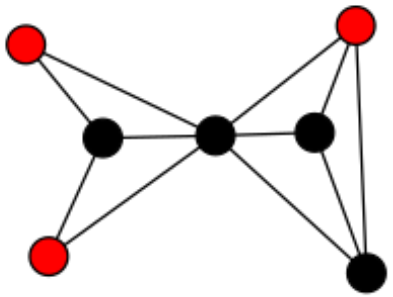


Our focus: Maximum Independent Set of Rectangles (MISR)

- Input: A set of n axis-parallel rectangles on a plane.
- Goal: Output a set of non-overlapping rectangles of maximum cardinality.

⟵ NP-hard. [R J Fowler et al., '81]





MISR: Previous work

* $O(\log n)$ [Agarwal et al, CG '98; …]

* $O(\log \log n)$ [Chalermsook-Chuzhoy, SODA '09]

- QPTAS [Adamaszek-Weise, FOCS '13; Chuzhoy-Ene, FOCS '16]

  ⇒MISR not APX hard unless $NP \subseteq DTIME\left(2^{poly(\log n)}\right)$

  PTAS is expected but even $O(1)$ is not known

* $O(\log \log n)$ for the weighted case [Chalermsook-Walczak, SODA '21]

- 10-approximation [JSB Mitchell, FOCS '21]

MISR: Our Result

A 2+ ε -approximation Algorithm

General framework:

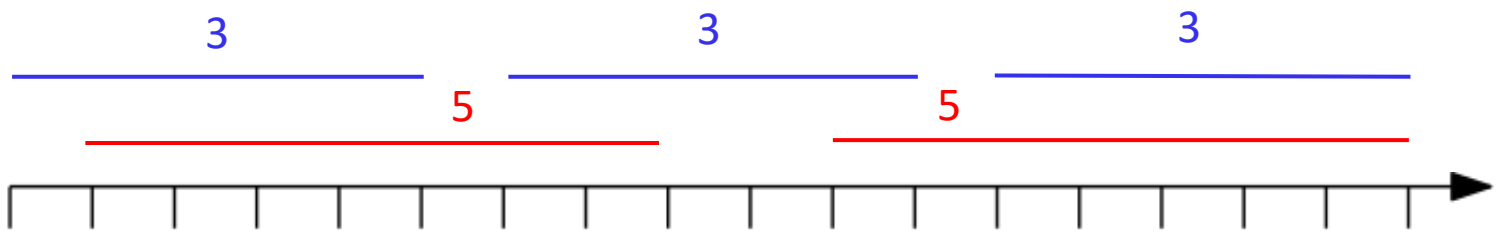
Step 1. Show existence of a "good" structured solution:
- Set of candidate solutions: C. -- size can be exponential.
- Set of structured solution: S. -- need to make the size to be polynomial.
- S is a good approximation of C: For any candidate solution $I \in C$ there is a structured solution $I' \subseteq I$ and $I' \in S$ such that $\alpha|I'| \geq |I|$.

Step 2. DP-based algorithm that finds the best structured solution.

Let us sketch a 6-approximation algorithm

Technique/Idea

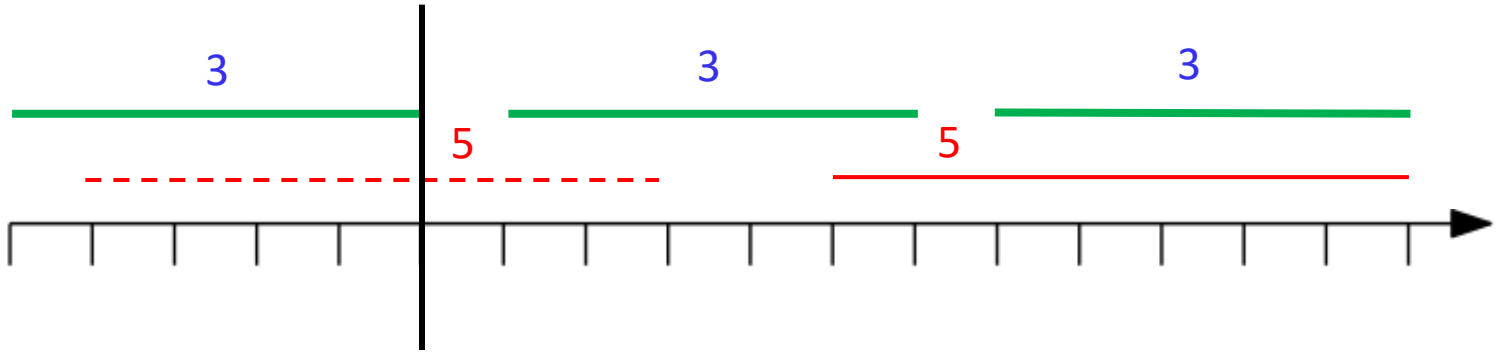
Question: How do you solve Max-Weight indep. set of intervals on a line?



- Divide and Conquer?

Technique/Idea

Question: How do you solve Max-Weight indep. set of intervals on a line?

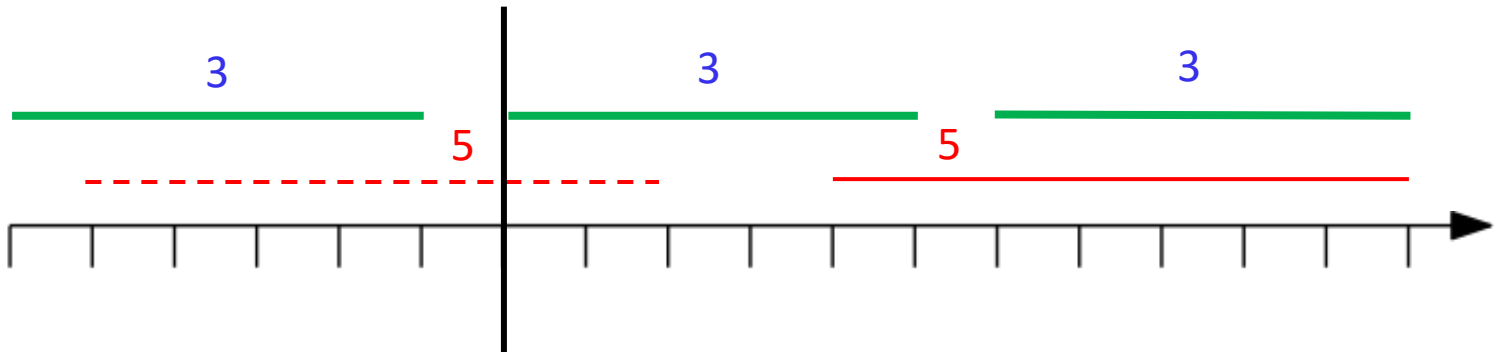


Cut, Solve recursively, Patch

- Divide and Conquer?

Technique/Idea

Question: How do you solve Max-Weight indep. set of intervals on a line?

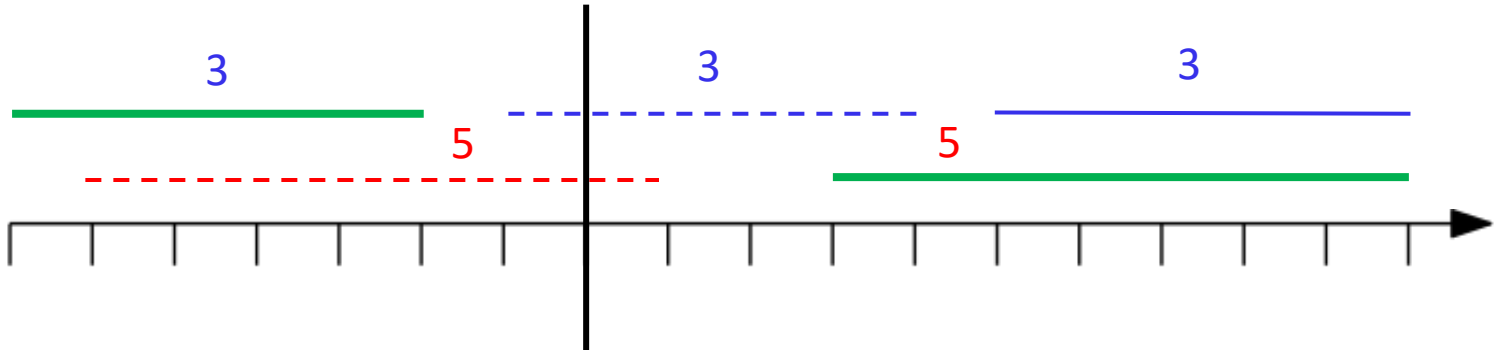


Cut, Solve recursively, Patch

- Divide and Conquer?

Technique/Idea

Question: How do you solve Max-Weight indep. set of intervals on a line?

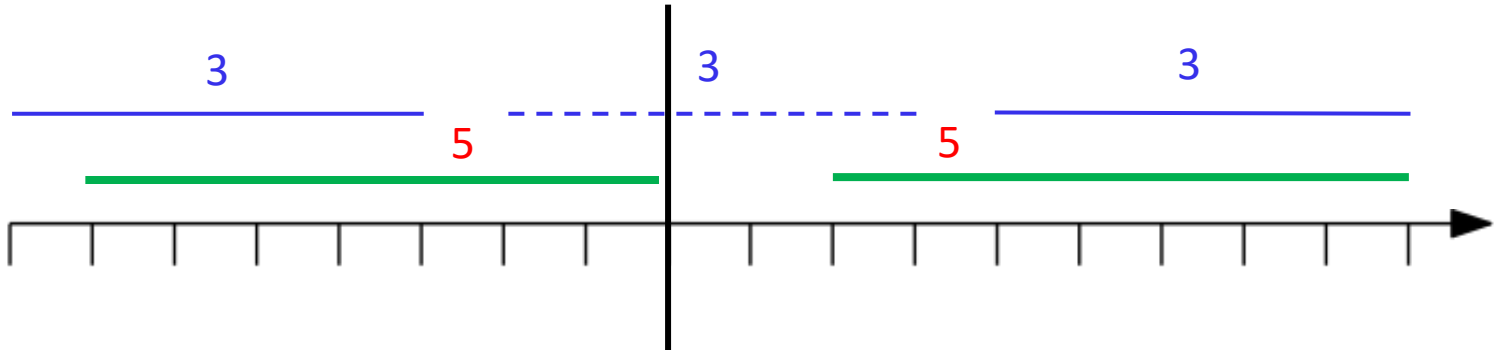


Cut, Solve recursively, Patch

- Divide and Conquer?

Technique/Idea

Question: How do you solve Max-Weight indep. set of intervals on a line?

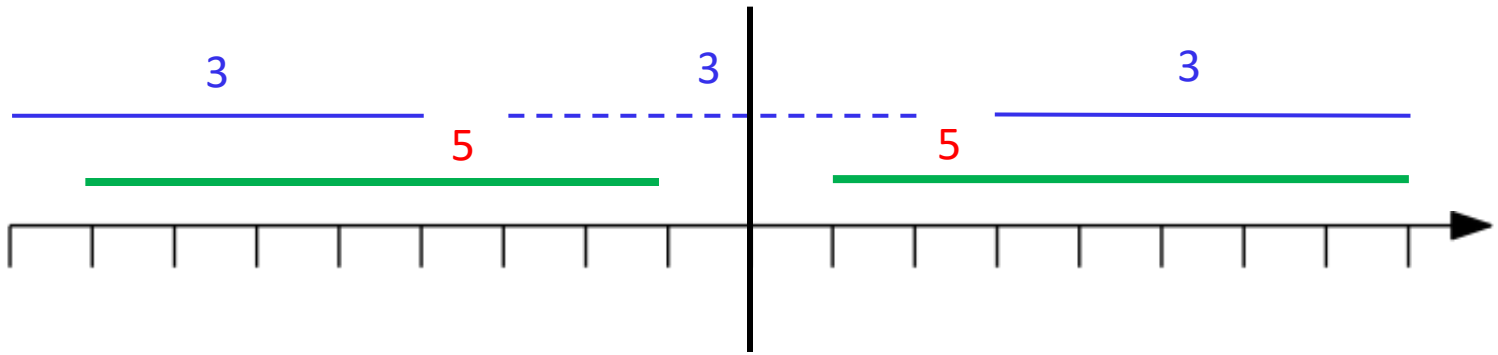


Cut, Solve recursively, Patch

- Divide and Conquer?

Technique/Idea

Question: How do you solve Max-Weight indep. set of intervals on a line?

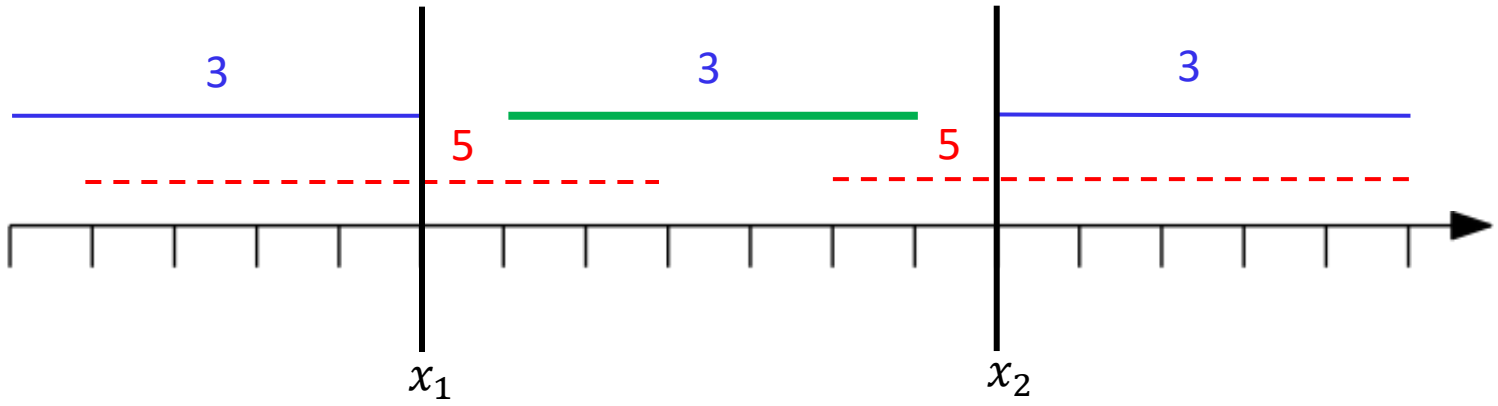


Cut, Solve recursively, Patch

- Divide and Conquer?
- Can be implemented as a dynamic program
- DP$[x_1, x_2]$ contains the optimal solution in the range $[x_1, x_2]$

Technique/Idea

Question: How do you solve Max-Weight indep. set of intervals on a line?

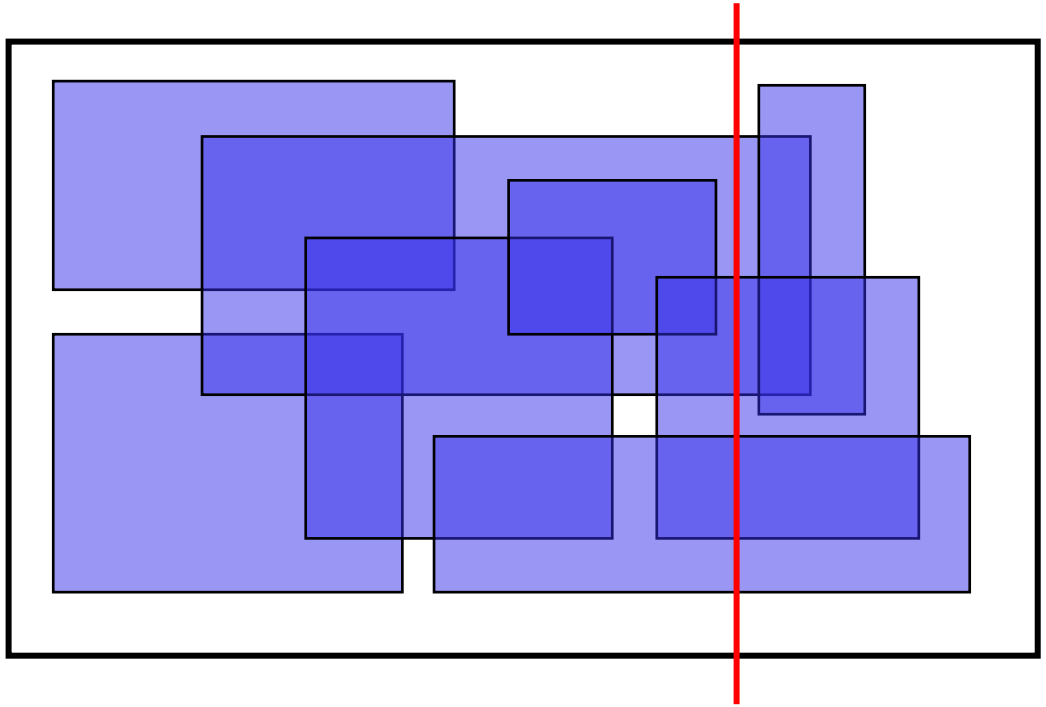


Cut, Solve recursively, Patch

- Divide and Conquer?
- Can be implemented as a dynamic program
- DP$[x_1, x_2]$ contains the optimal solution in the range $[x_1, x_2]$

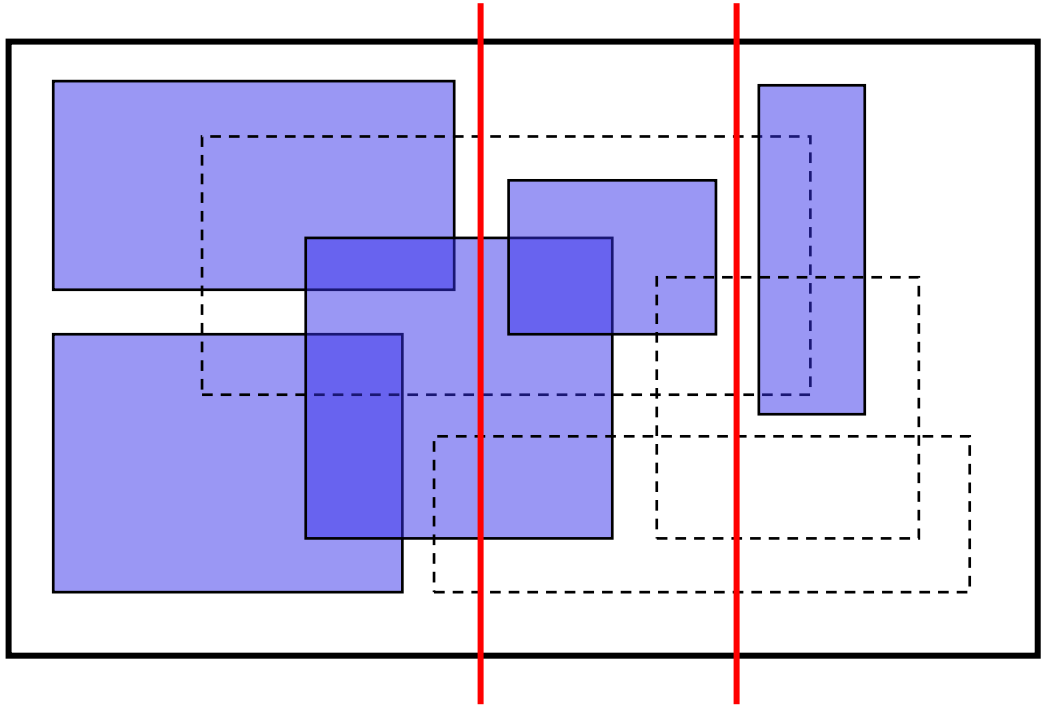
Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR



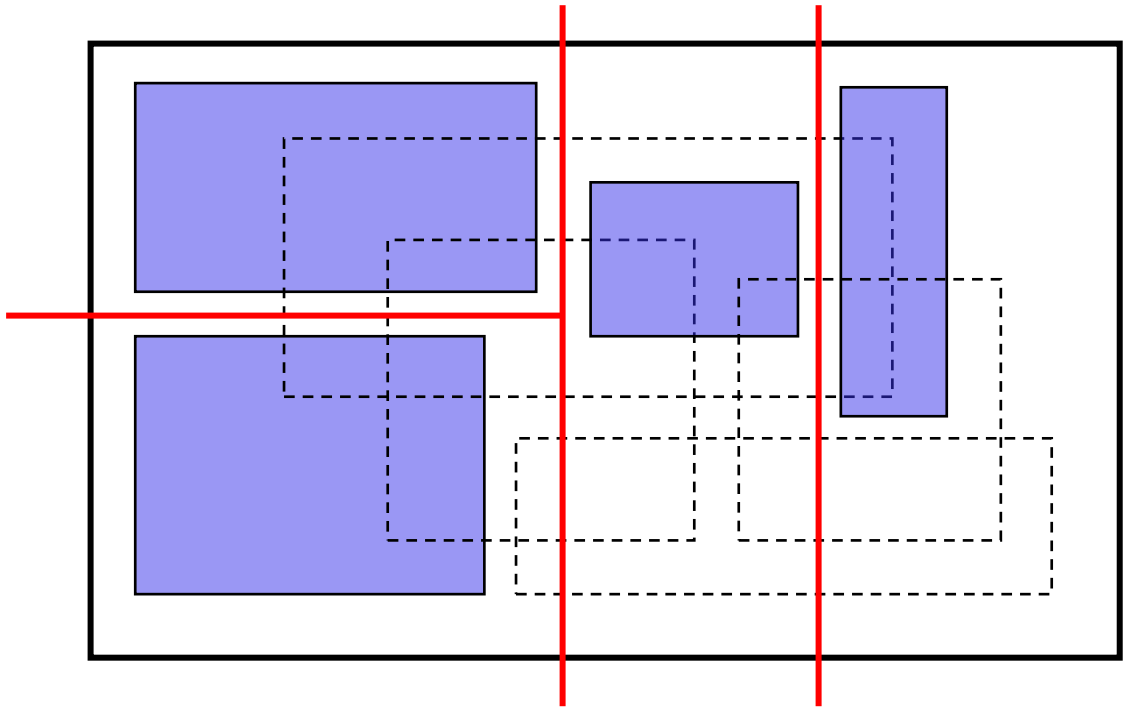
Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR



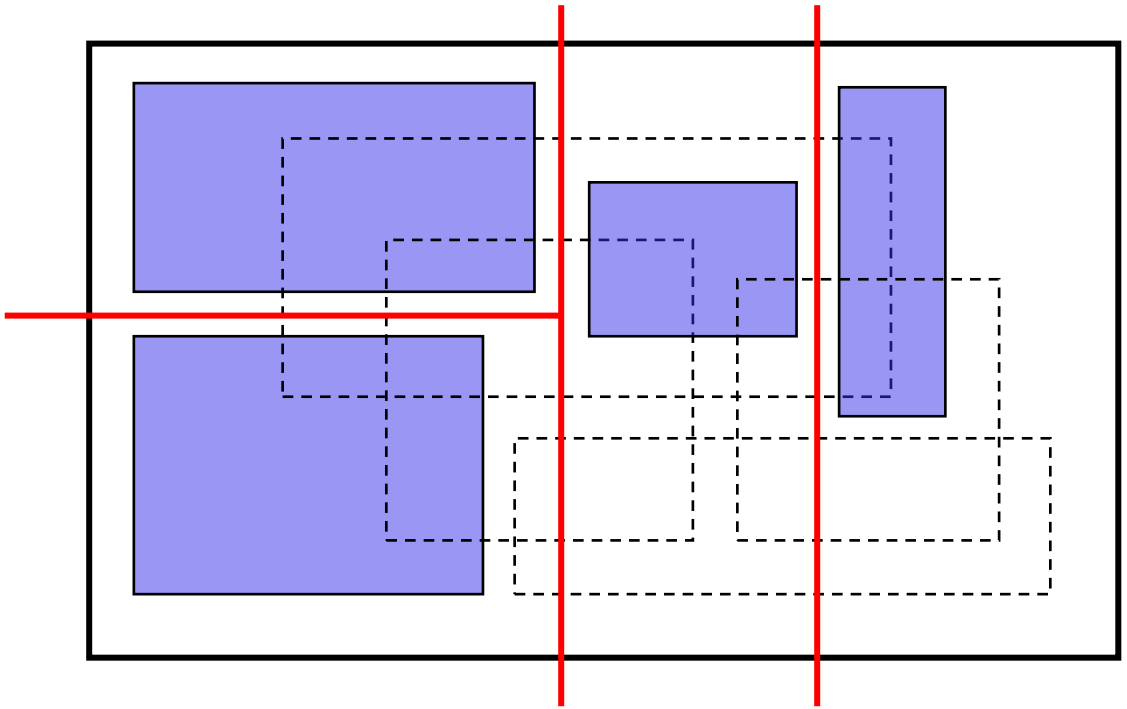
Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR



Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

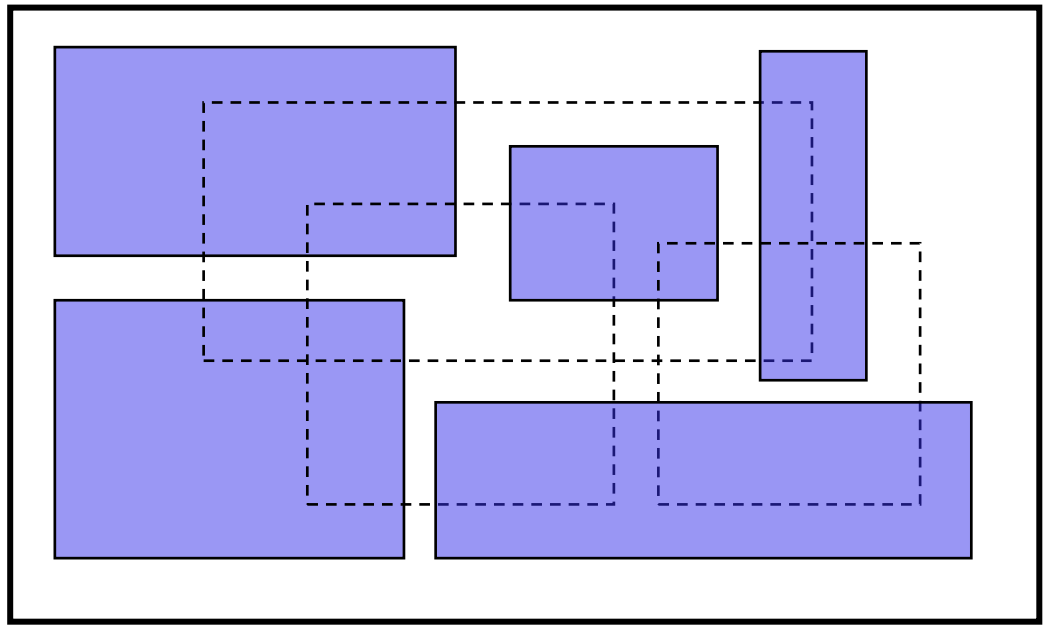


- These end-to-end cuts splitting the piece into two sub-pieces are called Guillotine cuts
- A set of objects in the plane are Guillotine separable if there exists a sequence of guillotine cuts that separates each object into a different piece

Example of non-guillotine separable instance?

Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

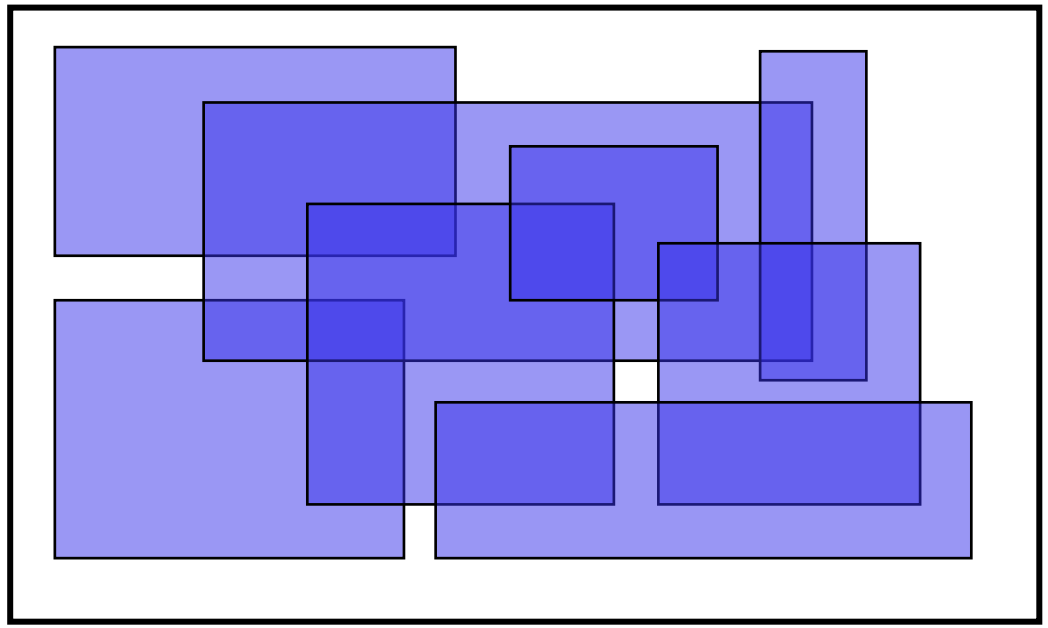


- These end-to-end cuts splitting the piece into two sub-pieces are called Guillotine cuts
- A set of objects in the plane are Guillotine separable if there exists a sequence of guillotine cuts that separates each object into a different piece

Example of non-guillotine separable instance?

Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

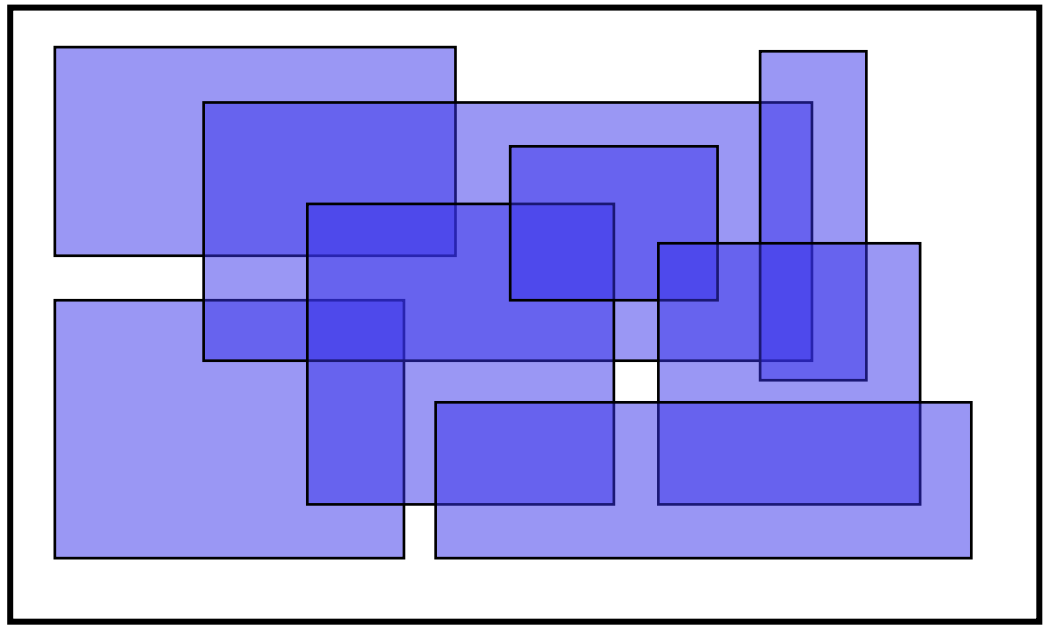


- These end-to-end cuts splitting the piece into two sub-pieces are called Guillotine cuts
- A set of objects in the plane are Guillotine separable if there exists a sequence of guillotine cuts that separates each object into a different piece

Example of non-guillotine separable instance?

Technique/Idea

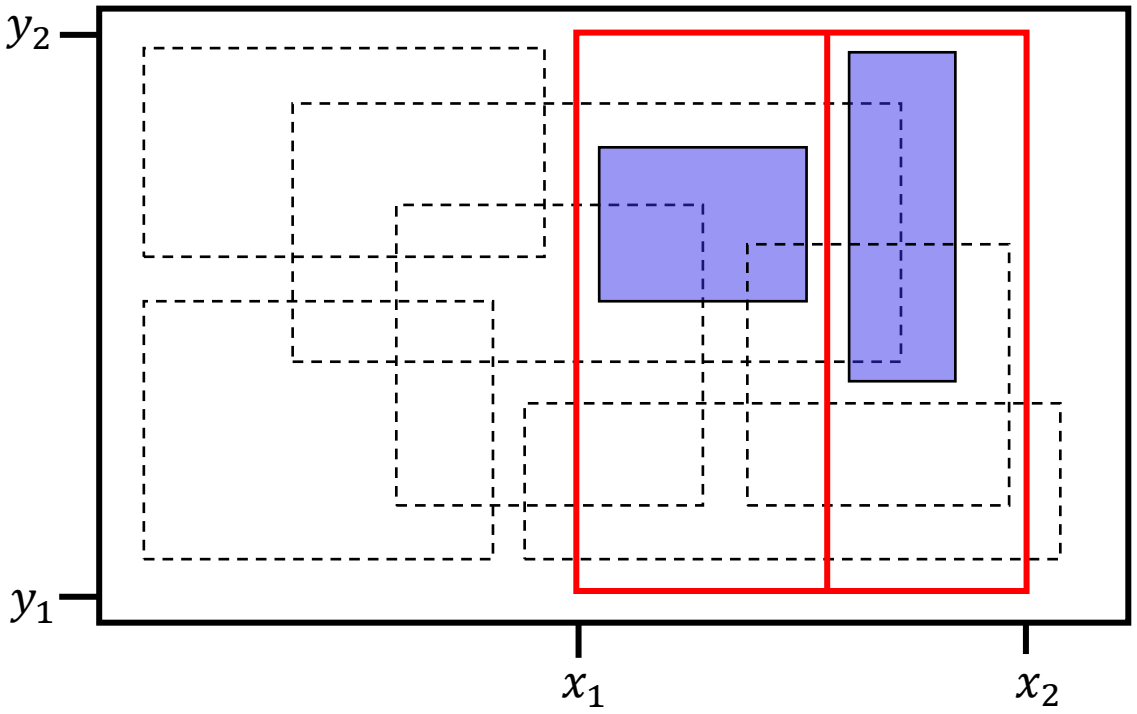
Idea: Generalize and apply the recursive algorithm to MISR



- Fact: Optimal Guillotine separable independent set can be found using a dynamic program.

Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

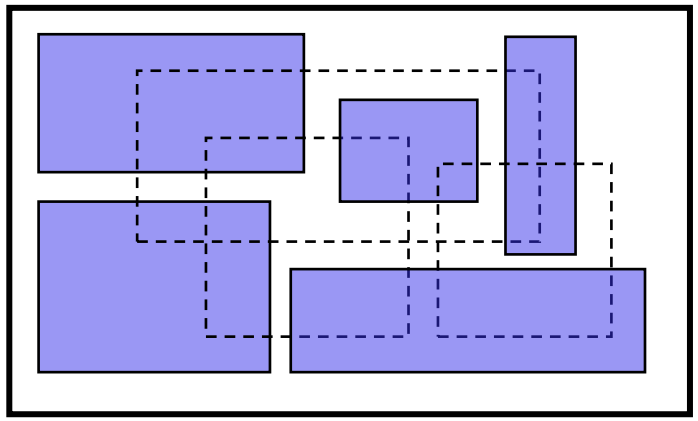


- Fact: Optimal Guillotine separable independent set can be found using a dynamic program.
- $DP[x_1, x_2, y_1, y_2]$ contains the optimal guillotine separable independent set in the window $[x_1, x_2] \times [y_1, y_2]$
- Larger table entry filled by guessing best first guillotine cut and concatenating optimal solutions of two pieces

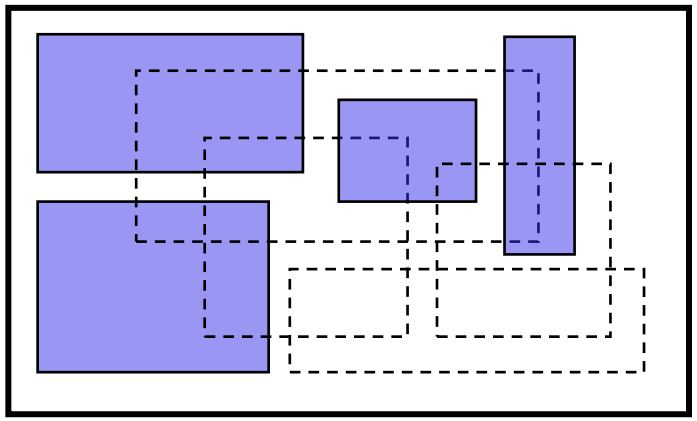
Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

- Question: How good is the Optimal Guillotine separable independent set with respect to the Optimal independent set?
- Conjecture[Pach-Tardos] : Any axis parallel independent set of rectangles has a guillotine separable subset of relative size $\Omega(1)$



Optimal, not guillotine separable

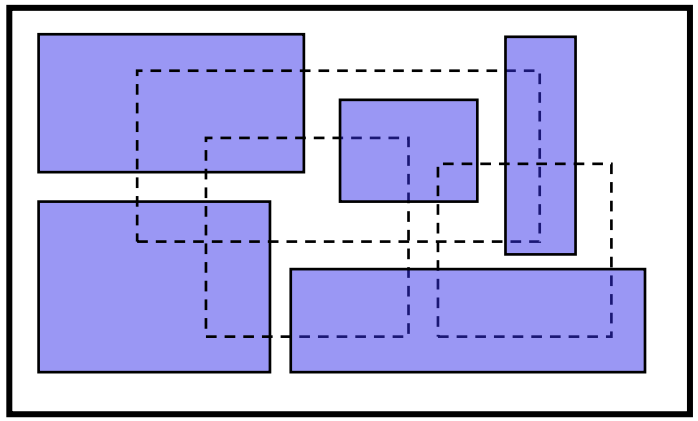


Optimal guillotine separable

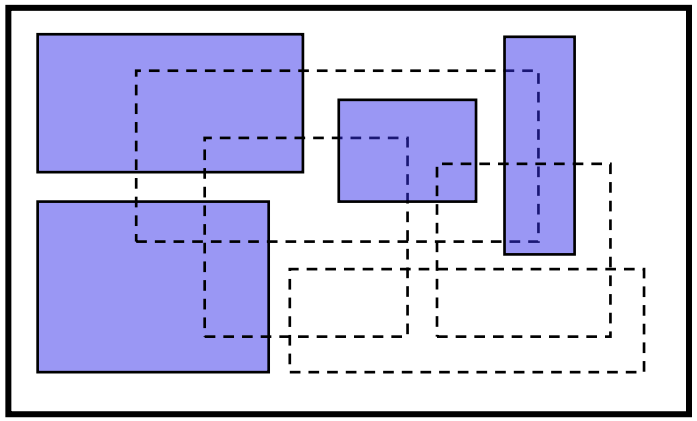
Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

- Question: How good is the Optimal Guillotine separable independent set with respect to the Optimal independent set?
- Conjecture[Madhusudhan] : Any axis parallel independent set of rectangles has a guillotine separable subset of relative size $\frac{1}{2}$ even in the weighted case



Optimal, not guillotine separable

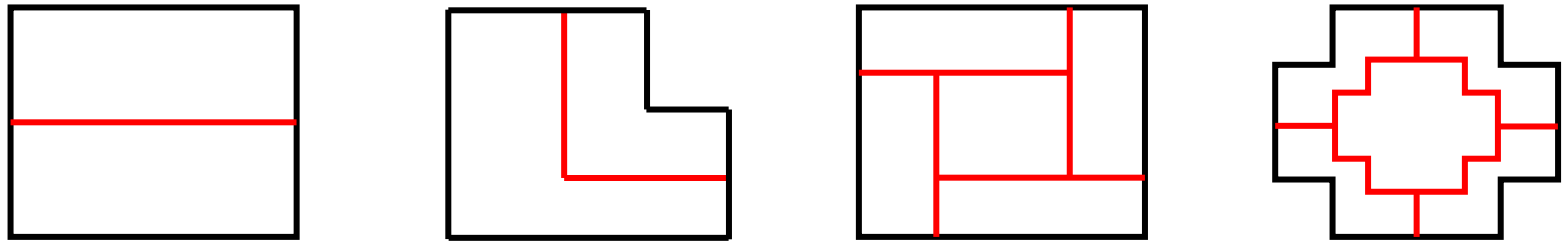


Optimal guillotine separable

Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

- Question: What other structured solutions that generalize guillotine separability can be found optimally in polynomial time ?
- Fact: Optimal (p, c)-separable independent set can be found in $n^{O(pc)}$ time using DP



$(2,4)$ $(2,6)$ $(5,4)$ $(5,12)$

Pieces and cuts can be more general

- A cut is called a (p, c)-cut if it cuts a piece into at most p pieces and each piece has at most c complexity (edges)
- Independent set is (p, c)-separable if a sequence of (p, c)-cuts separate all rectangles without cutting any

Technique/Idea

Idea: Generalize and apply the recursive algorithm to MISR

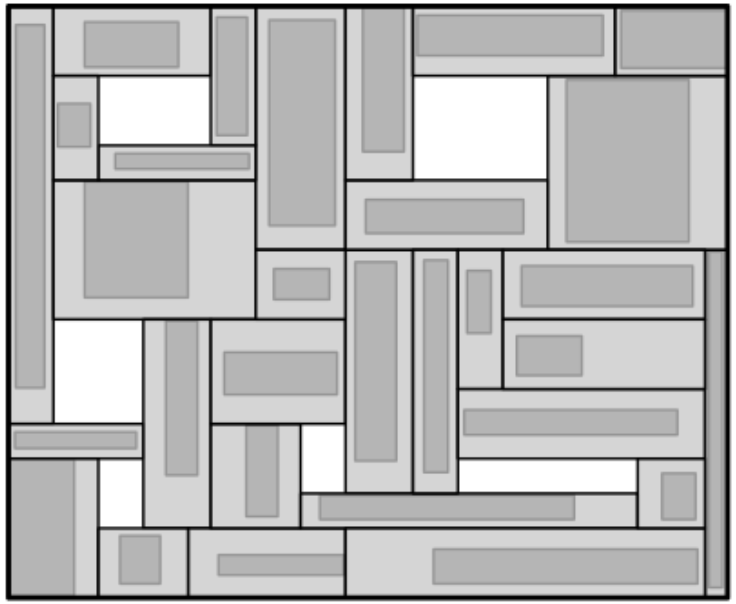
- Question: What other structured solutions that generalize guillotine separability can be found optimally in polynomial time ?
- Fact: Optimal (p, c)-separable independent set can be found in $n^{O(pc)}$ time using DP
- Theorem: For any indep. set $\mathcal{R}$, there exists a $(3,26)$- separable subset $\hat{\mathcal{R}} \subseteq \mathcal{R}$ such that $|\hat{\mathcal{R}}| \geq \frac{1}{6}|\mathcal{R}|$ $\Rightarrow$ 6-approximation
- Full Theorem: For any indep. set $\mathcal{R}$, there exists a $(2, O(\frac{1}{\varepsilon}))$ - separable subset $\hat{\mathcal{R}} \subseteq \mathcal{R}$ such that $|\hat{\mathcal{R}}| \geq \frac{1}{2+\varepsilon}|\mathcal{R}|$ $\Rightarrow$ $(2+\varepsilon)$-approximation
- Existence shown by providing the cuts algorithmically
- Builds on Mitchell's results with additional new ideas

Proof?

- Assume Maximality for $\mathcal{R}$

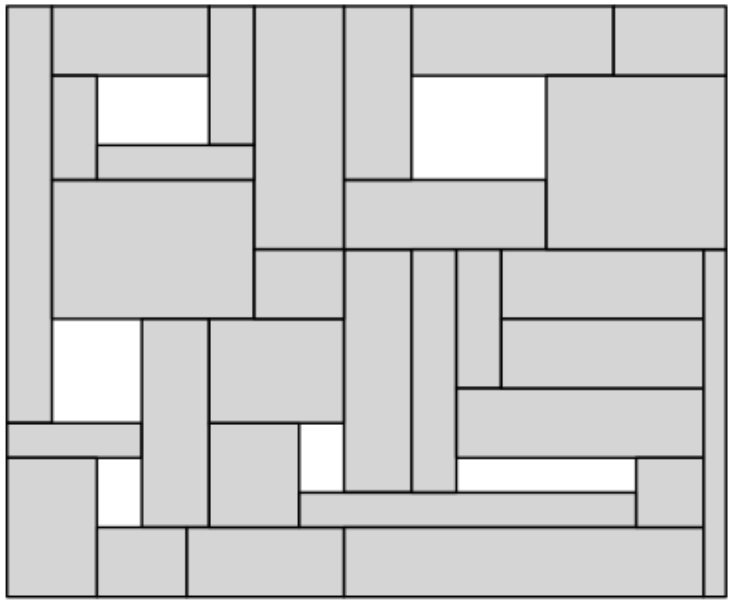
Proof?

- Assume Maximality for $\mathcal{R}$



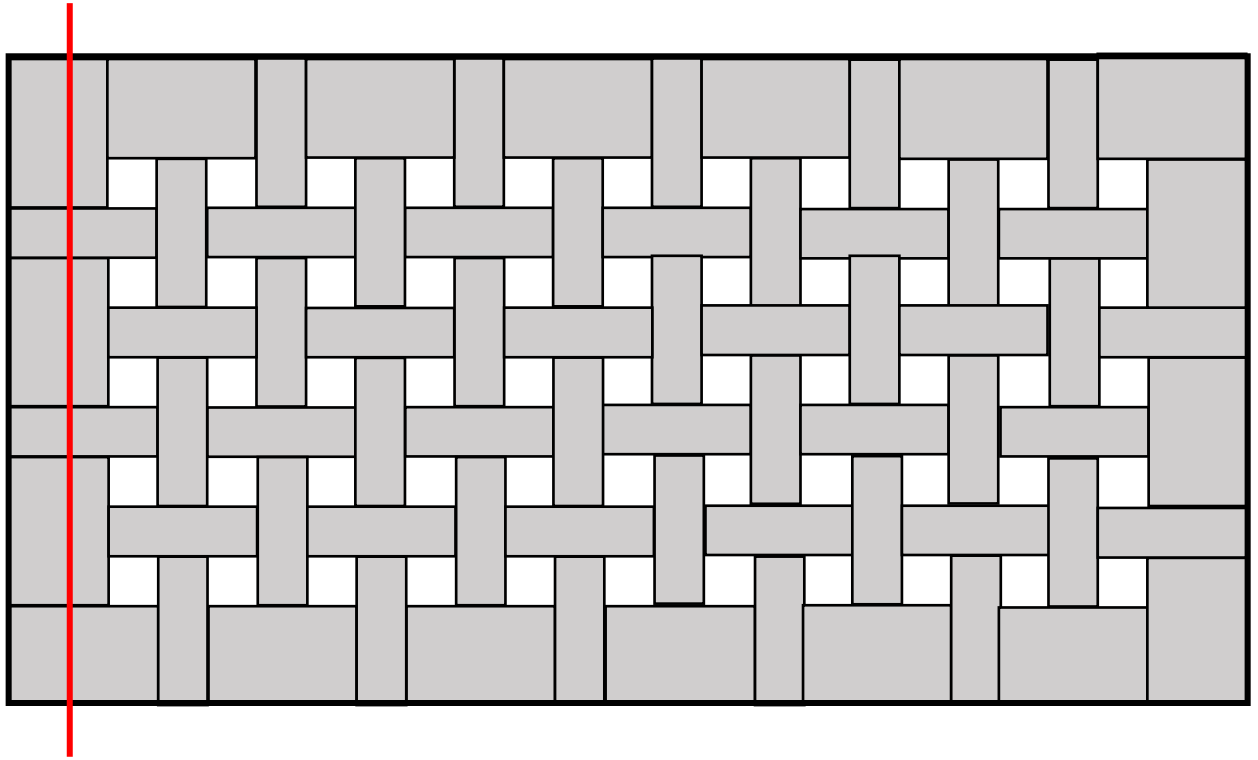
Proof?

- Assume Maximality for $\mathcal{R}$



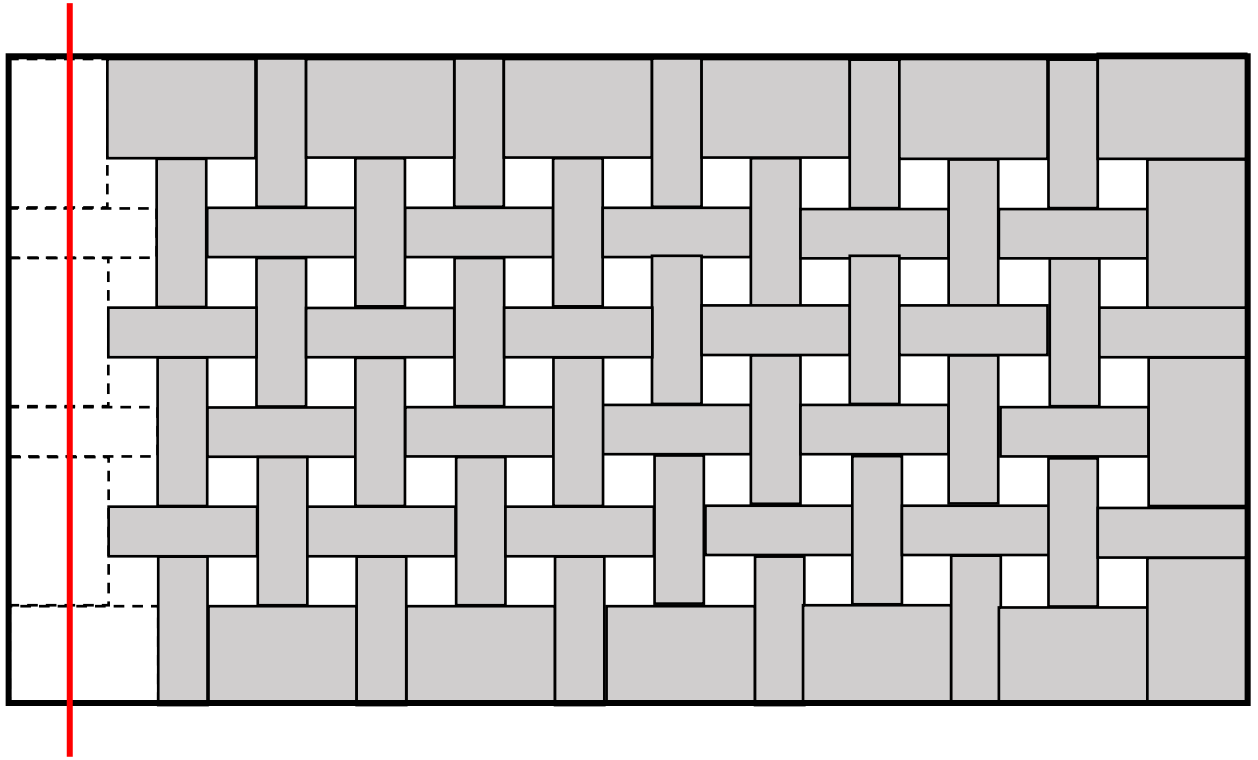
Proof?

- Fences and Cutting



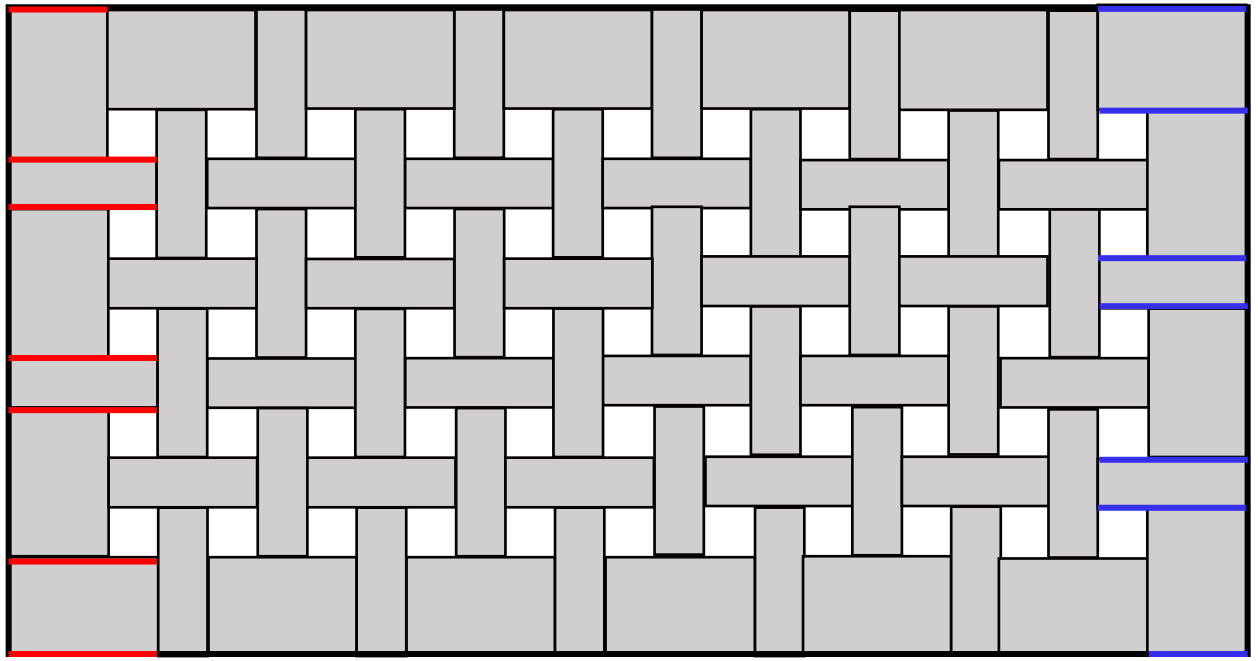
Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - We made no progress



Proof?

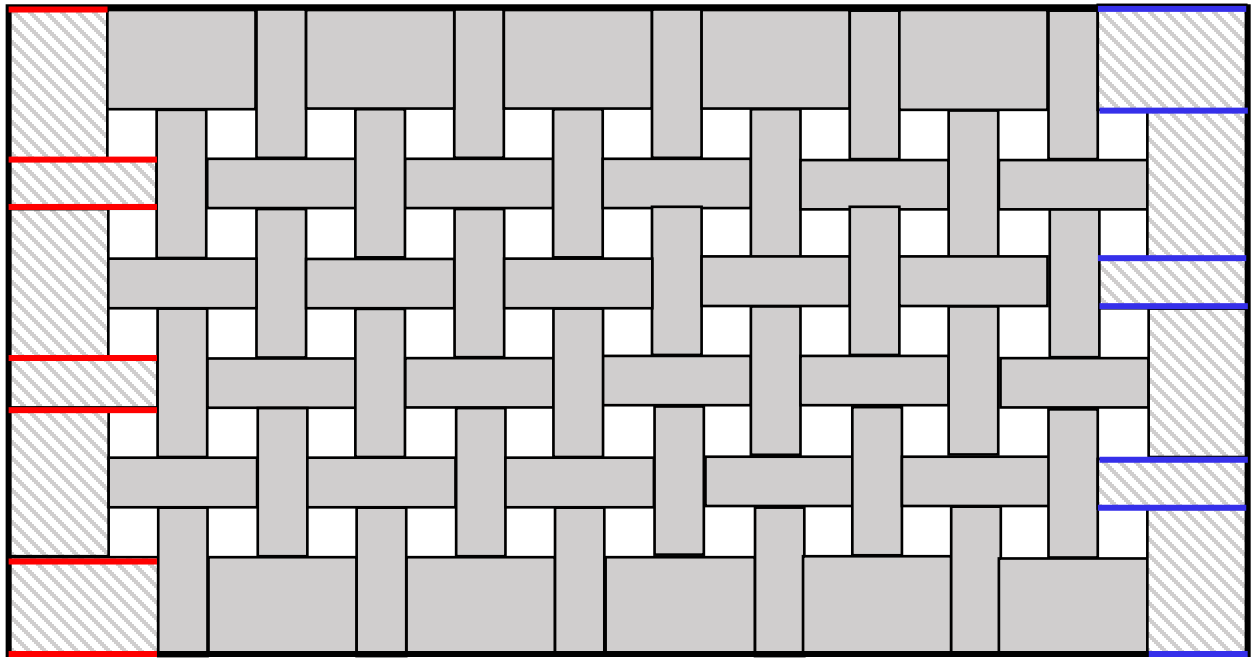
- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences



- Fences are horizontal line segments that join boundary of piece to boundary of a rectangle without intersecting any rectangles
- A rectangle is boundary rectangle in a piece if any of its horizontal edge is contained in a fence

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences



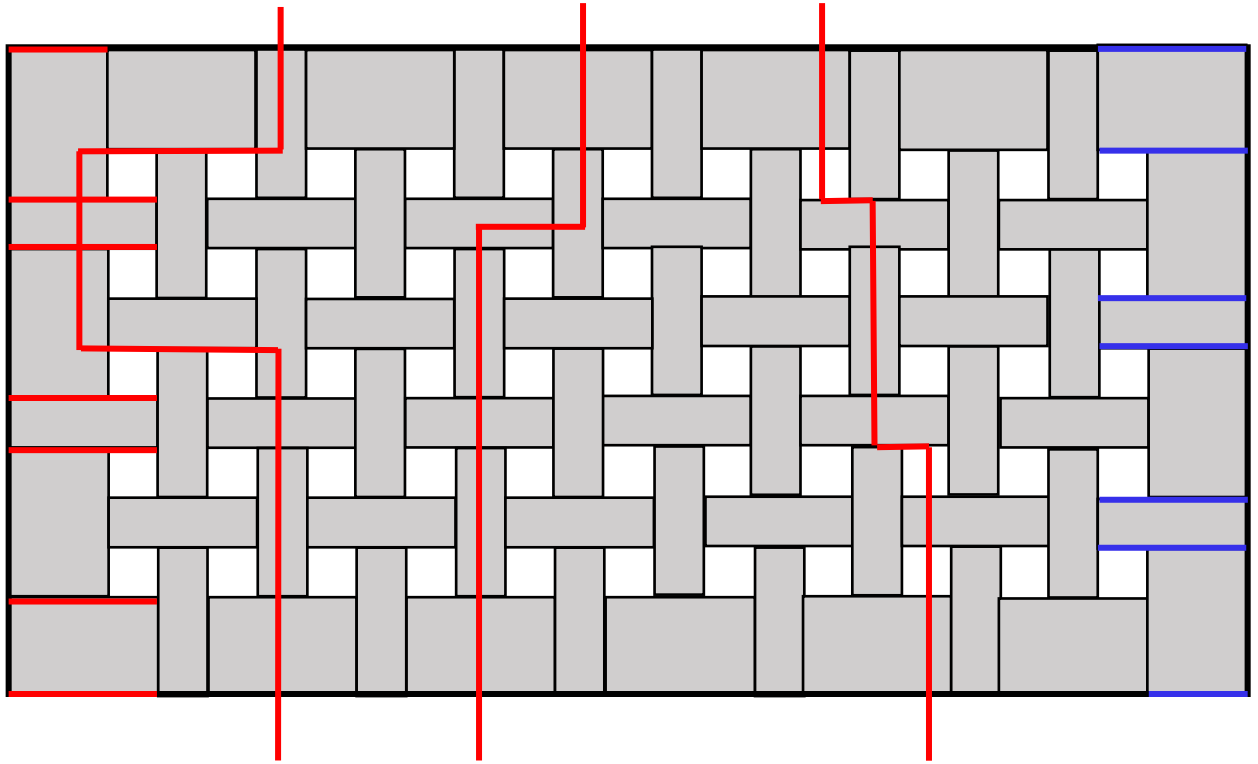
- Fences are horizontal line segments that join boundary of piece to boundary of a rectangle without intersecting any rectangles
- A rectangle is boundary rectangle in a piece if any of its horizontal edge is contained in a fence

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



Violates 1 Violates 2

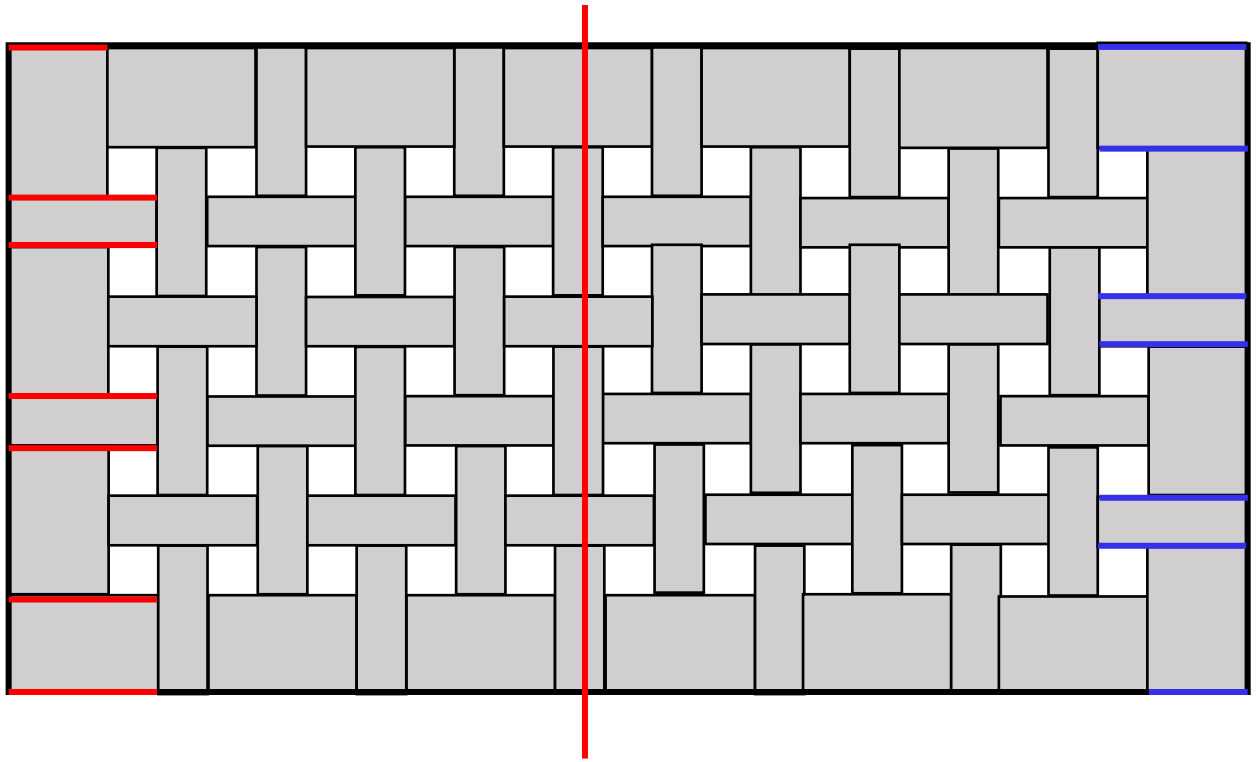
- Fences are horizontal line segments that join boundary of piece to boundary of a rectangle without intersecting any rectangles
- A rectangle is boundary rectangle in a piece if any of its horizontal edge is contained in a fence

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



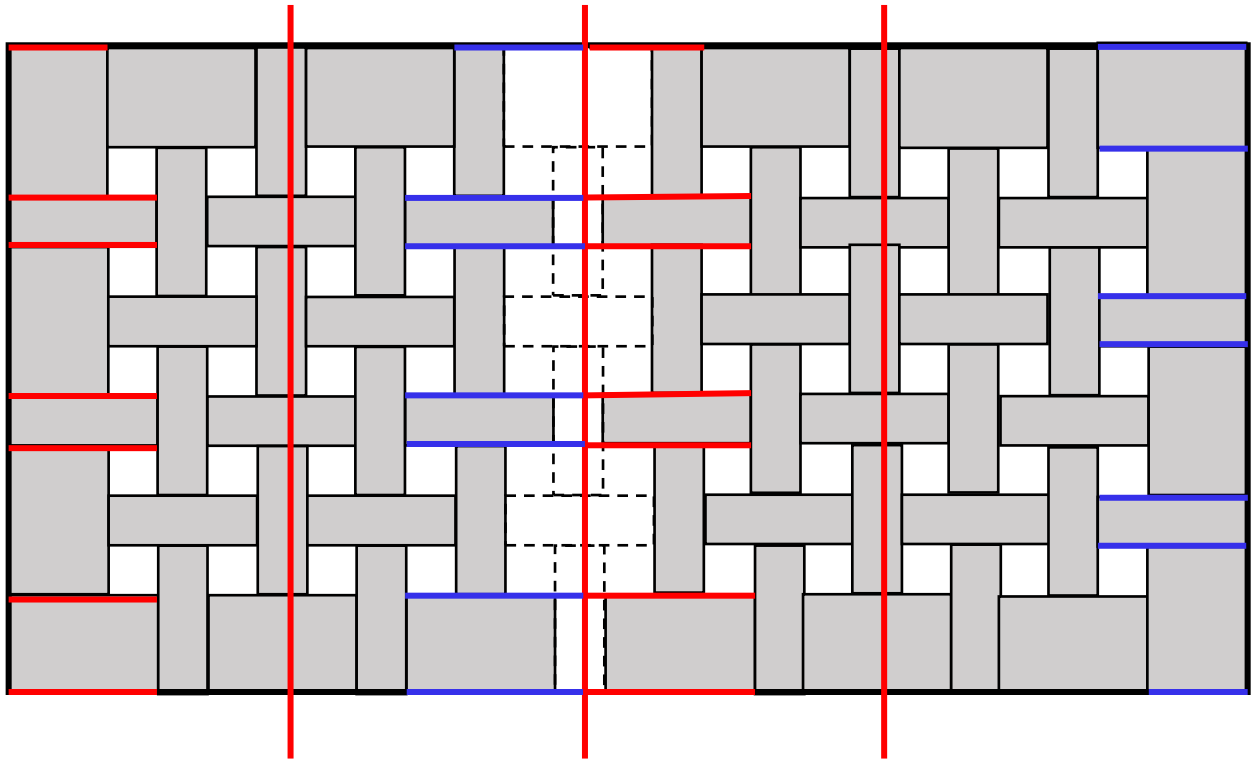
- Fences are horizontal line segments that join boundary of piece to boundary of a rectangle without intersecting any rectangles
- A rectangle is boundary rectangle in a piece if any of its horizontal edge is contained in a fence

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



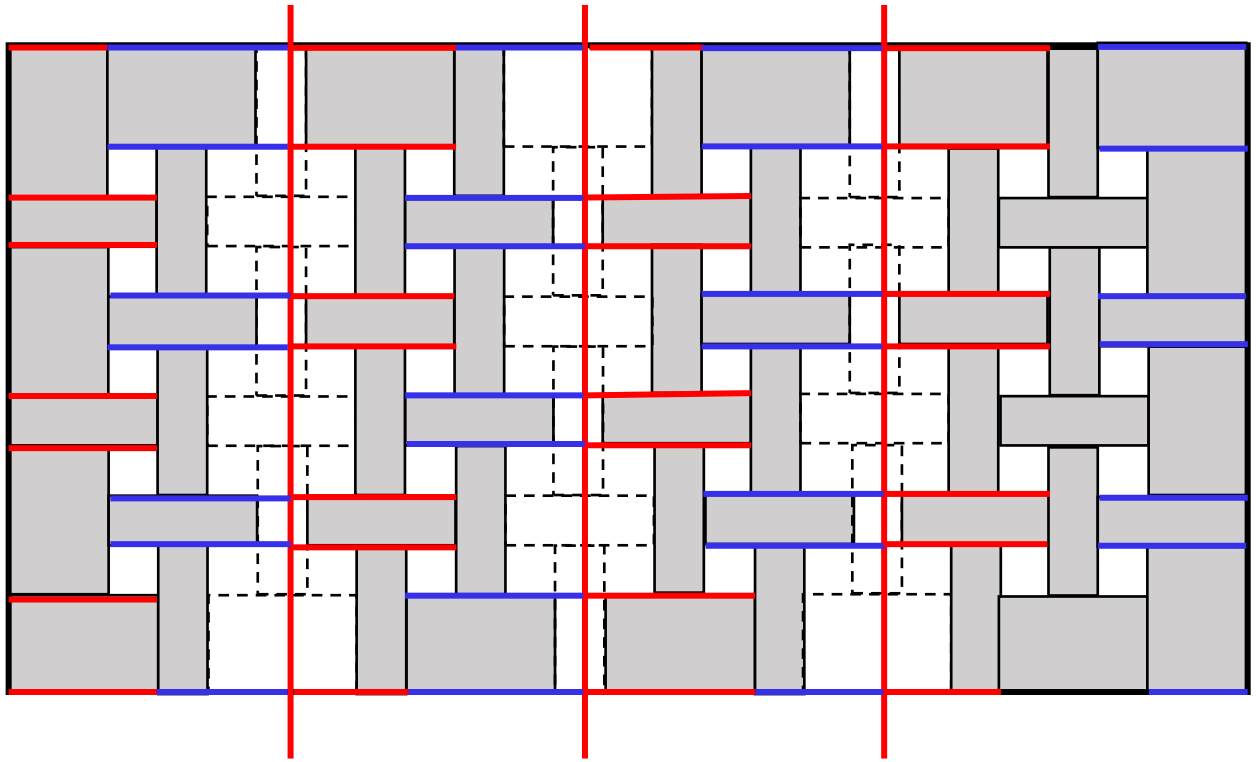
- Fences are horizontal line segments that join boundary of piece to boundary of a rectangle without intersecting any rectangles
- A rectangle is boundary rectangle in a piece if any of its horizontal edge is contained in a fence

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



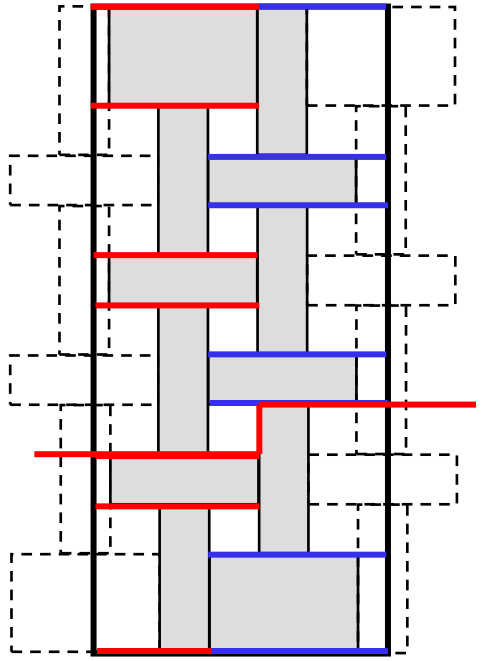
- Fences are horizontal line segments that join boundary of piece to boundary of a rectangle without intersecting any rectangles
- A rectangle is boundary rectangle in a piece if any of its horizontal edge is contained in a fence

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



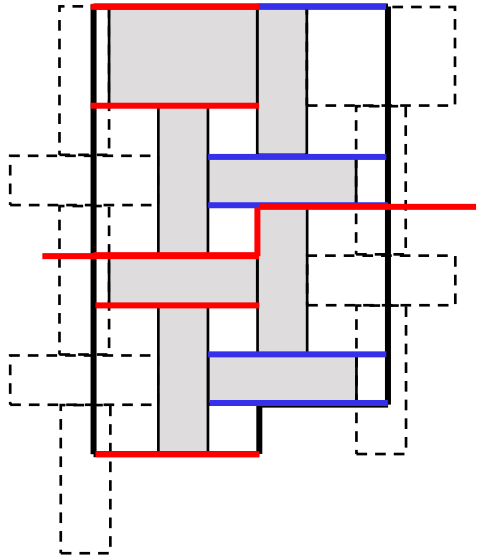
- Cannot make a straight vertical or horizontal cut without cutting any of the rectangles
- Use bends!

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



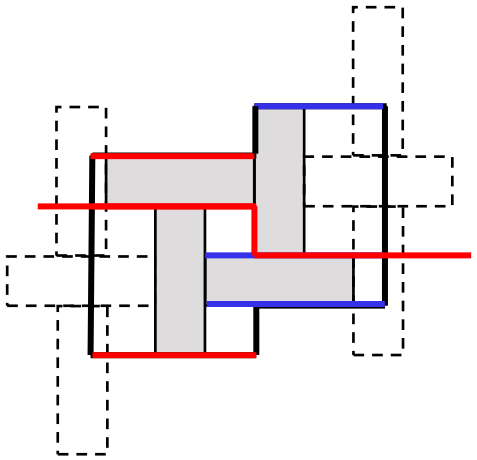
- Cannot make a straight vertical or horizontal cut without cutting any of the rectangles
- Use bends!

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



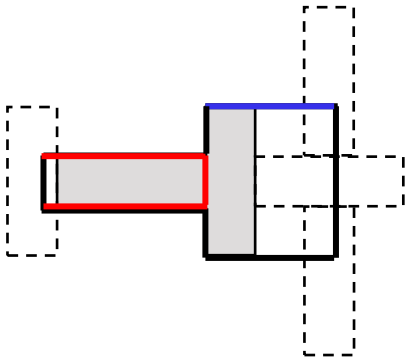
- Cannot make a straight vertical or horizontal cut without cutting any of the rectangles
- Use bends!

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles



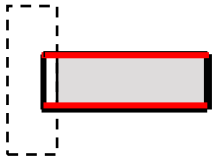
- Cannot make a straight vertical or horizontal cut without cutting any of the rectangles
- Use bends!

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don’t pass through fences
2. Horizontal segments of cuts don’t intersect any rectangles
3. Partition rule



- Cannot make a straight vertical or horizontal cut without cutting any of the rectangles
- Use bends!

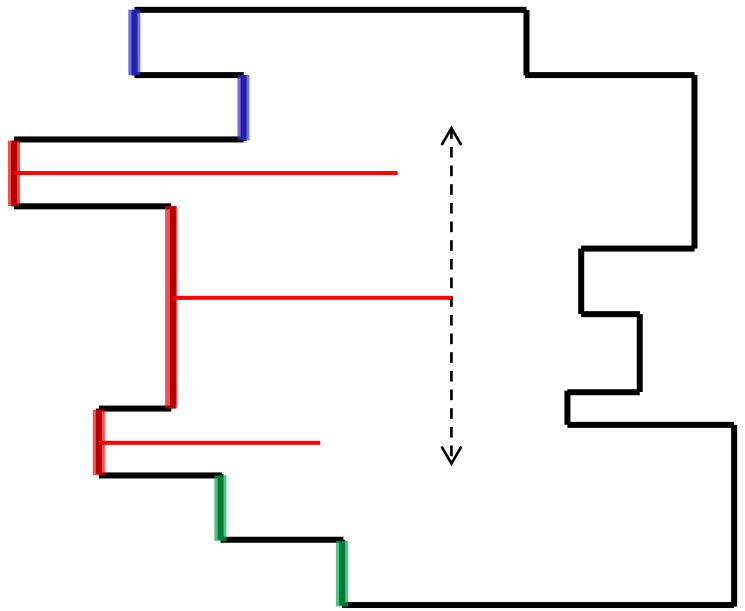
How do we make sure that the piece complexity doesn't go up in this process?

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles
3. Partition rule



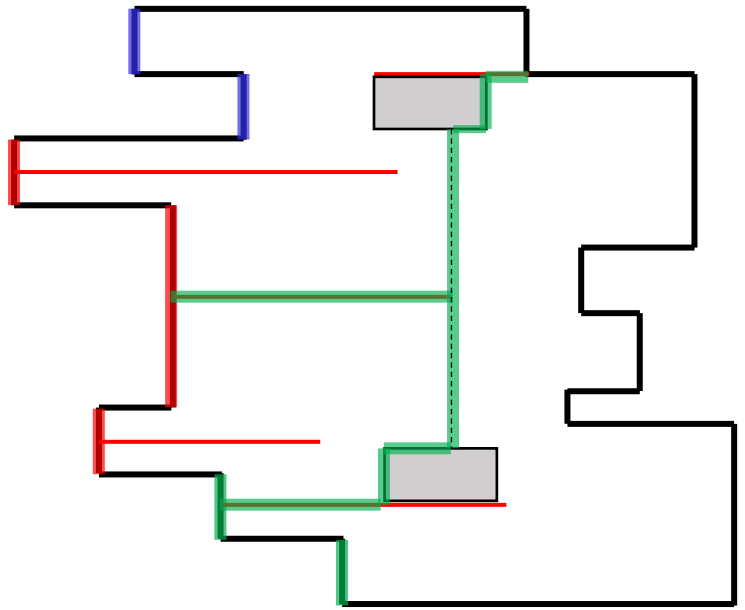
- Invariant: Assume the pieces are horizontally convex

Proof?

- Fences and Cutting
  - Intuitively, better cuts are not near the boundary
  - Block/Protect all boundary rectangles in every piece using fences

- Rules of cutting:

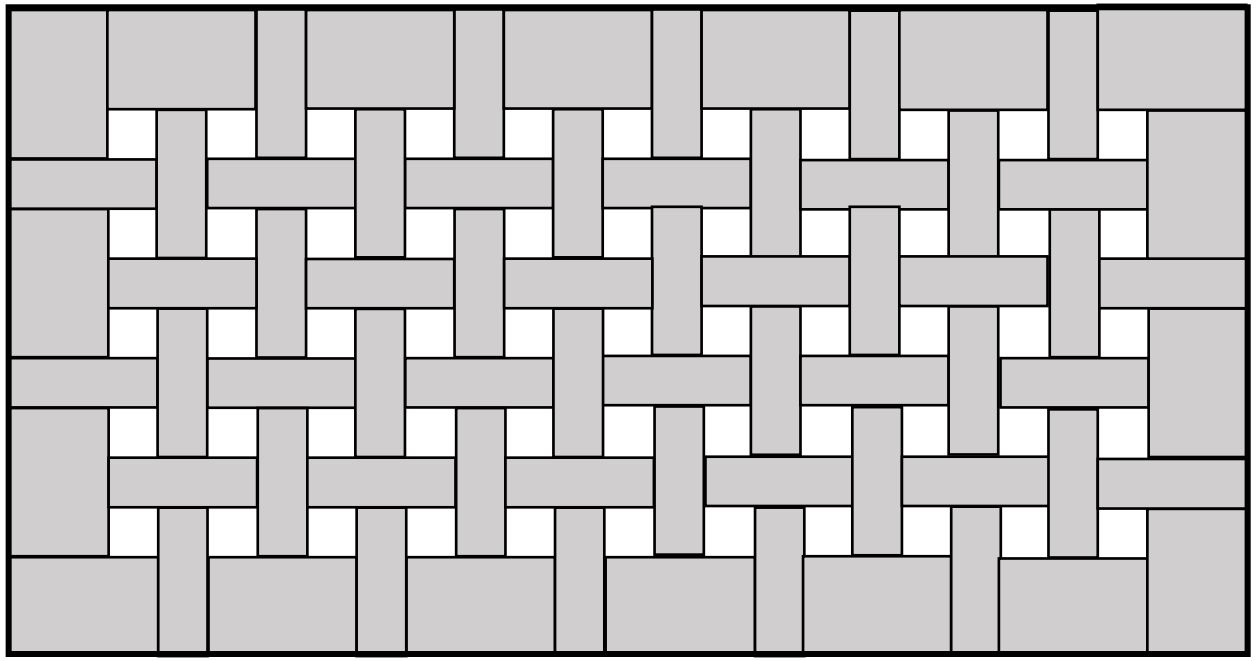
1. Vertical segments of cuts don't pass through fences
2. Horizontal segments of cuts don't intersect any rectangles
3. Partition rule



- Invariant: Assume the pieces are horizontally convex
- Invariant is maintained and piece complexity does not increase

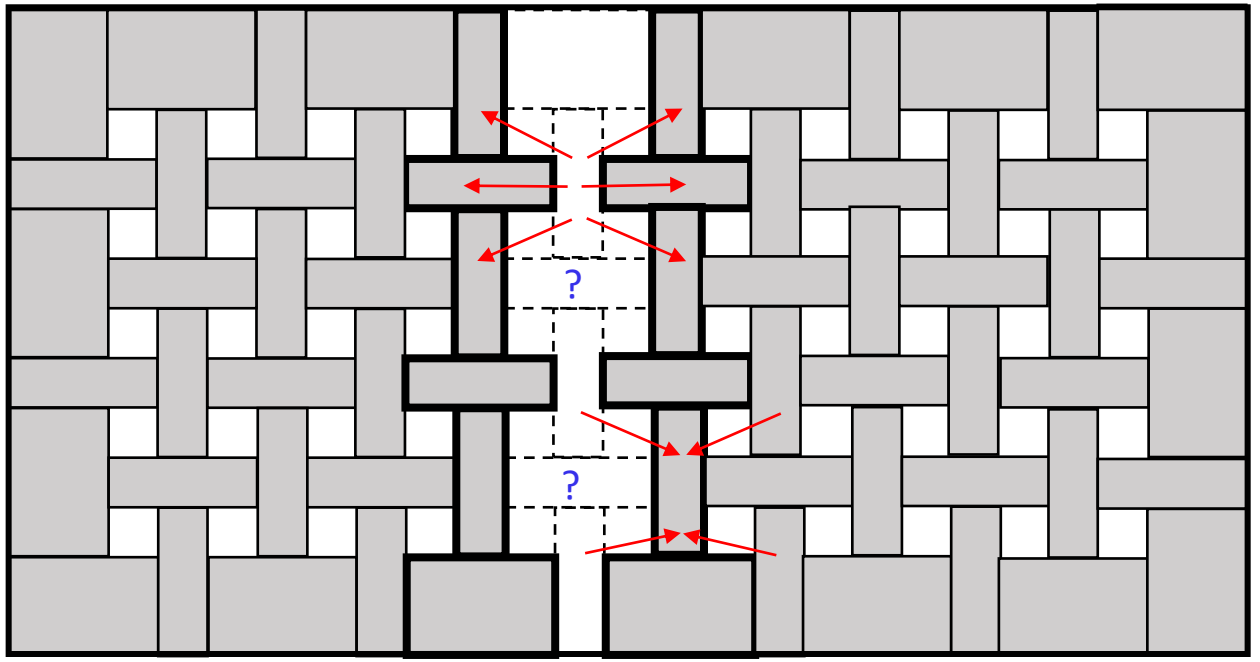
Proof?

- Charging Scheme



Proof?

- Charging Scheme



Observation:
Every cut creates some new boundary rectangles

We can charge the rectangles that are cut (killed) to these new boundary rectangles

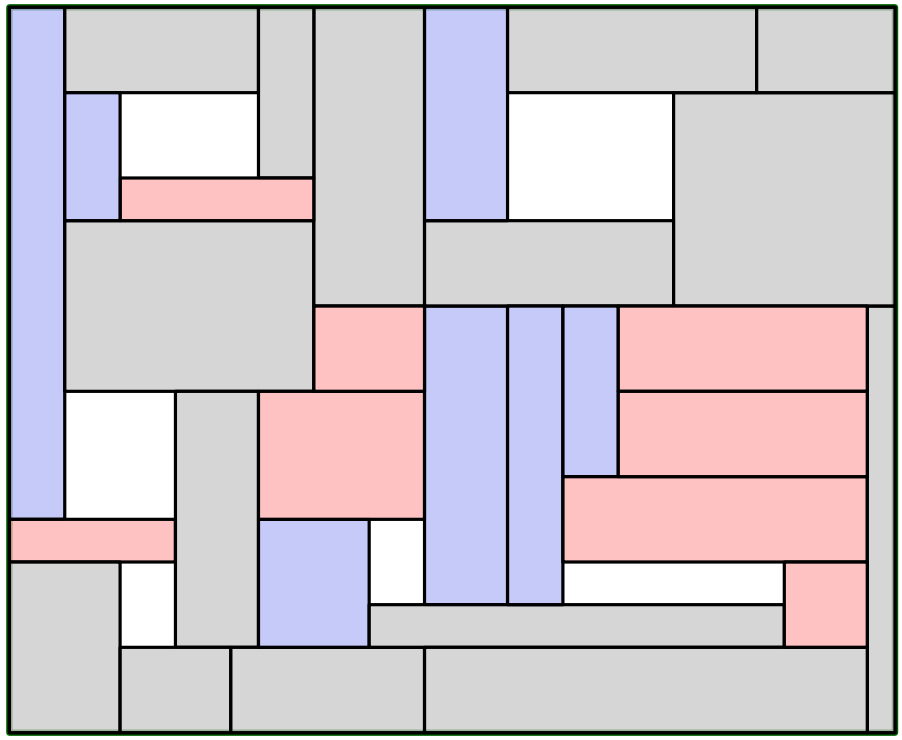
A boundary rectangle can be charged at most four times

Not every rectangle can be charged to some rectangle

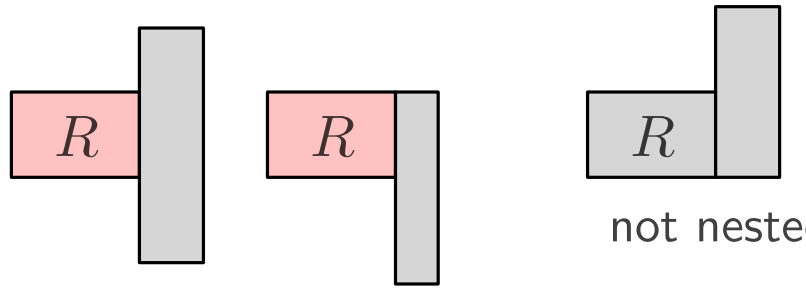
More details

Recursive partitioning

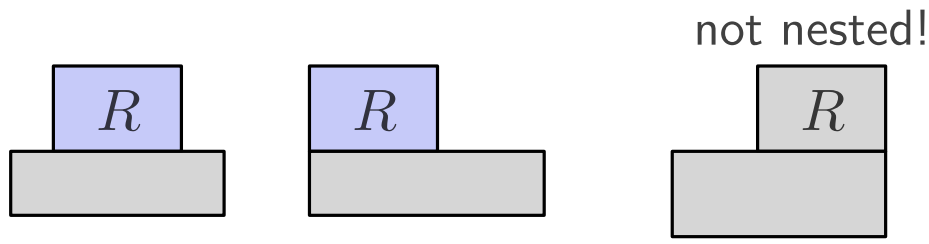
Assumption: All rectangles in OPT are *maximal*.



R is nested on its right

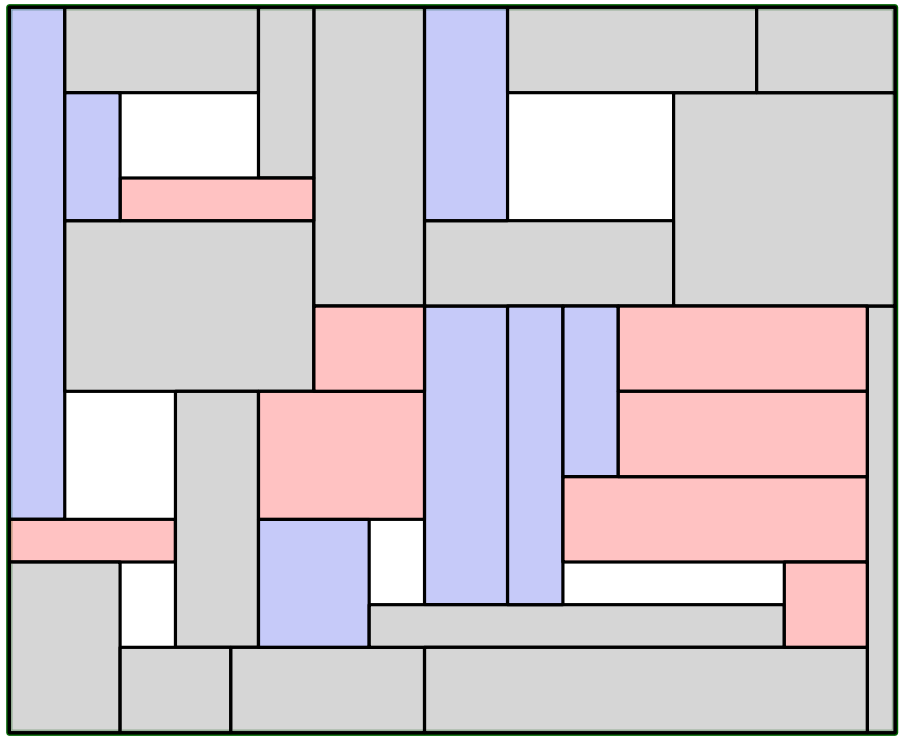


R is nested on its bottom

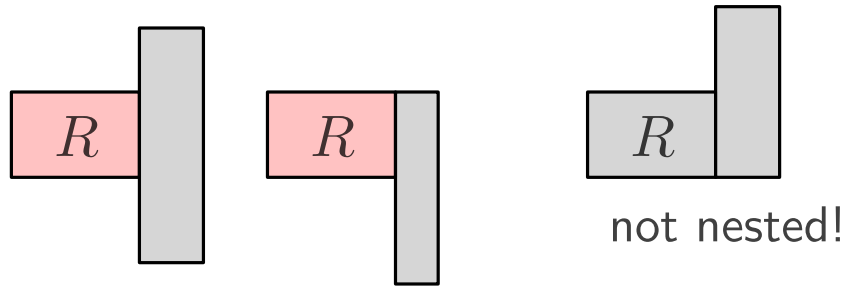


Recursive partitioning

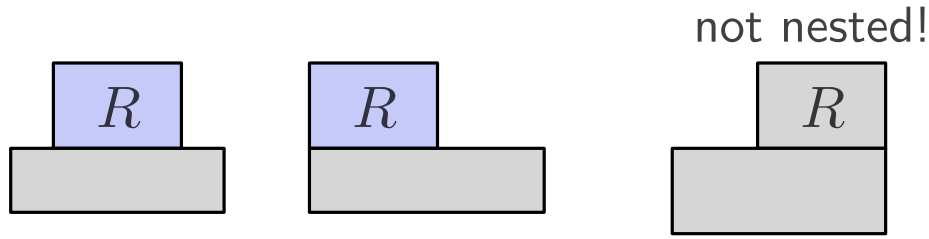
Assumption: All rectangles in OPT are *maximal*.



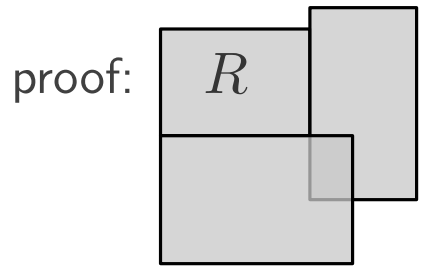
R is nested on its right



R is nested on its bottom

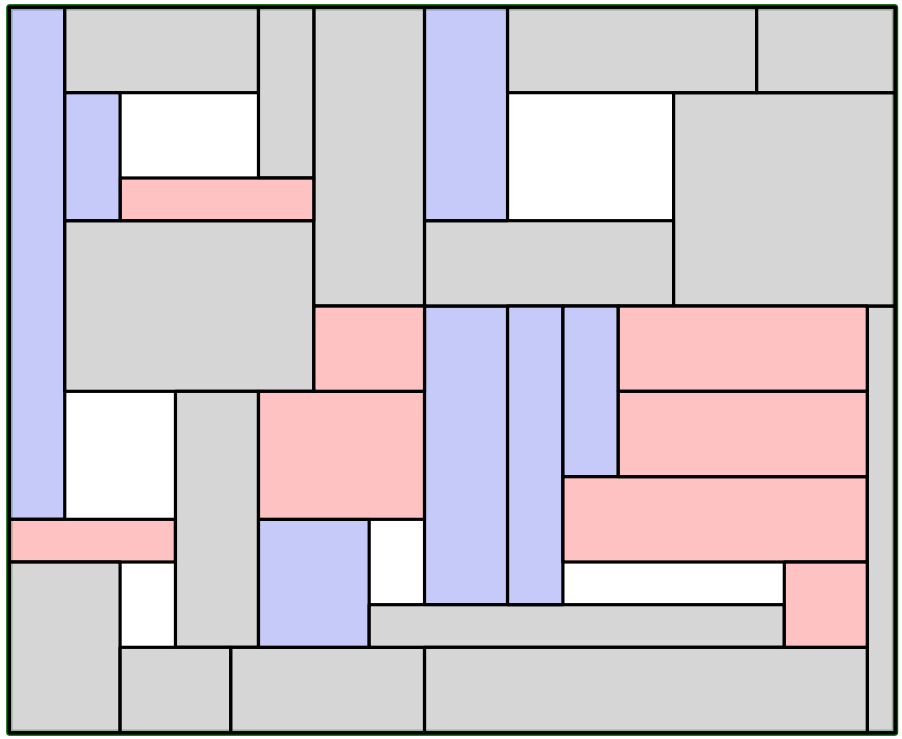


Claim: R cannot be nested on both directions.

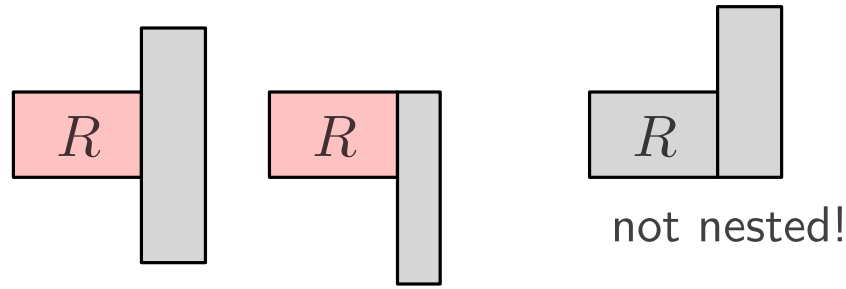


Recursive partitioning

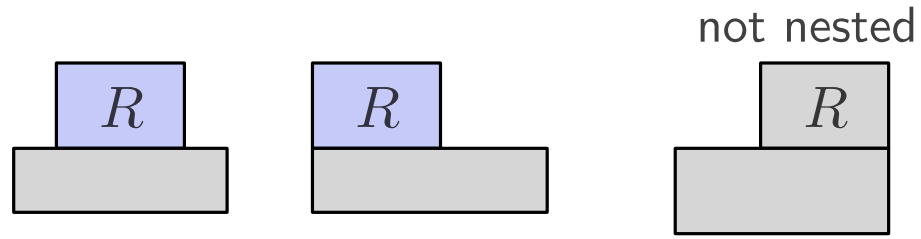
Assumption: All rectangles in OPT are *maximal*.



R is nested on its right



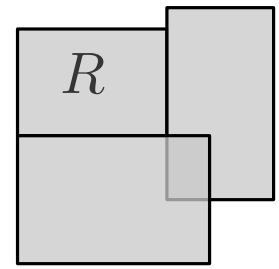
R is nested on its bottom



Claim: R cannot be nested on both directions.

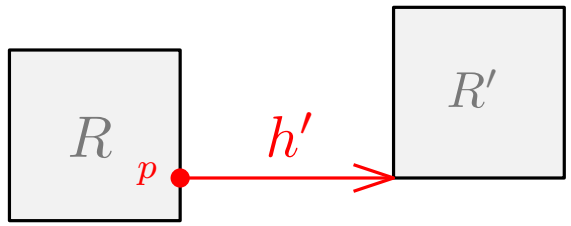
WLOG, we assume that at most $|OPT|/2$ are nested on their right

proof:



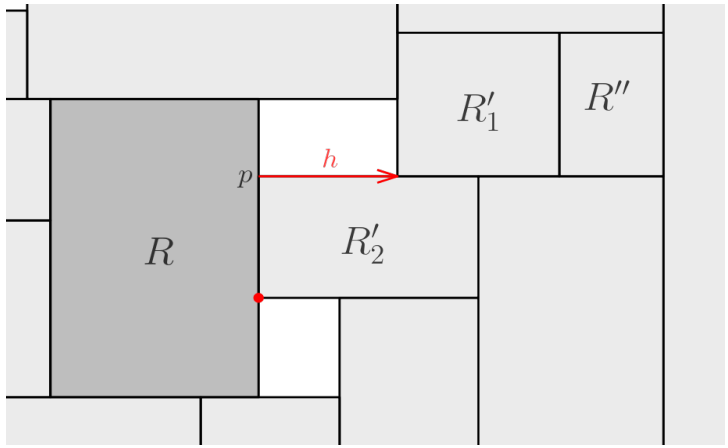
If R is intersected, who do we charge ?

answer: R charges a rectangle R' that it sees (on its right).



- h' does not intersect any rectangle in OPT.
- p is not the top-right corner of R.
- h does not containt the bottom edge of any other rectangles.

example.



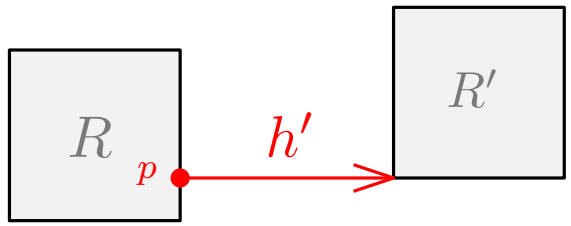
R sees R'_2 and R'_1

R does not see R''

R'_2 does not see sees R''

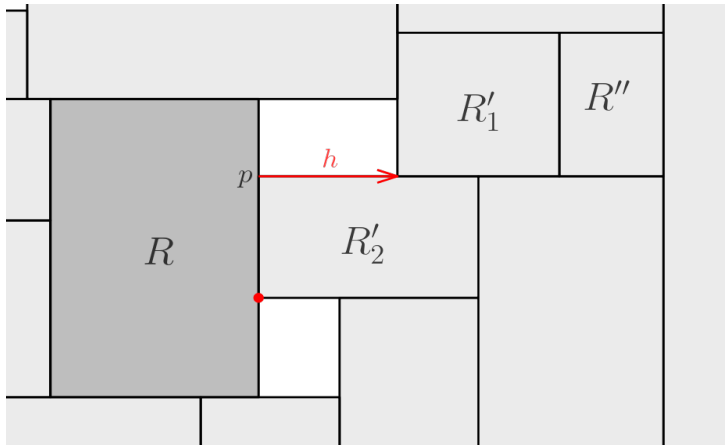
If R is intersected, who do we charge ?

answer: R charges a rectangle R' that it sees (on its right).



- h' does not intersect any rectangle in OPT.
- p is not the top-right corner of R.
- h does not containt the bottom edge of any other rectangles.

example.



R sees R'_2 and R'_1

R does not see R''

R'_2 does not see sees R''

→ a rectangle R' is seen by at most one rectangle

→ a rectangle can be charged by at most once

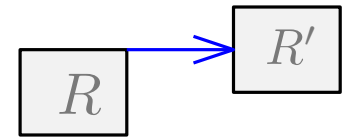
Lemma.

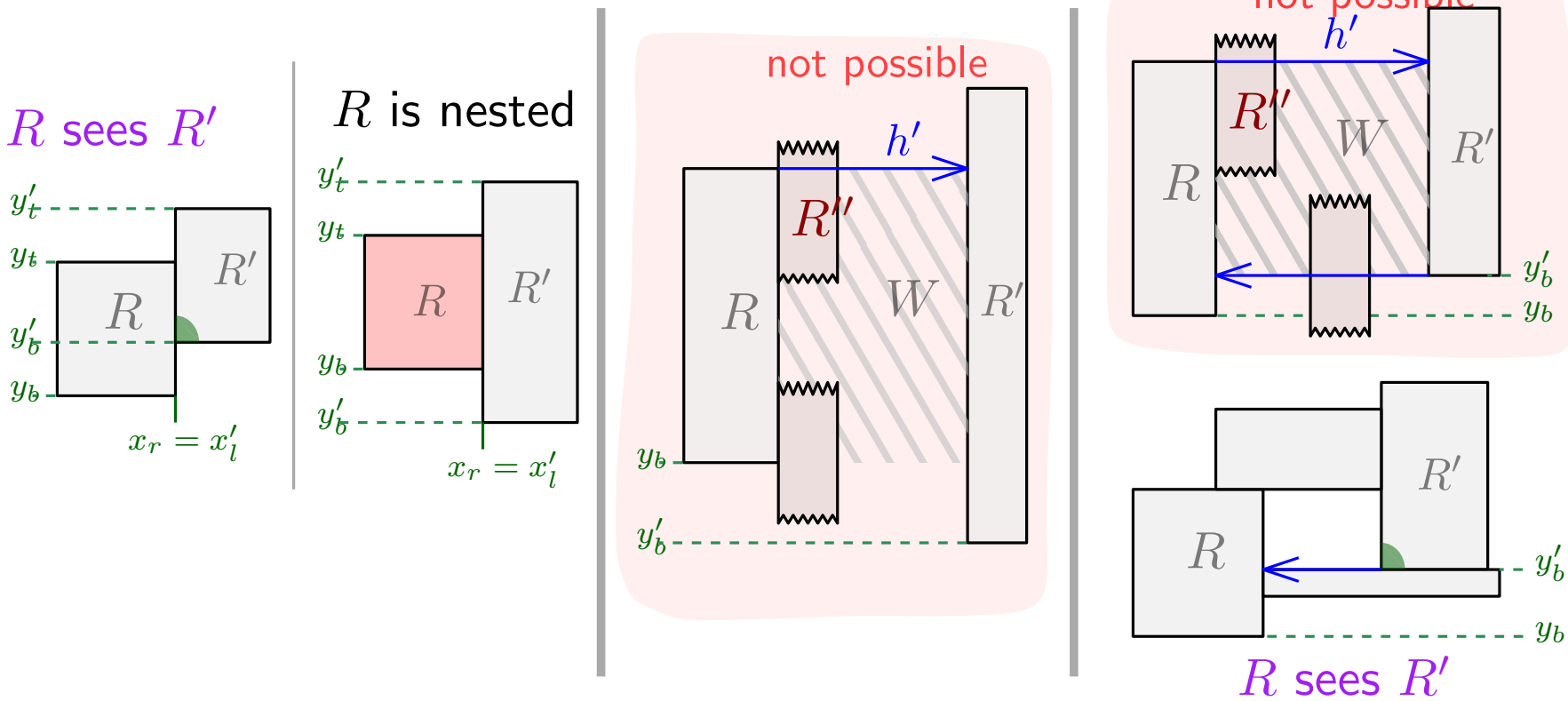
Let $R \in OPT$. Either:

- R sees someone,
- or, R is nested on its right.

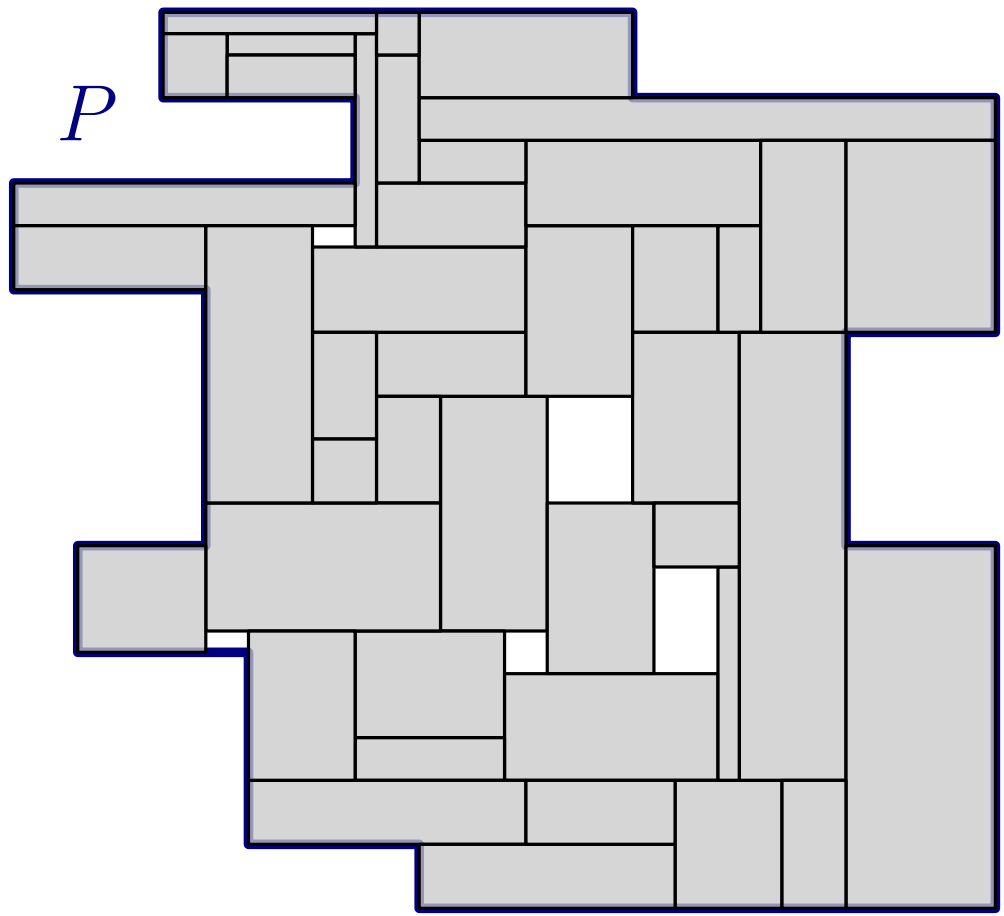
(or R has its right edge contained in the bounding-box S)

proof: we look on the right from our top-right corner:

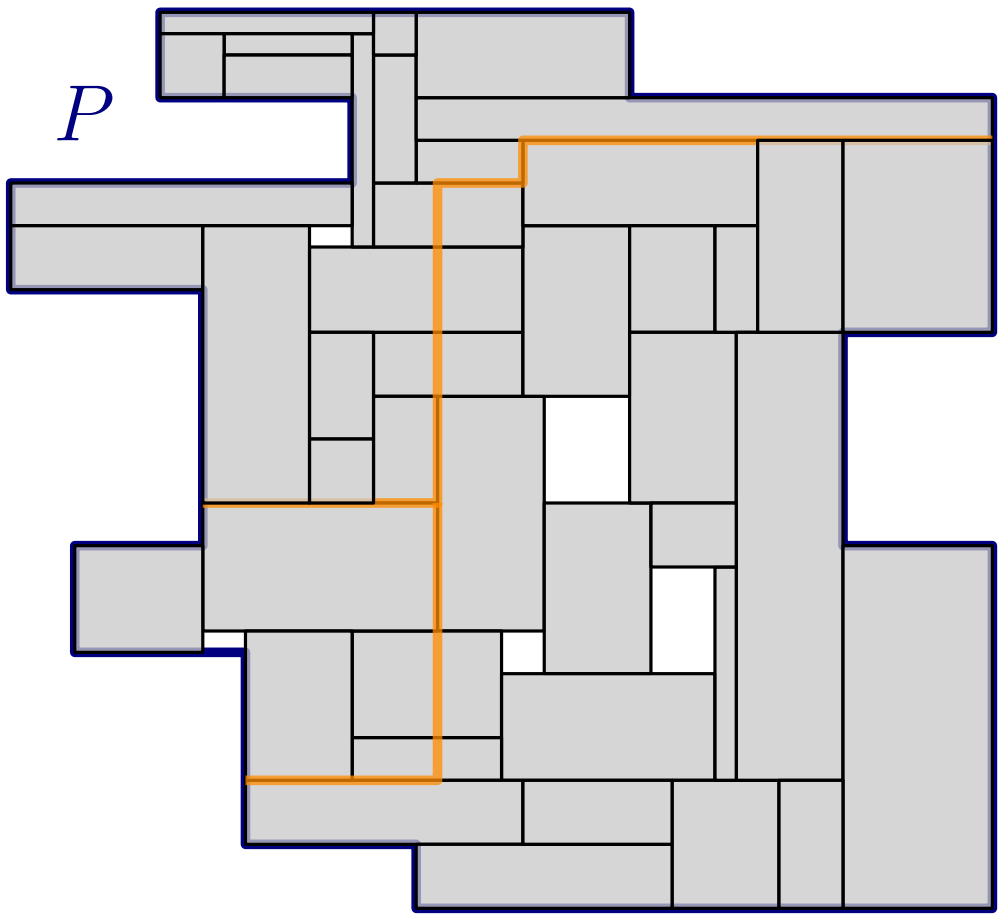




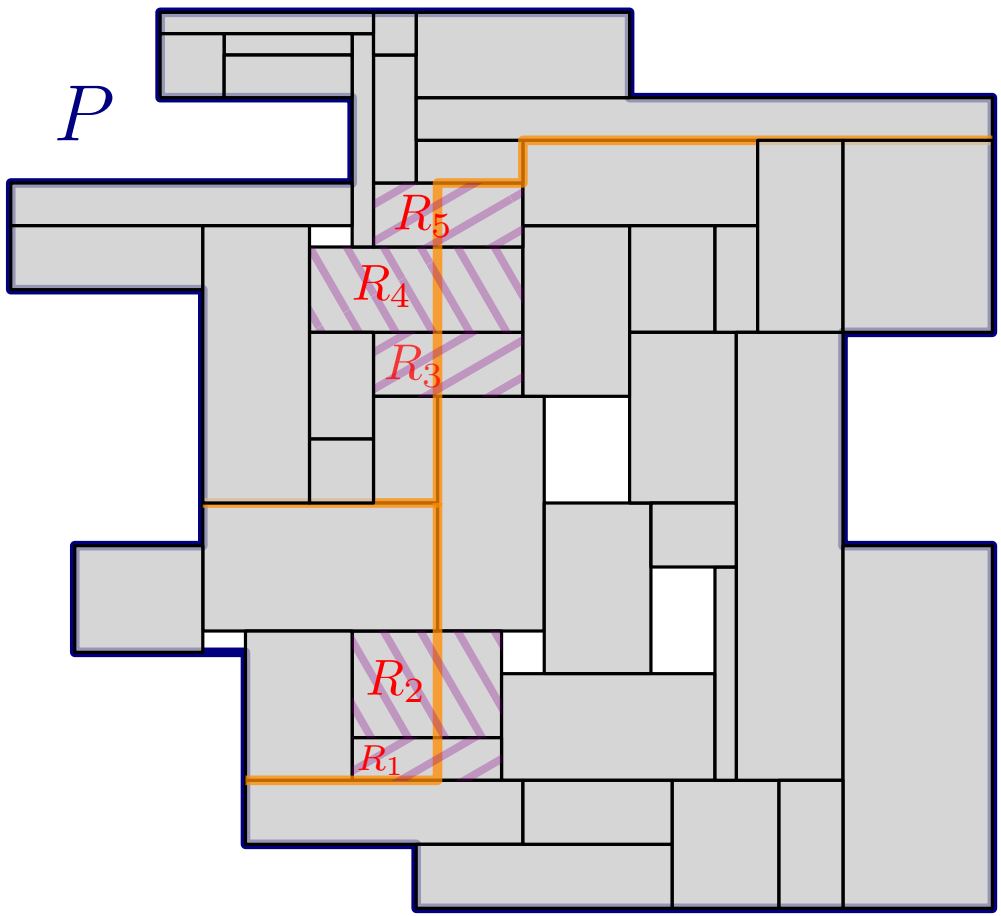
example of the charging process.



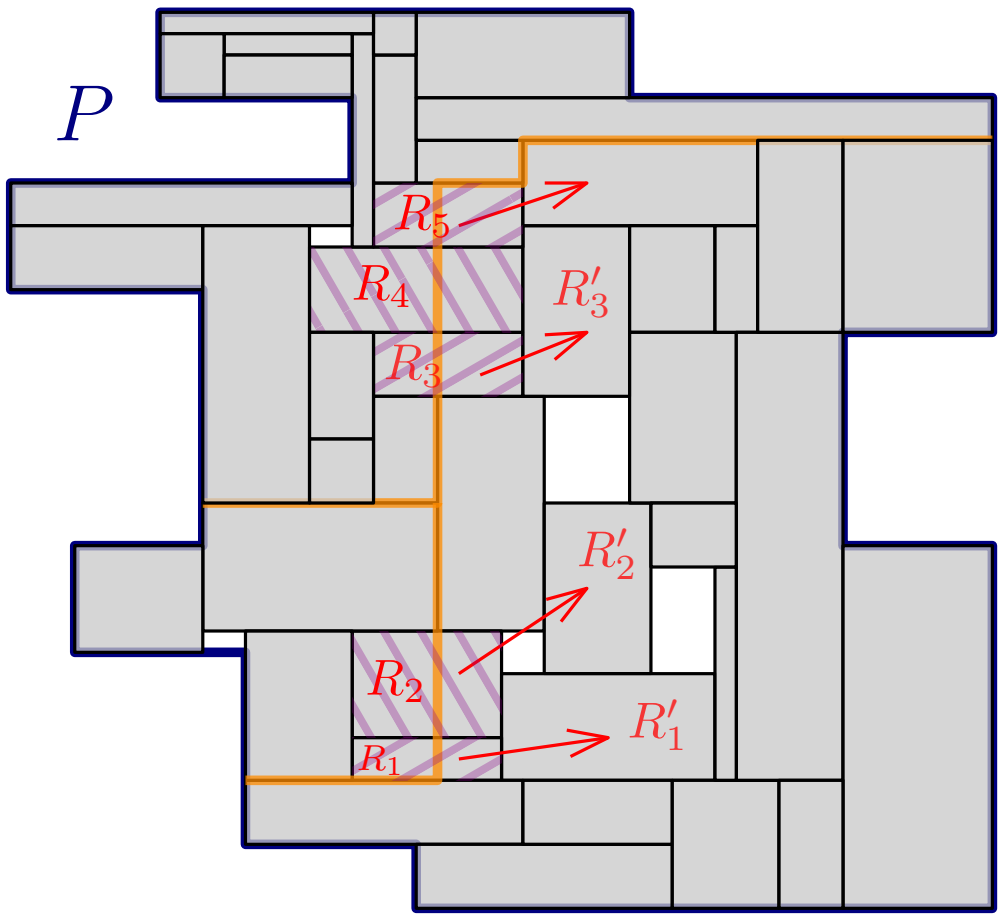
example of the charging process.



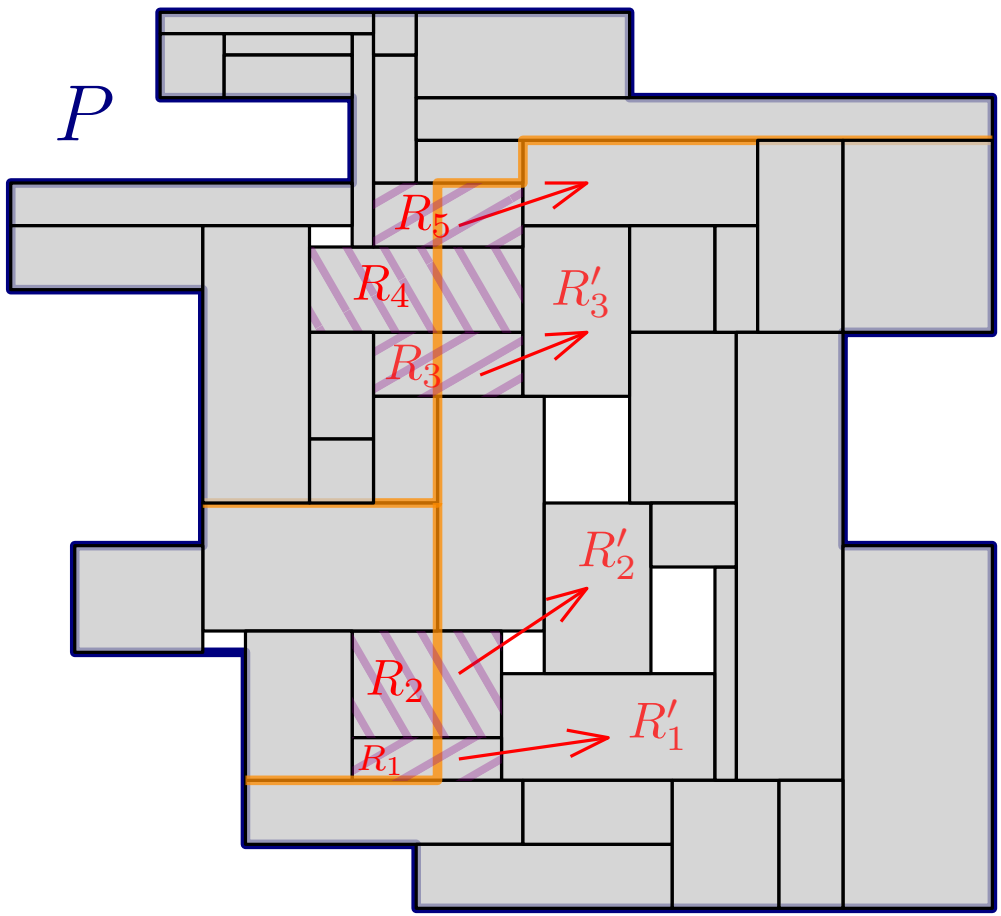
example of the charging process.



example of the charging process.

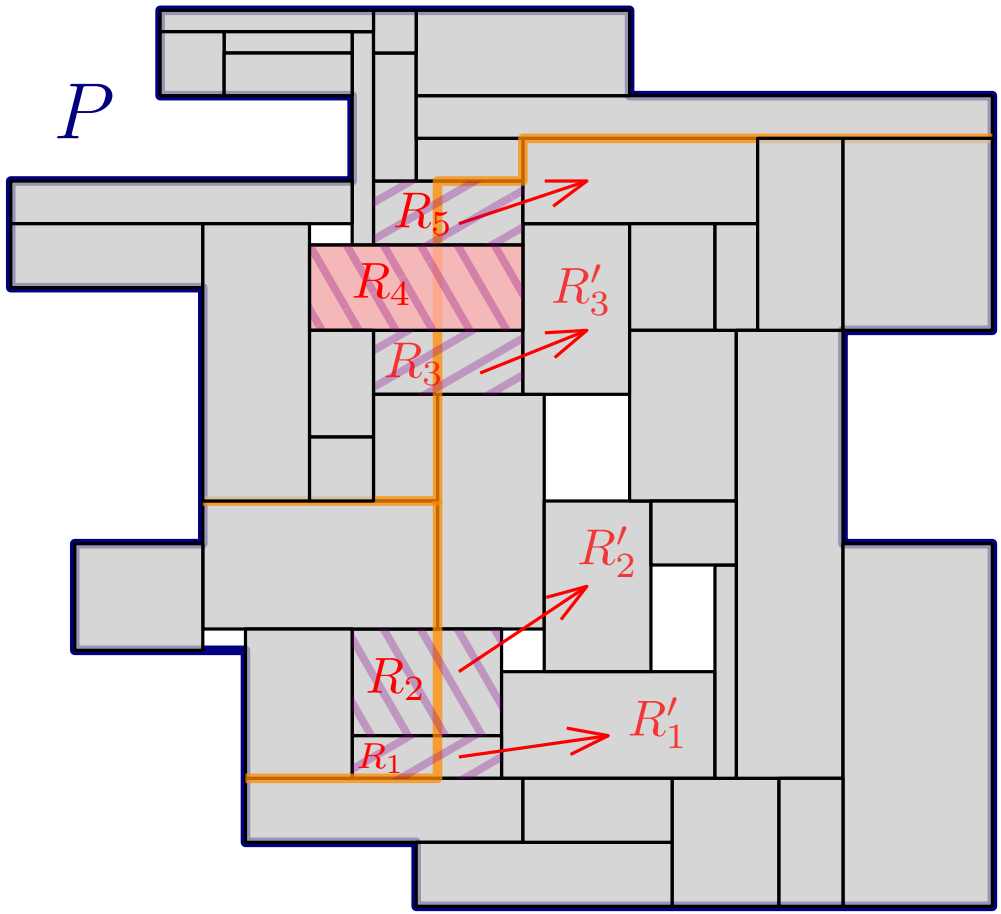


example of the charging process.



R'_3 is charged by R_3 and by R_4 ? problem... ?

example of the charging process.



R'_3 is charged by R_3 and by R_4 ? problem... ?

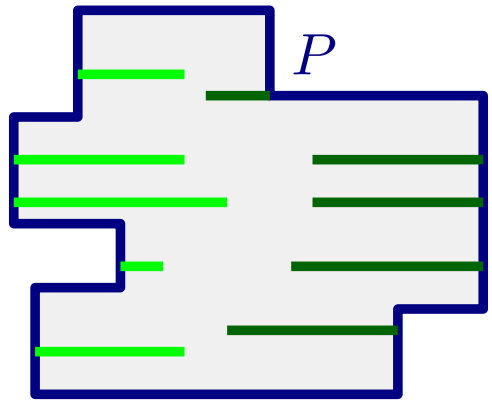
No! R_4 is nested on its right, so when we kill him, we don't need to pay for it (not again)

How to protect rectangles that charged ?

answer: using line fences

= horizontal line segment in P with one endpoint on the boundary that does not intersect rectangles from OPT that lie inside P.

(but might contain edges of rectangles and might intersect rectangles previously intersected)



line fences in P

emerging from the left

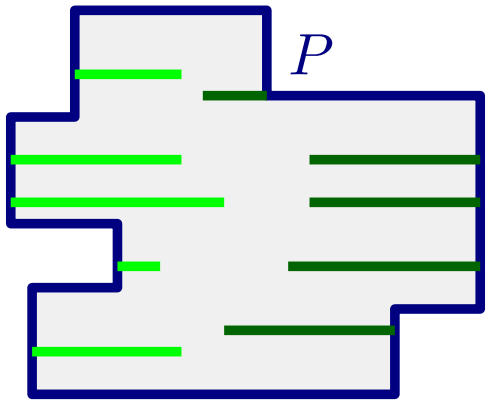
emerging from the right

How to protect rectangles that charged ?

answer: using line fences

= horizontal line segment in P with one endpoint on the boundary that does not intersect rectangles from OPT that lie inside P.

(but might contain edges of rectangles and might intersect rectangles previously intersected)



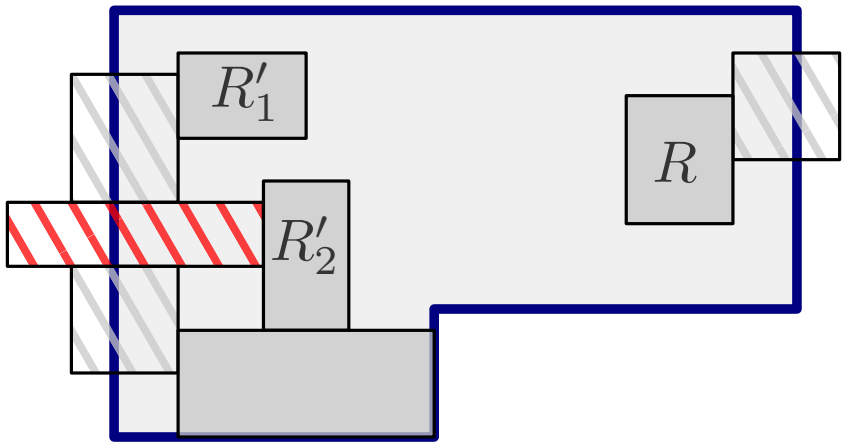
line fences in P

emerging from the left

emerging from the right

def: protected

A rectangle R in P is protected if its bottom or top edge is contained in a line fence.

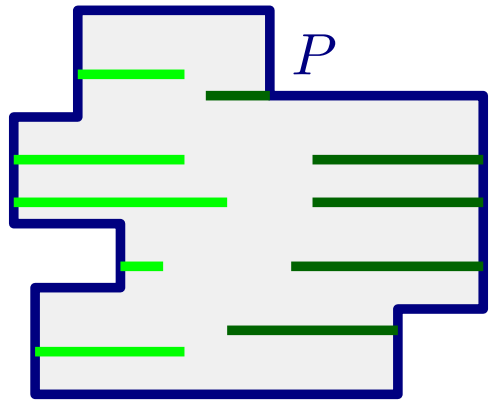


How to protect rectangles that charged ?

answer: using line fences

= horizontal line segment in P with one endpoint on the boundary that does not intersect rectangles from OPT that lie inside P.

(but might contain edges of rectangles and might intersect rectangles previously intersected)



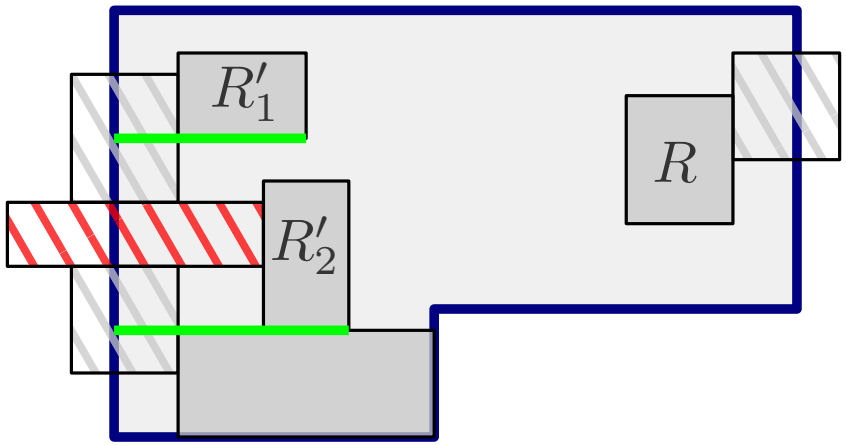
line fences in P

emerging from the left

emerging from the right

def: protected

A rectangle R in P is protected if its bottom or top edge is contained in a line fence.



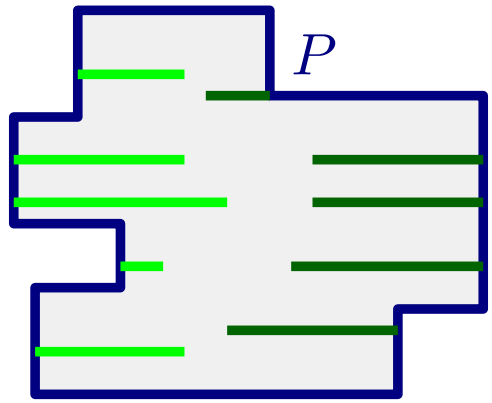
- R_1' and R_2' are protected by fences from the left

How to protect rectangles that charged ?

answer: using line fences

= horizontal line segment in P with one endpoint on the boundary that does not intersect rectangles from OPT that lie inside P.

(but might contain edges of rectangles and might intersect rectangles previously intersected)



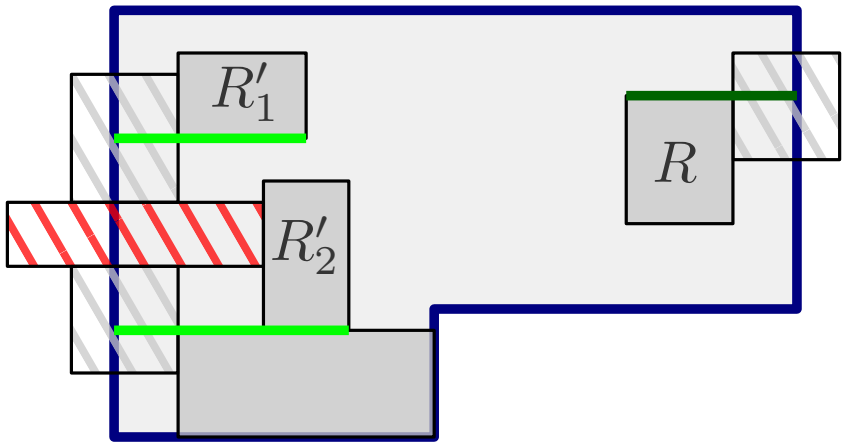
line fences in P

emerging from the left

emerging from the right

def: protected

A rectangle R in P is protected if its bottom or top edge is contained in a line fence.



- R'_1 and R'_2 are protected by fences from the left
- R is protected by a fence from the right

Line-partitioning Lemma

Given a horizontally convex polygon with at 26 sides, there exists a cut C, s.t.

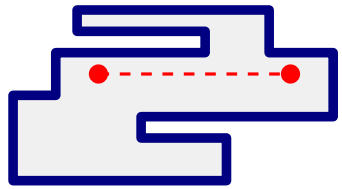
(1) C has at most 8 line segments,

(2) C divides P into at most 3 horizontally convex prolygons, with at most 26 sides each,

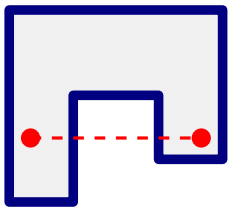
(3) There is only one single vertical line segment ℓ that can intersect rectangles in P,

(4) ℓ does not intersect any protected rectangles.

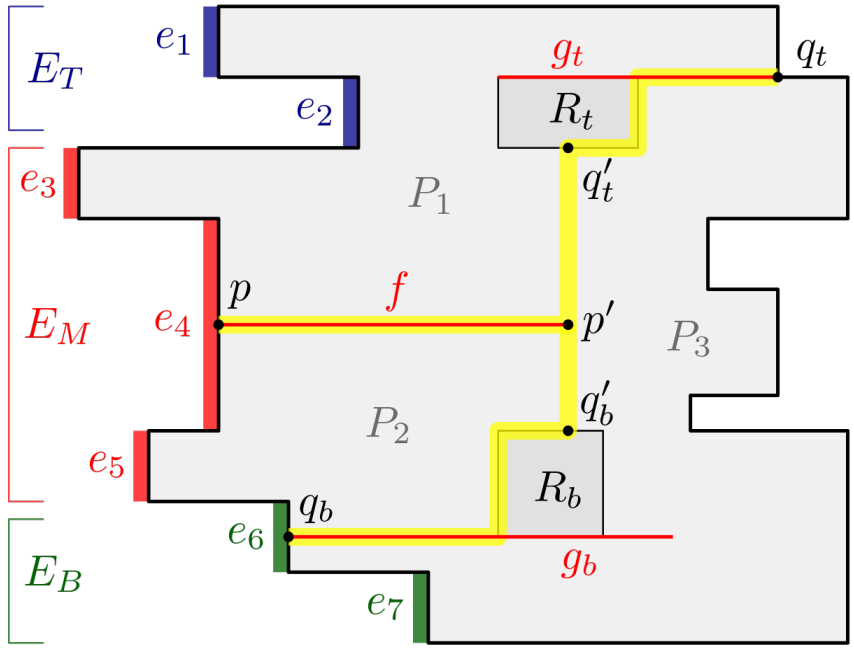
horizontally convex



not horizontally convex



proof:



How to improve the approximation ratio to $(2 + \varepsilon)$?

How to improve the approximation ratio to $(2+\varepsilon)$?

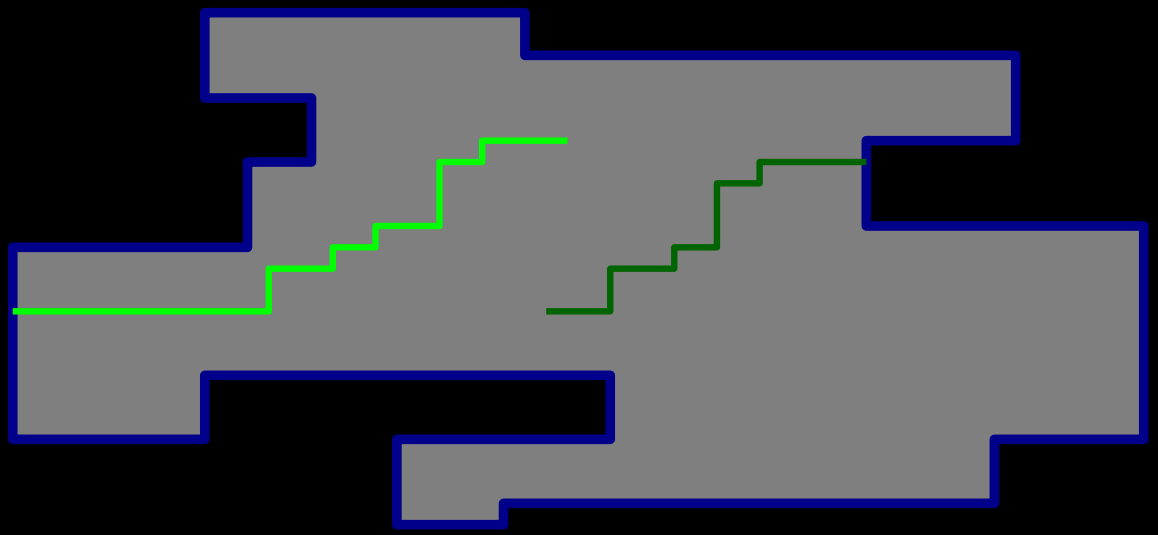
- we need to charge more rectangles

How to improve the approximation ratio to $(2+\varepsilon)$?

- we need to charge more rectangles
  $O(1/\varepsilon)$

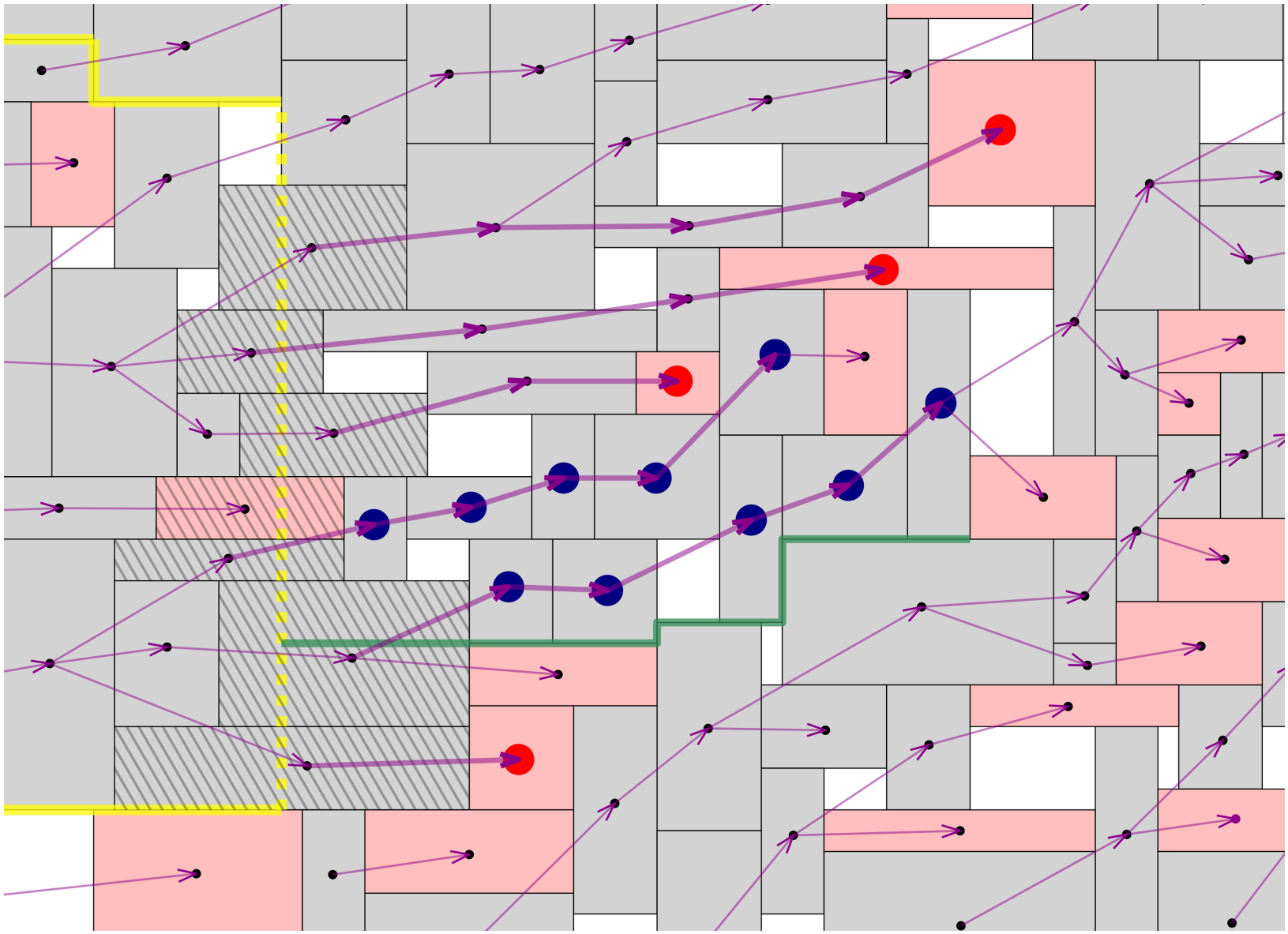
How to improve the approximation ratio to $(2+\varepsilon)$?

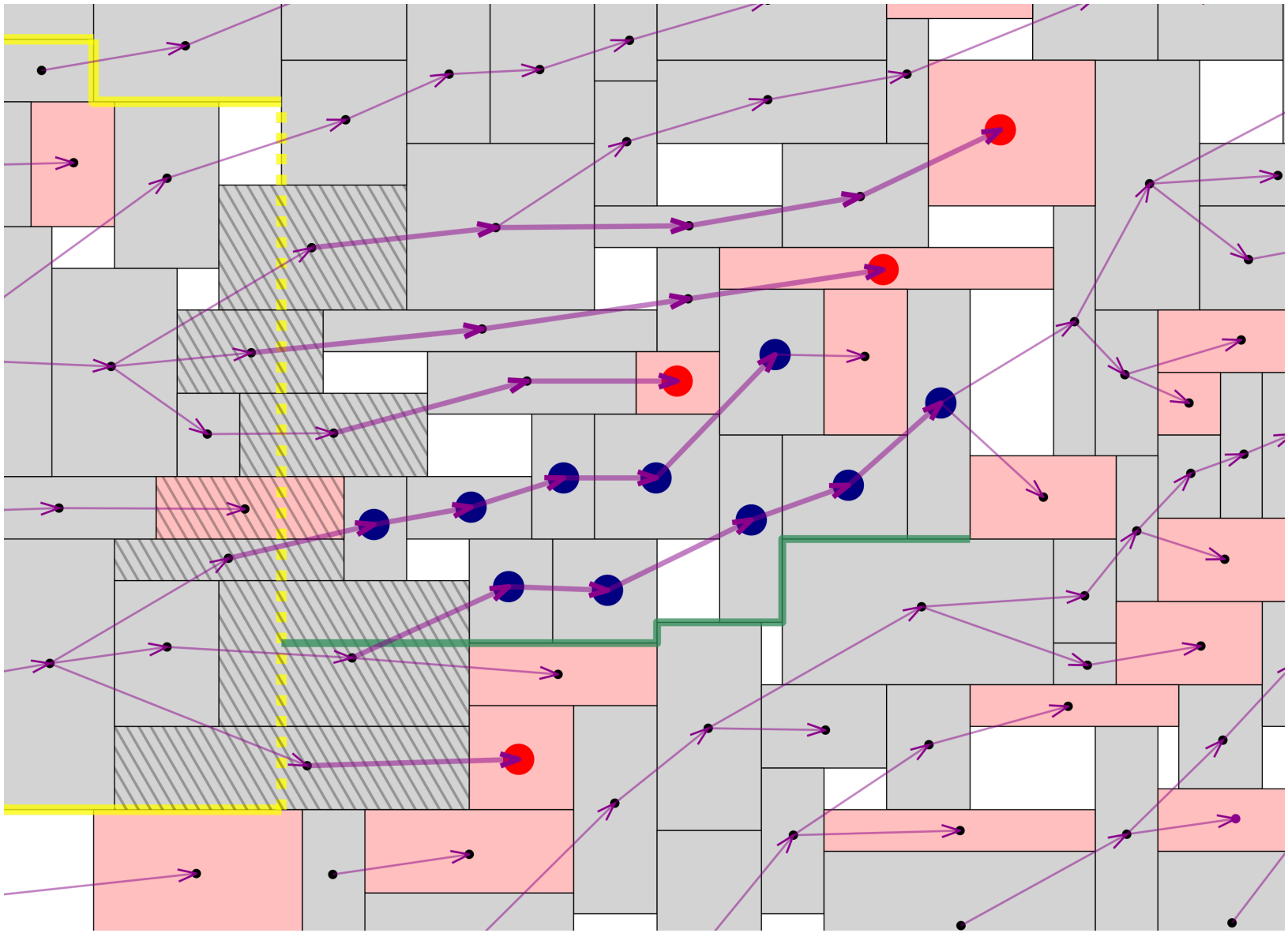
- we need to charge more rectangles
  $O(1/\varepsilon)$
- but we need to protect them, so we need longer fences!



τ-fences with τ bends

$\tau = O(1/\varepsilon)$



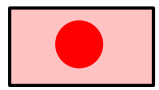


charging process:

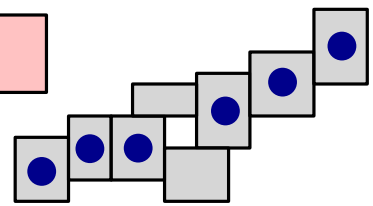
A rectangle intersected charges either

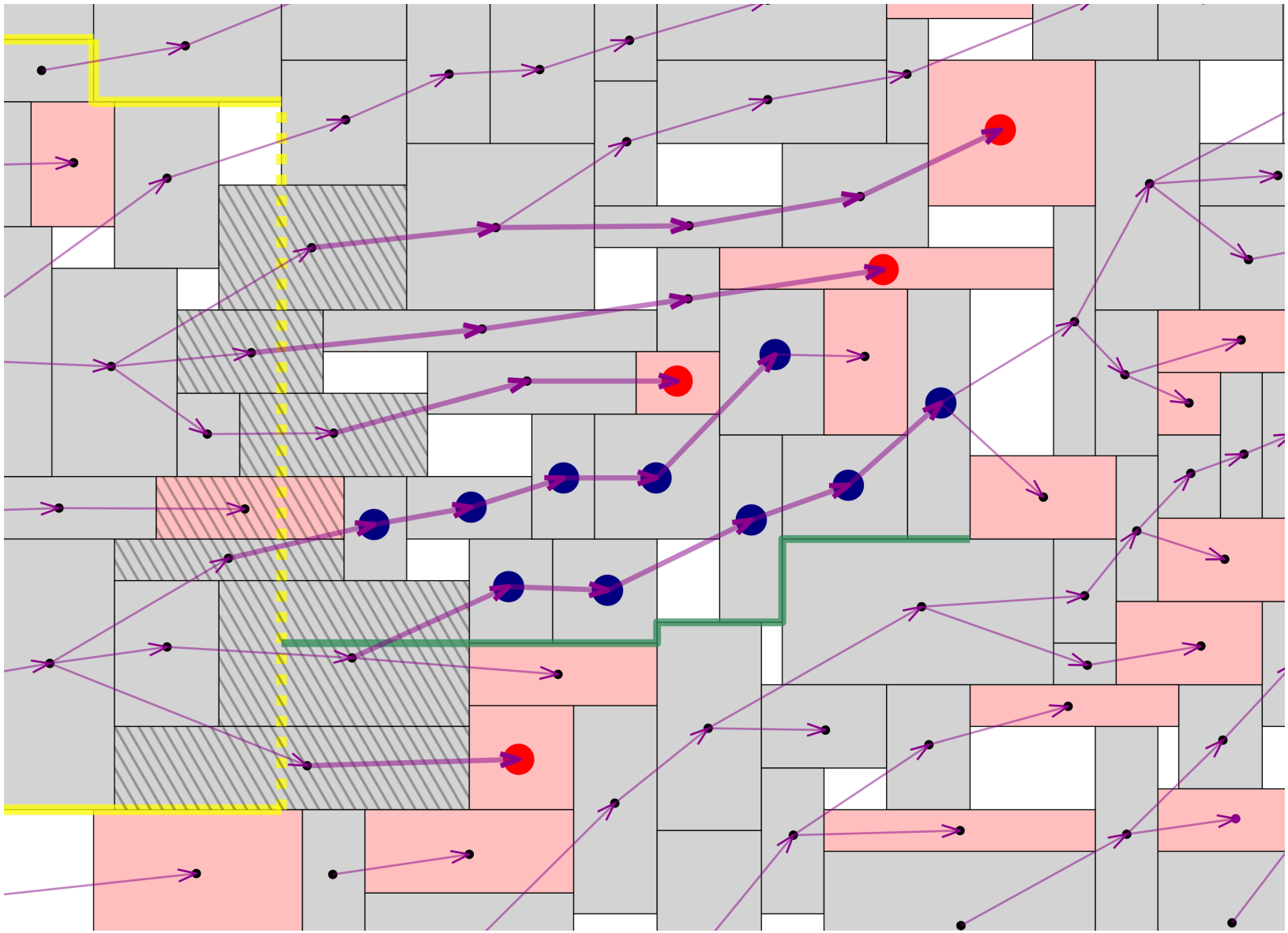
(and not nested)
(and not τ-protected)

- one nested rectangle



- $1/\varepsilon$ non-nested rectangles





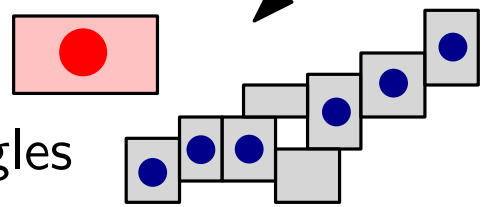
And all that charged rectangles will be protected by $(1/\varepsilon)$-fences.

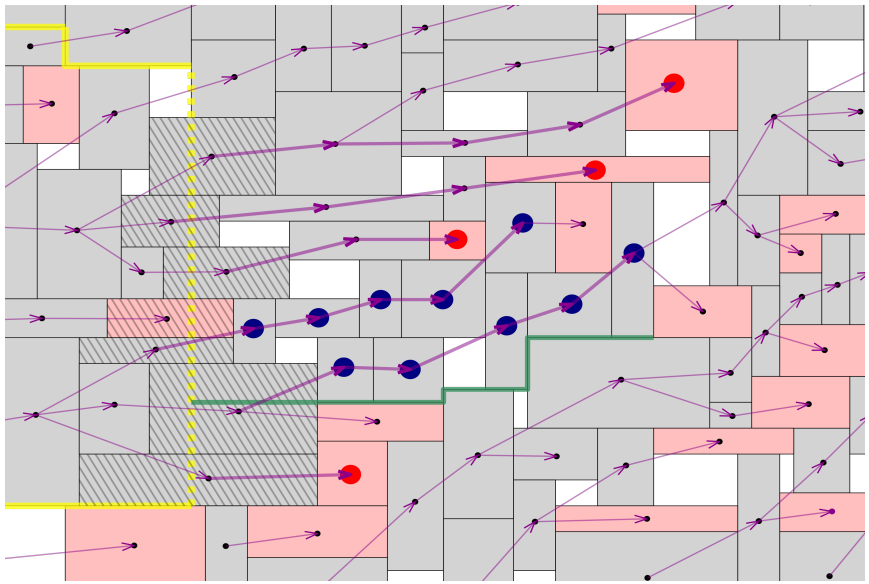
charging process:

A rectangle intersected charges either

(and not nested)
(and not τ-protected)

- one nested rectangle
- $1/\varepsilon$ non-nested rectangles



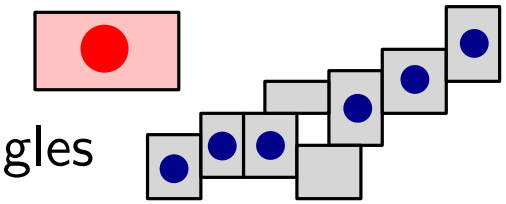


charging process:

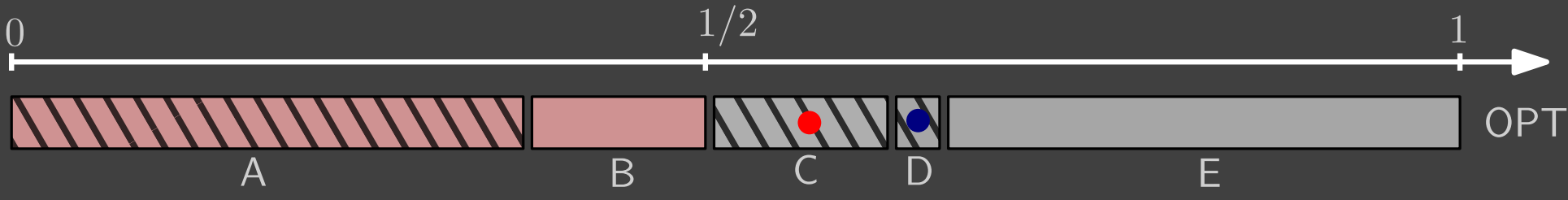
A rectangle intersected charges either

- one nested rectangle
- $1/\varepsilon$ non-nested rectangles

(and not nested)
(and not τ-protected)



Analysis of the Approximation ratio.



$$\begin{cases} A + B \le 1/2 \\ C \le B \\ D \le \varepsilon \end{cases} \longrightarrow B + E \ge 1/2 - \varepsilon$$

Lemma 19 (Partitioning Lemma)**.** *Let $\tau \geq 1$ be an integer. Let P be a simple axis-parallel polygon with at most $30\tau + 18$ edges of integral coordinates, such that P contains at least two rectangles from OPT'. Then there exists a sequence C of line segments with integral coordinates such that:*

(1) C is composed of at most $2\tau + 1$ horizontal or vertical line segments that are all contained in P.

(2) $P \setminus C$ has exactly two connected components, and each of them is a simple axis-parallel polygon with at most $30\tau + 18$ edges.

(3) There is a vertical line segment ℓ of C that intersects all the rectangles in $\mathrm{OPT}'(P)$ that are intersected by C.

(4) ℓ does not intersect any rectangle that is τ-protected in P.

similar as the 4-approximation

Lemma 19 (Partitioning Lemma). *Let $\tau \geq 1$ be an integer. Let P be a simple axis-parallel polygon with at most $30\tau + 18$ edges of integral coordinates, such that P contains at least two rectangles from OPT'. Then there exists a sequence C of line segments with integral coordinates such that:*

(1) C is composed of at most $2\tau + 1$ horizontal or vertical line segments that are all contained in P.

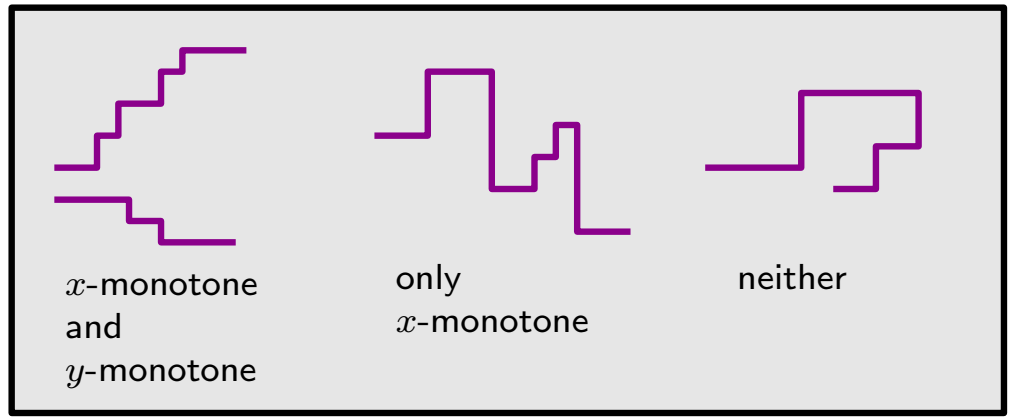
(2) $P \setminus C$ has exactly two connected components, and each of them is a simple axis-parallel polygon with at most $30\tau + 18$ edges.

(3) There is a vertical line segment ℓ of C that intersects all the rectangles in $\mathrm{OPT}'(P)$ that are intersected by C.

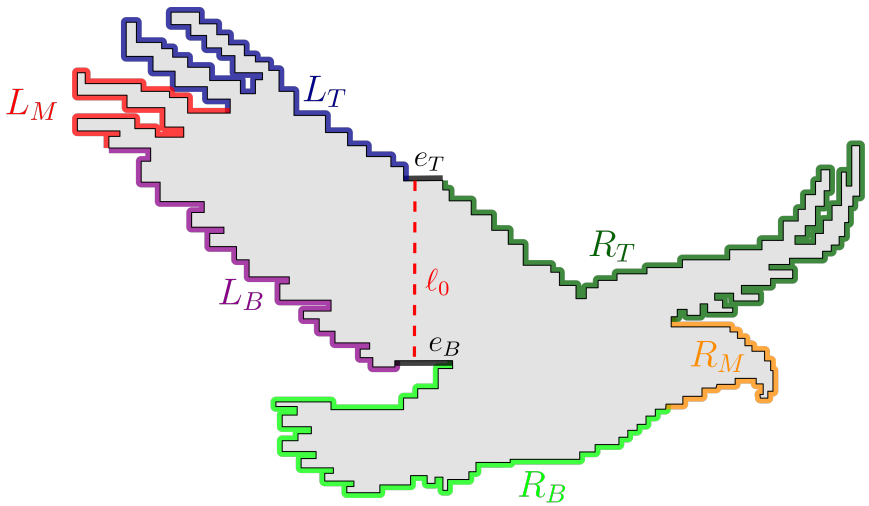
(4) ℓ does not intersect any rectangle that is τ-protected in P.

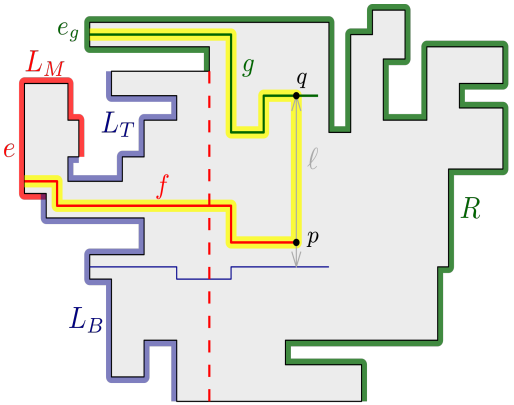
similar as the 4-approximation

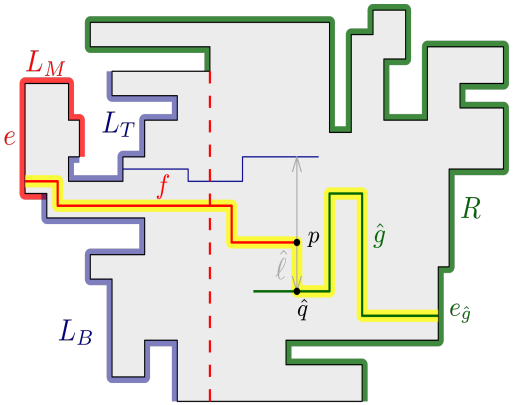
works more generally for only x-monotone fences



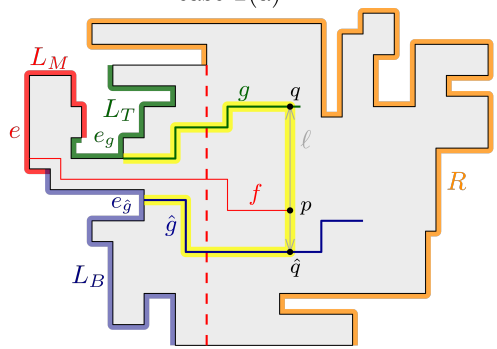
proof: you can read the paper :)



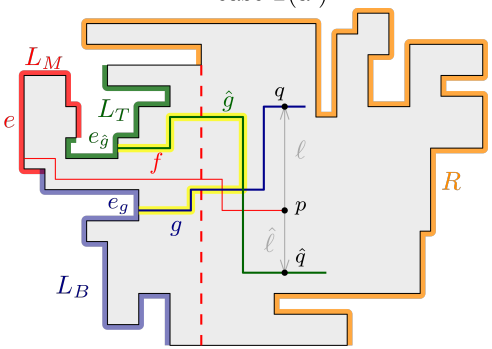


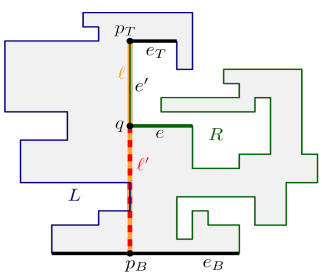


case 2(a)

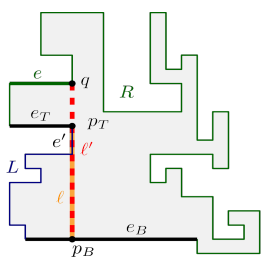


case 2(a')

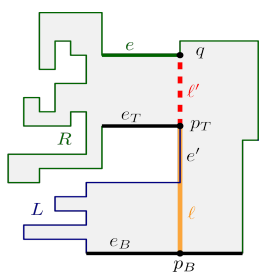




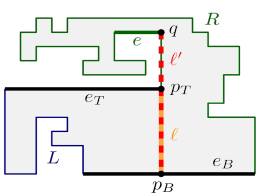
case 1(a)



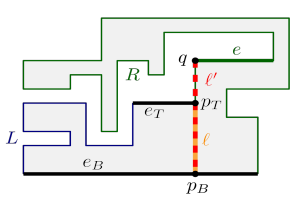
case 1(b)



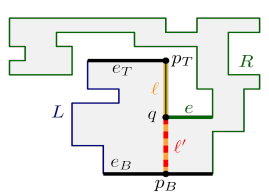
case 1(b)



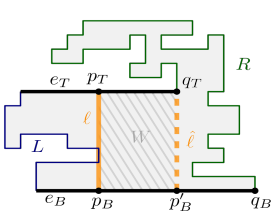
case 2



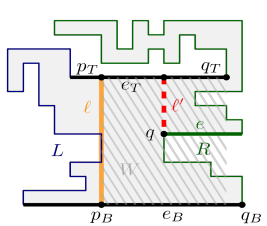
case 2



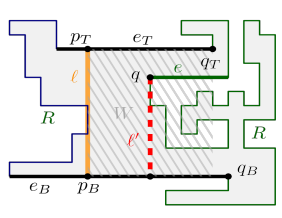
case 2



case 3

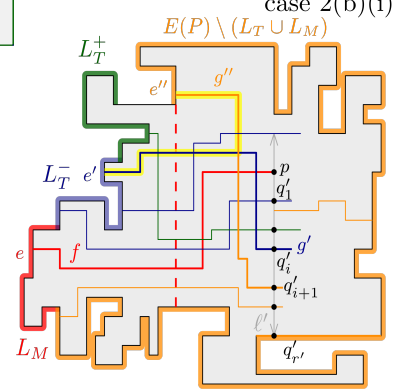


case 4

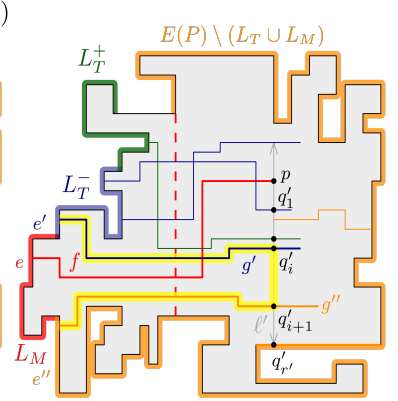


case 4

case 2(b)(i)

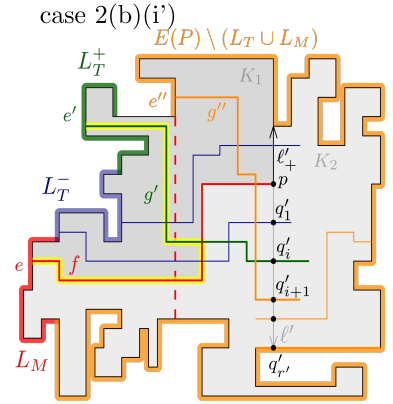


case 2(b)(ii)(A)

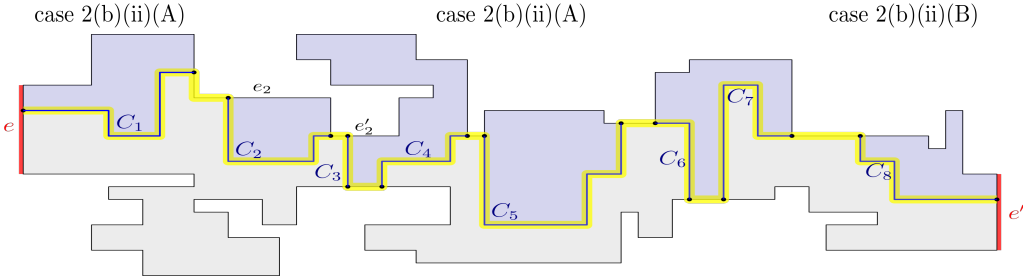


case 2(b)(ii)(A)

case 2(b)(i')



case 2(b)(ii)(B)



Open questions and future directions

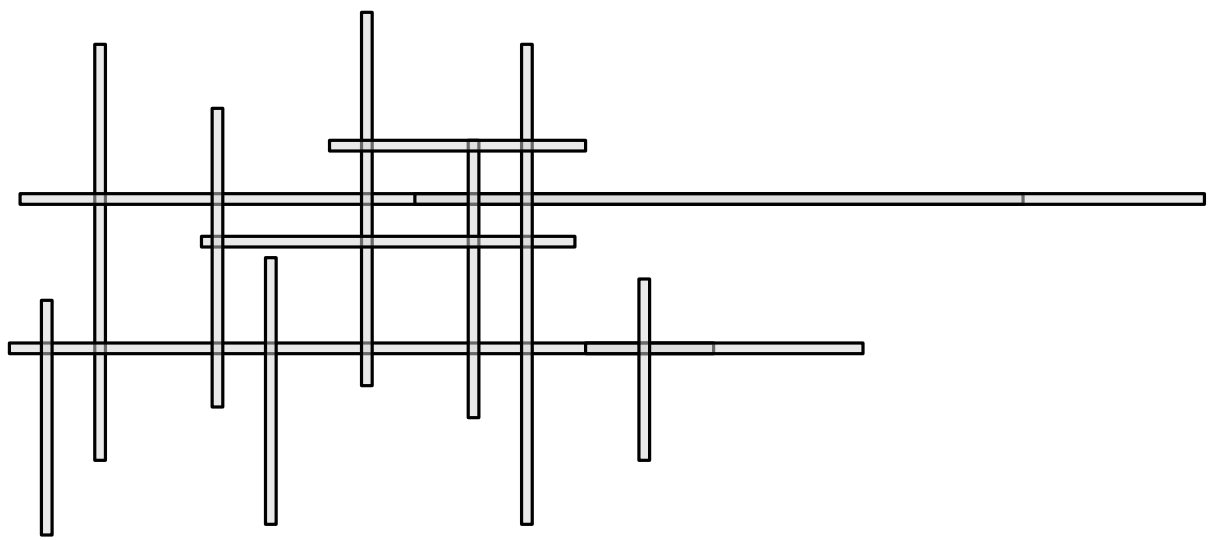
Beat $2 + \varepsilon$?

- Beat $2 + \varepsilon$?

Beat $2 + \varepsilon$?

- Beat $2 + \varepsilon$?

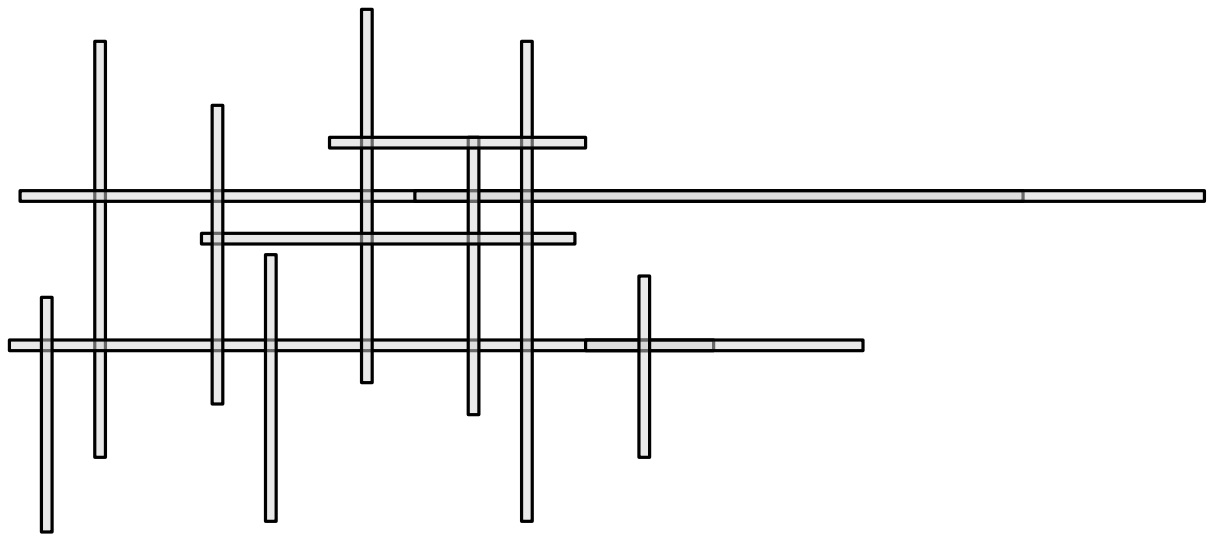
  Open problem.
  A (< 2)-approximation for the maximum independent set problem on vertical and horizontal line segments ?



Beat $2 + \varepsilon$?

- Beat $2 + \varepsilon$?

  Open problem.
  A (< 2)-approximation for the maximum independent set problem on vertical and horizontal line segments ?



- Pach-Tardos conjecture: a $O(1)$-approx with guillotine cuts ?

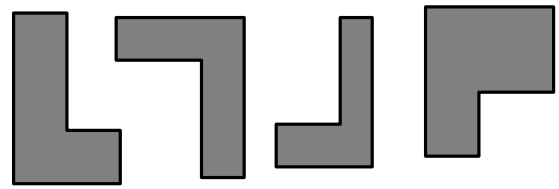
Generalisations and variants.

General polygons. best apx polynomial time : $O(n^{\varepsilon})$

→ solved: a $O(d)$-appx for convex polygons with at most d directions.

open

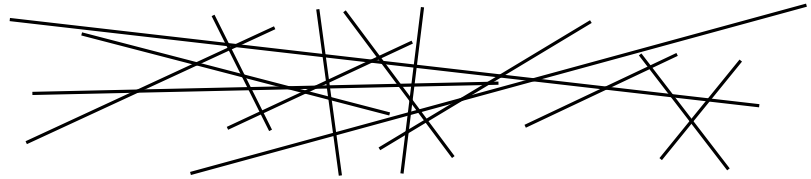
- A $O(1)$-approx for L-shapes ?



open

- Arbitrary line segments ?

  best: $O(n^{\varepsilon})$



---

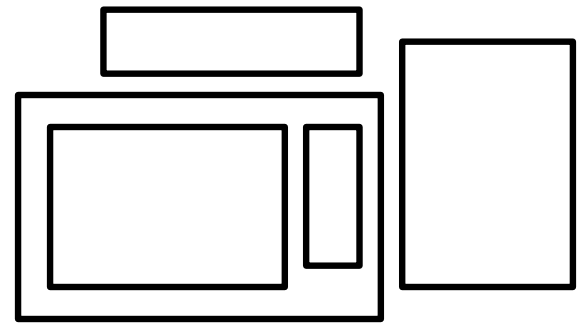
open

- A $O(1)$-approx for 3-dimentional rectangles ?

open

- A $O(1)$-approx for hollow rectangles ?

  → more general than the weighted case



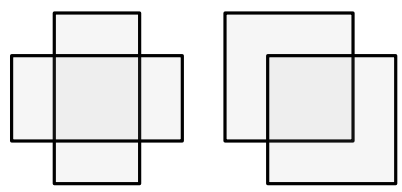
The weighted case

Our techniques don't extend to the weighted case.

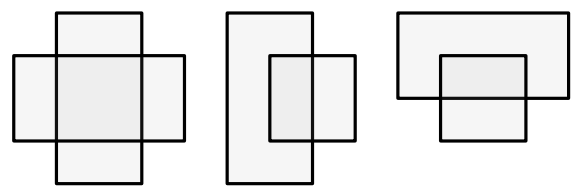
Best approximation: $O(\log \log n)$ [Chalermsook, Walczak, SODA'21]

Showed: χ is $O(\omega \log \omega)$, where χ is the chromatic number and ω the clique number; using LP rounding.

- conjecture: χ is $O(\omega)$.



open

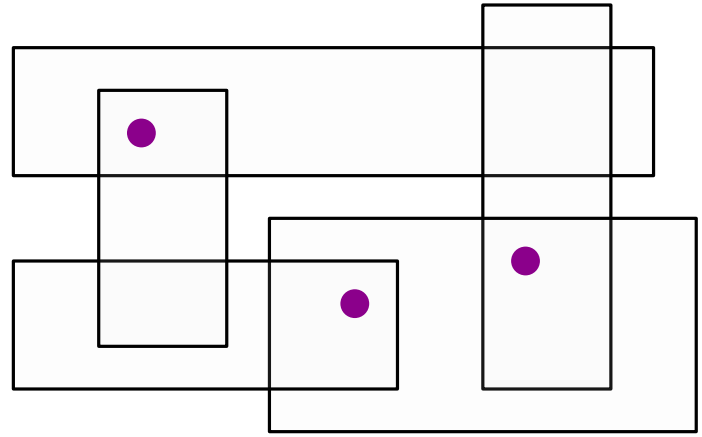


open

→ The conjecture, if true, will give $O(1)$-approximation for MWIS

- open: a $O(1)$-appx when weight $=$ area ?

Minimum Hitting Set



prop : MHS $\geq$ MIS

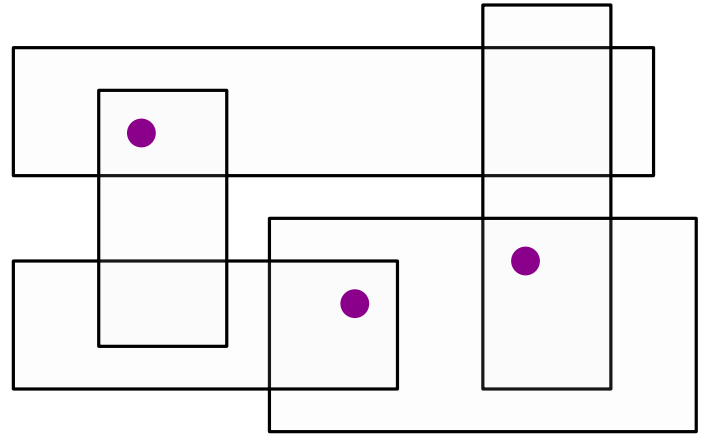
- Open: MHS ≤ 2 MIS ? (best: MHS $= O($ MIS $\log\log$ MIS$)$)

The conjecture, if true, implies a integrality gap of at most 2 for the clique-constrained LP.

$$\max \sum_R w_R x_R : \sum_{R \in Q} x_R \leq 1 \text{ for every clique } Q,\ 0 \leq x_R \leq 1.$$

- Open: the integrality gap of this LP is $O(1)$?
- Open: A $O(1)$-appx for MHS ? (Best: $O(\log\log(OPT))$)
- Open: An ϵ-net of size $O(1/\epsilon)$-appx for MHS ? (Best: $O(\frac{1}{\epsilon}\log\log(\frac{1}{\epsilon}))$)

Minimum Hitting Set



prop : MHS $\geq$ MIS

- Open: MHS ≤ 2 MIS ? (best: MHS $= O($ MIS $\log\log$ MIS$)$)

The conjecture, if true, implies a integrality gap of at most 2 for the clique-constrained LP.

$$\max \sum_R w_R x_R : \sum_{R \in Q} x_R \leq 1 \text{ for every clique } Q,\ 0 \leq x_R \leq 1.$$

- Open: the integrality gap of this LP is $O(1)$?
- Open: A $O(1)$-appx for MHS ? (Best: $O(\log\log(OPT))$)
- Open: An ϵ-net of size $O(1/\epsilon)$-appx for MHS ? (Best: $O(\frac{1}{\epsilon}\log\log(\frac{1}{\epsilon}))$)

Merci !