Applications of Combinatorial Geometry to Geometric Optimization

Sathish Govindarajan

Department of Computer Science and Automation
Indian Institute of Science, Bangalore

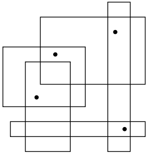
Expository lectures on Graph and Geometric Algorithms
Birla Institute of Technology and Science, Hyderabad
September 21-22, 2018

Geometric Problems

- $\mathcal{C}$ - Collection of simple geometric objects (rectangles, squares, circles, lines, points)

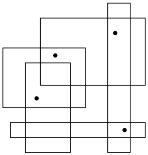
Geometric Problems

- $\mathcal{C}$ - Collection of simple geometric objects (rectangles, squares, circles, lines, points)



Geometric Problems

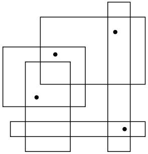
- $\mathcal{C}$ - Collection of simple geometric objects (rectangles, squares, circles, lines, points)



- Combinatorial Geometry: Understand the interactions among objects in $\mathcal{C}$
(Structural, Combinatorial questions)

Geometric Problems

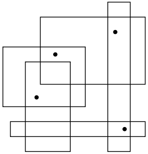
- $\mathcal{C}$ - Collection of simple geometric objects (rectangles, squares, circles, lines, points)



- Combinatorial Geometry: Understand the interactions among objects in $\mathcal{C}$ (Structural, Combinatorial questions)
- Computational Geometry: Design efficient algorithms (Computational questions)

Geometric Problems

- $\mathcal{C}$ - Collection of simple geometric objects (rectangles, squares, circles, lines, points)



- Combinatorial Geometry: Understand the interactions among objects in $\mathcal{C}$ (Structural, Combinatorial questions)
- Computational Geometry: Design efficient algorithms (Computational questions)
- Connection between these two areas

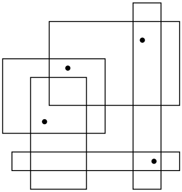
Overview of talk

- Introduction to Combinatorial Geometry
- Approximation algorithms in Geometric Optimization
  - Greedy based
  - Linear Programming based
  - Local Search based

Introduction to Combinatorial Geometry

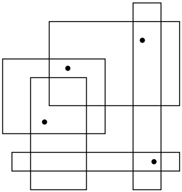
Combinatorial Geometry

- Structural, Combinatorial properties of a collection of geometric objects



Combinatorial Geometry

- Structural, Combinatorial properties of a collection of geometric objects



- Sub-areas
  - Geometric graphs, Incidences, Distance based problems, Arrangements, Epsilon nets, Geometric Discrepency
- Classical Theorems
  - Radon's Theorem, Caratheodory Theorem, Tverberg Theorem, Helly's Theorem, Centerpoint Theorem,

Combinatorial Geometry

- Nature of Questions (curious, intuitive, elementary)
- Elegant solutions
- Multiple proofs

Combinatorial Geometry - Illustrative Problem

Combinatorial Geometry - Illustrative Problem

A (Curious) Question

Combinatorial Geometry - Illustrative Problem

A (Curious) Question

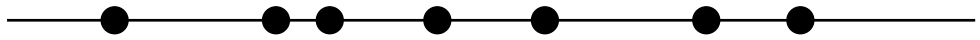
Can we construct a set of points P in the plane such that there is no line that passes through exactly two points of P?

Combinatorial Geometry - Illustrative Problem

A (Curious) Question

Can we construct a set of points P in the plane such that there is no line that passes through exactly two points of P?

- Yes. All points on a line.



Combinatorial Geometry - Illustrative Problem

Combinatorial Geometry - Illustrative Problem

Revised Question (Curious)

Combinatorial Geometry - Illustrative Problem

Revised Question (Curious)

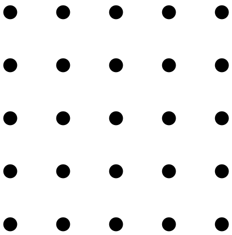
Can we construct a set of points P in the plane, not all in a line, such that there is no line that passes through exactly two points of P?

Combinatorial Geometry - Illustrative Problem

Revised Question (Curious)

Can we construct a set of points P in the plane, not all in a line, such that there is no line that passes through exactly two points of P?

- Yes. Integer grid.



Combinatorial Geometry - Illustrative Problem

Combinatorial Geometry - Illustrative Problem

Revised Question (Curious)

Combinatorial Geometry - Illustrative Problem

Revised Question (Curious)

Can we construct a finite set of points P in the plane, not all in a line, such that there is no line that passes through exactly two points of P?

Combinatorial Geometry - Illustrative Problem

Theorem (Sylvester-Gallai Theorem)

We cannot construct a finite set of points P in the plane, not all in a line, such that there is no line that passes through exactly two points of P

Combinatorial Geometry - Illustrative Problem

Theorem (Sylvester-Gallai Theorem)

We cannot construct a finite set of points P in the plane, not all in a line, such that there is no line that passes through exactly two points of P

- Posed by Sylvester in 1893 and reposed by Erdos in 1943
- Solved by Gallai in 1944
- Many alternate proofs
- Elegant proof by Kelly (Communicated by Coxeter)

Combinatorial Geometry - Illustrative Problem

Theorem (Sylvester-Gallai Theorem)

We cannot construct a finite set of points P in the plane, not all in a line, such that there is no line that passes through exactly two points of P

- Posed by Sylvester in 1893 and reposed by Erdos in 1943
- Solved by Gallai in 1944
- Many alternate proofs
- Elegant proof by Kelly (Communicated by Coxeter)
- Sylvester-Gallai in finite fields: Connections to showing circuit lower bounds

Combinatorial Geometry - Illustrative Problem

Theorem (Sylvester-Gallai Theorem)

Given any finite set of points P in the plane, not all in a line, there exists a line that passes through exactly two points of P

Proof:

Combinatorial Geometry - Illustrative Problem

Theorem (Sylvester-Gallai Theorem)

Given any finite set of points P in the plane, not all in a line, there exists a line that passes through exactly two points of P

Proof:

- Look at all the lines $\mathcal{L}$ that connect two points of P
- $d(l)$: distance of closest point from l, $l \in \mathcal{L}$
- L : line in $\mathcal{L}$ with smallest $d(l)$

Combinatorial Geometry - Illustrative Problem

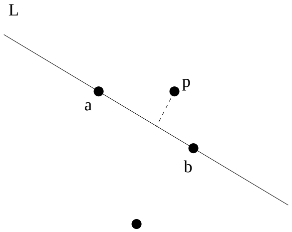
Theorem (Sylvester-Gallai Theorem)

Given any finite set of points P in the plane, not all in a line, there exists a line that passes through exactly two points of P

Proof:

- Look at all the lines $\mathcal{L}$ that connect two points of P
- $d(l)$: distance of closest point from l, $l \in \mathcal{L}$
- L : line in $\mathcal{L}$ with smallest $d(l)$

- Claim: L passes through exactly 2 points of P



Combinatorial Geometry - Illustrative Problem

Theorem (Sylvester-Gallai Theorem)

Given any finite set of points P in the plane, not all in a line, there exists a line that passes through exactly two points of P

Proof (contd):

- L : line in $\mathcal{L}$ with smallest $d(l)$
- Claim: L passes through exactly 2 points of P

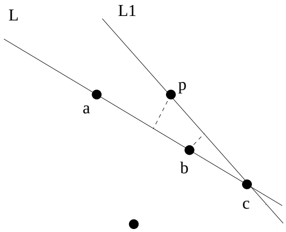
Combinatorial Geometry - Illustrative Problem

Theorem (Sylvester-Gallai Theorem)

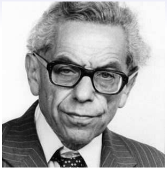
Given any finite set of points P in the plane, not all in a line, there exists a line that passes through exactly two points of P

Proof (contd):

- L : line in $\mathcal{L}$ with smallest $d(l)$
- Claim: L passes through exactly 2 points of P
- Suppose L passes through 3 or more points
- $\exists$ a line $L1 \in \mathcal{L}$ with smaller $d(l)$



Paul Erdos



- Fascination for elegant proofs (Proofs from the “BOOK”)
- 1500 papers
- About 500 co-authors
- Erdos number
- Biography: “My brain is open”

Geometric Optimization using Combinatorial Geometry

Geometric optimization using Combinatorial Geometry

- Geometric optimization problems
  - Set Cover, Hitting Set, Independent Set, etc

Geometric optimization using Combinatorial Geometry

- Geometric optimization problems
  - Set Cover, Hitting Set, Independent Set, etc

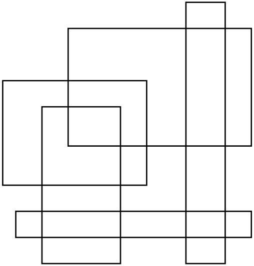
- Approximation algorithm using Combinatorial Geometry

Geometric optimization using Combinatorial Geometry

- Geometric optimization problems
    - Set Cover, Hitting Set, Independent Set, etc
- Approximation algorithm using Combinatorial Geometry
    - Connect the optimization problem to an appropriate combinatorial geometry problem
    - Solve this combinatorial geometry problem
    - Use this to get the approximate solution for the optimization problem

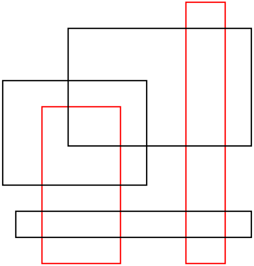
Independent Set

- S - set of m geometric objects
- Compute maximum sized subset $T \subseteq S$ such that all objects in T are "independent", i.e., $r \cap s = \emptyset, \forall r, s \in T$



Independent Set

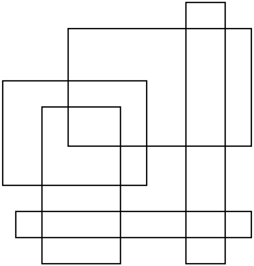
- S - set of m geometric objects
- Compute maximum sized subset $T \subseteq S$ such that all objects in T are "independent", i.e., $r \cap s = \emptyset, \forall r, s \in T$



- Motivation
  - Map labelling, data mining, Sensor and wireless networks, Unsplittable flow, . . .

Piercing Set

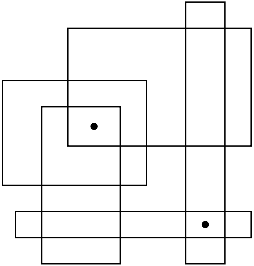
- S - set of m geometric objects
- Compute minimum sized subset $Q \subseteq R^2$ such that all objects in S are "pierced", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



- Motivation
  - Critical facility location, Robotics, Sensor and wireless networks, VLSI, . . .
- Special case of hitting set : $P = R^2$

Piercing Set

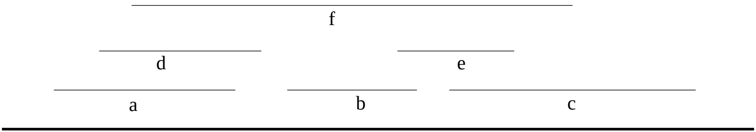
- S - set of m geometric objects
- Compute minimum sized subset $Q \subseteq R^2$ such that all objects in S are "pierced", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



Greedy based

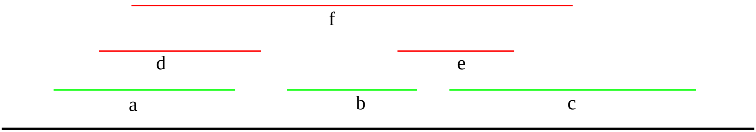
Independent Set of Intervals

- S - set of m intervals on the real line
- Compute maximum sized subset $T \subseteq S$ such that all objects in T are "independent", i.e., $r \cap s = \emptyset, \forall r, s \in T$



Independent Set of Intervals

- S - set of m intervals on the real line
- Compute maximum sized subset $T \subseteq S$ such that all objects in T are "independent", i.e., $r \cap s = \emptyset, \forall r, s \in T$



Greedy Algo: Independent Set of Intervals

- S - set of m intervals on the real line
- Greedy Algoithm

Greedy Algo: Independent Set of Intervals

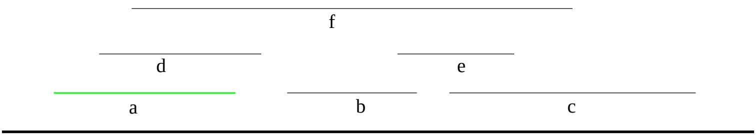
- S - set of m intervals on the real line
- Greedy Algoithm
  - Pick the first ending interval j in I

Greedy Algo: Independent Set of Intervals

- S - set of m intervals on the real line
- Greedy Algoithm
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j

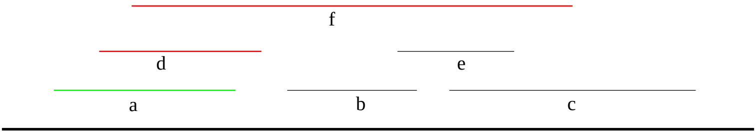
Greedy Algo: Independent Set of Intervals

- S - set of m intervals on the real line
- Greedy Algoithm
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left



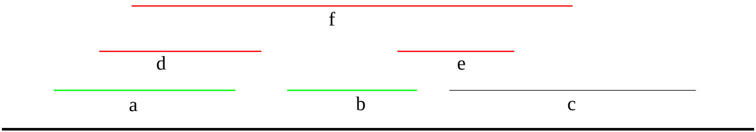
Greedy Algo: Independent Set of Intervals

- Greedy Algoithm
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left



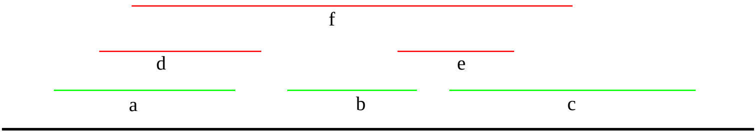
Greedy Algo: Independent Set of Intervals

- Greedy Algoithm
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left



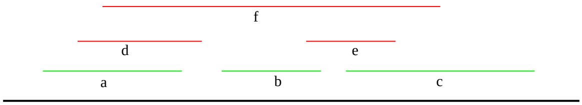
Greedy Algo: Independent Set of Intervals

- Greedy Algoithm
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left



Optimality of Greedy Algorithm

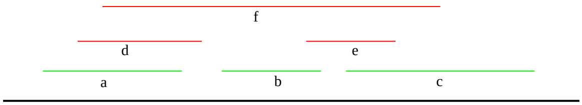
- Greedy Algoithm
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left



- Why is the Greedy Algorithm optimal?

Optimality of Greedy Algorithm

- Greedy Algoithm
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left



- Why is the Greedy Algorithm optimal?
- Textbook Proof
  - Compare Greedy solution with an optimal solution
  - At each iteration, greedy stays ahead of optimal (using induction)
  - At the end, greedy is not worse than optimal

Combinatorial Problem

- *S* - set of *n* intervals

- ν - Optimal Independent Set size of *S*
- τ - Optimal Piercing Set size of *S*

- Question: Relation between τ and ν
- $\tau \geq \nu$ (Lower bound)
- Question: Upper bound τ as a function of ν (Worst case over all possible *S*)

τ versus ν

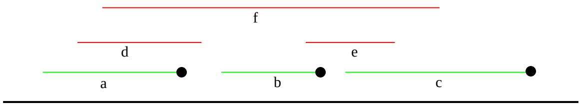
- Greedy Algoithm for Independent Set
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left

τ versus ν

- Greedy Algoithm for Independent Set
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left
- P : The end points of the intervals in I

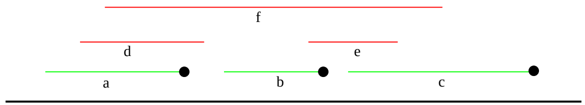
τ versus ν

- Greedy Algoithm for Independent Set
  - Pick the first ending interval j in I
  - Remove all the intervals that intersect with j
  - Repeat above steps until no intervals are left
- P : The end points of the intervals in I
- P is a piercing set for S



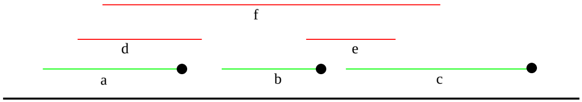
τ versus ν

- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



τ versus ν

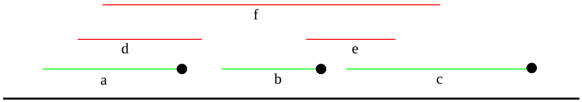
- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



- $\nu \geq |I|$ (ν is optimal independent set size)

τ versus ν

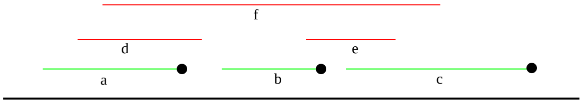
- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)

τ versus ν

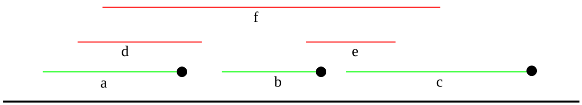
- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)
- $\nu \geq |I| = |P| \geq \tau$ (τ is optimal piercing set size)

τ versus ν

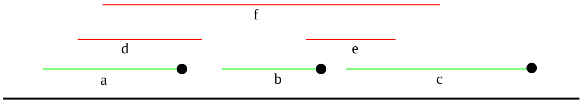
- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)
- $\nu \geq |I| = |P| \geq \tau$ (τ is optimal piercing set size)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$ (Lower bound)

τ versus ν

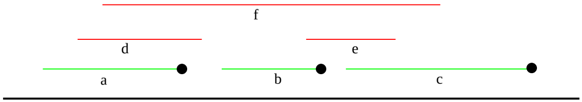
- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)
- $\nu \geq |I| = |P| \geq \tau$ (τ is optimal piercing set size)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$ (Lower bound)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$

τ versus ν

- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I

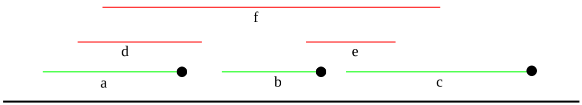


- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)
- $\nu \geq |I| = |P| \geq \tau$ (τ is optimal piercing set size)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$ (Lower bound)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$

- All inequalities are equality

τ versus ν

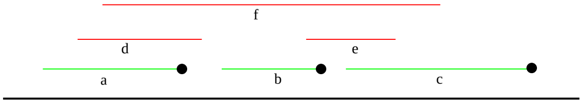
- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)
- $\nu \geq |I| = |P| \geq \tau$ (τ is optimal piercing set size)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$ (Lower bound)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$
- All inequalities are equality
- $\nu = |I|$ (I is optimal independent set)

τ versus ν

- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



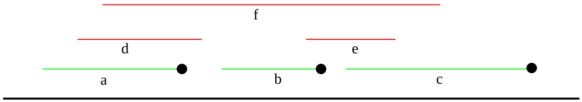
- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)
- $\nu \geq |I| = |P| \geq \tau$ (τ is optimal piercing set size)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$ (Lower bound)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$

- All inequalities are equality

- $\nu = |I|$ (I is optimal independent set)
- $|P| = \tau$ (P is optimal piercing set)

τ versus ν

- I : Greedy Independent Set for intervals
- P : The end points of the intervals in I
- P is a piercing set for I



- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|$ (By construction)
- $\nu \geq |I| = |P| \geq \tau$ (τ is optimal piercing set size)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$ (Lower bound)
- $\nu \geq |I| = |P| \geq \tau \geq \nu$

- All inequalities are equality

- $\nu = |I|$ (I is optimal independent set)
- $|P| = \tau$ (P is optimal piercing set)
- $\tau = \nu$

Greedy Algo: Independent Set of Squares

- S - set of axis parallel squares in the plane
- Greedy Algoithm

Greedy Algo: Independent Set of Squares

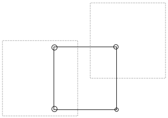
- S - set of axis parallel squares in the plane
- Greedy Algoithm
  - Pick the smallest square s in I
  - Pick the 4 corners of square s in P
  - Remove all the intervals that intersect with square s
  - Repeat above steps until no squares are left

Greedy Algo: Independent Set of Squares

- *S* - set of axis parallel squares in the plane
- Greedy Algoithm
  - Pick the smallest square s in I
  - Pick the 4 corners of square s in P
  - Remove all the intervals that intersect with square s
  - Repeat above steps until no squares are left
- I is an independent set of S

Greedy Algo: Independent Set of Squares

- S - set of axis parallel squares in the plane
- Greedy Algoithm
  - Pick the smallest square s in I
  - Pick the 4 corners of square s in P
  - Remove all the intervals that intersect with square s
  - Repeat above steps until no squares are left
- I is an independent set of S
- P is a piercing set for S



τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$

τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$
- $\nu \geq |I|$ (ν is optimal independent set size)

τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$

- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|/4$ (By construction)

τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$

- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|/4$ (By construction)
- $\nu \geq |I| = |P|/4 \geq \tau/4$ (τ is optimal piercing set size)

τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$

- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|/4$ (By construction)
- $\nu \geq |I| = |P|/4 \geq \tau/4$ (τ is optimal piercing set size)
- $\tau \geq \nu \geq |I| = |P|/4 \geq \tau/4 \geq \nu/4$ (Lower bound)

τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$

- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|/4$ (By construction)
- $\nu \geq |I| = |P|/4 \geq \tau/4$ (τ is optimal piercing set size)
- $\tau \geq \nu \geq |I| = |P|/4 \geq \tau/4 \geq \nu/4$ (Lower bound)

- $|I| \geq \nu/4$ (I is 4-approximate independent set)

τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$

- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|/4$ (By construction)
- $\nu \geq |I| = |P|/4 \geq \tau/4$ (τ is optimal piercing set size)
- $\tau \geq \nu \geq |I| = |P|/4 \geq \tau/4 \geq \nu/4$ (Lower bound)

- $|I| \geq \nu/4$ (I is 4-approximate independent set)
- $|P| \leq 4 * \tau$ (P is 4-approximate piercing set)

τ versus ν for squares

- I : Greedy Independent Set for squares
- P : Greedy piercing set for squares
- $|P| = 4 * |I|$

- $\nu \geq |I|$ (ν is optimal independent set size)
- $\nu \geq |I| = |P|/4$ (By construction)
- $\nu \geq |I| = |P|/4 \geq \tau/4$ (τ is optimal piercing set size)
- $\tau \geq \nu \geq |I| = |P|/4 \geq \tau/4 \geq \nu/4$ (Lower bound)

- $|I| \geq \nu/4$ (I is 4-approximate independent set)
- $|P| \leq 4 * \tau$ (P is 4-approximate piercing set)
- $\tau \leq 4 * \nu$

τ versus ν problem

τ versus ν problem

- Similar argument for disks (Exercise)

τ versus ν problem

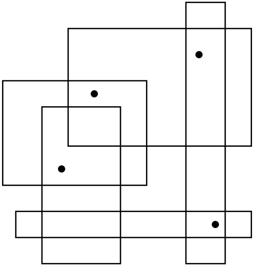
- Similar argument for disks (Exercise)
- Similar argument for fat objects

Linear Programming based Algorithms
Local Search based Algorithms

Hitting Set

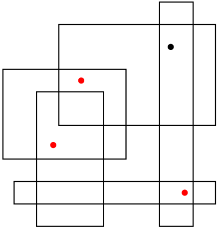
- P - set of n points
- S - set of m geometric objects

- Compute minimum sized subset $Q \subseteq P$ such that all objects in S are "hit", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



Hitting Set

- P - set of n points
- S - set of m geometric objects
- Compute minimum sized subset $Q \subseteq P$ such that all objects in S are "hit", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



- Motivation
  - Critical facility location, Robotics, Sensor and wireless networks, VLSI, . . .

Hitting Set - Known results

- NP-hard for unit squares

Hitting Set - Known results

- NP-hard for unit squares
- PTAS approximation for squares, disks [MR '09]

Hitting Set - Known results

- NP-hard for unit squares
- PTAS approximation for squares, disks [MR '09]
    - Local Search algorithm
      (Analysis: Existence of planar bipartite graph)

Hitting Set - Known results

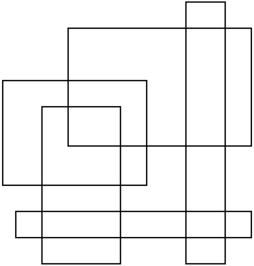
- NP-hard for unit squares
- PTAS approximation for squares, disks [MR ’09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)
- $\log \log n$-approximation for rectangles [AES09, BG94]

Hitting Set - Known results

- NP-hard for unit squares
- PTAS approximation for squares, disks [MR '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)
- $\log \log n$-approximation for rectangles [AES09, BG94]
  - Linear Programming based algorithm
    (Using bounds for Epsilon-nets)

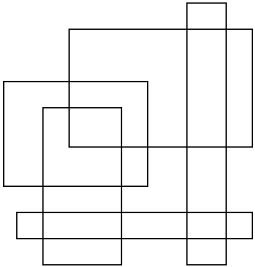
Piercing Set

- S - set of m geometric objects
- Compute minimum sized subset $Q \subseteq R^2$ such that all objects in S are "pierced", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



Piercing Set

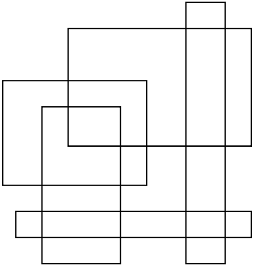
- S - set of m geometric objects
- Compute minimum sized subset $Q \subseteq R^2$ such that all objects in S are "pierced", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



- Special case of hitting set : $P = R^2$

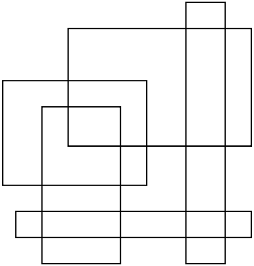
Independent Set

- S - set of m geometric objects
- Compute maximum sized subset $T \subseteq S$ such that all objects in T are "independent", i.e., $r \cap s = \emptyset, \forall r, s \in T$



Independent Set

- S - set of m geometric objects
- Compute maximum sized subset $T \subseteq S$ such that all objects in T are "independent", i.e., $r \cap s = \emptyset, \forall r, s \in T$



- Special case of discrete independent set : $P = R^2$

Independent Set - Known results

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)
- $\log\log n$-approximation for rectangles [Chalmersook '11]

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)
- $\log\log n$-approximation for rectangles [Chalmersook '11]
  - Linear Programming based algorithm
    (Using bounds for a coloring problem)

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)
- $\log \log n$-approximation for rectangles [Chalmersook '11]
  - Linear Programming based algorithm
    (Using bounds for a coloring problem)
- QPTAS for rectangles [Adamazek and Wiese '13]

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)
- $\log\log n$-approximation for rectangles [Chalmersook '11]
  - Linear Programming based algorithm
    (Using bounds for a coloring problem)
- QPTAS for rectangles [Adamazek and Wiese '13]
- Few results for discrete independent set

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)
- $\log\log n$-approximation for rectangles [Chalmersook '11]
  - Linear Programming based algorithm
    (Using bounds for a coloring problem)
- QPTAS for rectangles [Adamazek and Wiese '13]
- Few results for discrete independent set
  - Constant factor approx. for squares, disks [CP12, EPR09]

Independent Set - Known results

- PTAS approximation for squares, disks [Chan and Har-Peled '09]
  - Local Search algorithm
    (Analysis: Existence of planar bipartite graph)

- $\log \log n$-approximation for rectangles [Chalmersook '11]
  - Linear Programming based algorithm
    (Using bounds for a coloring problem)

- QPTAS for rectangles [Adamazek and Wiese '13]

- Few results for discrete independent set
  - Constant factor approx. for squares, disks [CP12, EPR09]
  - Poly-time solvable for skyline rectangles [CG14]

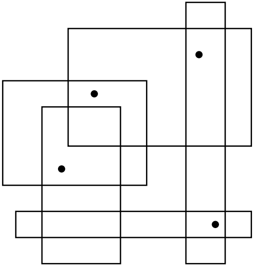
Linear Programming based Approximation

Hitting Set and Set Cover using Epsilon Nets

Hitting Set

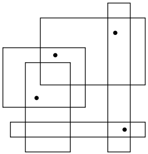
- P - set of n points
- S - set of m geometric objects

- Compute minimum sized subset $Q \subseteq P$ such that all objects in S are "hit", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



Linear Programming for Hitting Set

- P - set of n points, S - set of m geometric objects
- Compute minimum sized subset $Q \subseteq P$ such that all objects in S are "hit", i.e., $Q \cap r \neq \emptyset, \forall r \in S$



- Indicator variable: x_i for each point $p_i \in P$

$$Minimize \sum_{i=1}^{n} x_i$$

$$\sum_{j, p_j \in O_i} x_j \geq 1 \forall i = 1 \ldots m$$

$$x_i = \{0, 1\} \forall i = 1 \ldots n$$

Combinatorial problem

- Epsilon Nets: Combinatorial problem related to Hitting Set

Epsilon Nets : Formal definition

Definition

Let P be a set of n points in the plane. $N \subset P$ is a ϵ*-net* for a family of geometric objects $\mathcal{S}$ if $S \cap N \neq \emptyset$ for any $S \in \mathcal{S}$ such that $|S \cap P| > \epsilon n$.

- Epsilon nets: Hitting set for **dense** objects

Epsilon Nets

Theorem (ϵ-net theorem (Haussler,Welzl))

Let P be a set of n points and S be a set of geometric objects. Then there exists an ϵ-net of size $O(\frac{1}{\epsilon}\log\frac{1}{\epsilon})$.

- Epsilon nets of constant size (independent of n)

Hitting Set using Epsilon Nets

- Epsilon nets: Hitting set for **dense** objects, i.e., each object has $> \epsilon n$ points

Hitting Set using Epsilon Nets

- Epsilon nets: Hitting set for **dense** objects, i.e., each object has $> \epsilon n$ points
- Algorithm [Bronimann, Goodrich '94]

Hitting Set using Epsilon Nets

- Epsilon nets: Hitting set for **dense** objects, i.e., each object has $> \epsilon n$ points
- Algorithm [Bronimann, Goodrich '94]
  - Find an appropriate small ϵ', $0 < \epsilon' < 1$
  - Find weights w_i for each point p_i such that each object has $> \epsilon'$ fraction of the total weight of points

Hitting Set using Epsilon Nets

- Epsilon nets: Hitting set for **dense** objects, i.e., each object has $> \epsilon n$ points
- Algorithm [Bronimann, Goodrich '94]
  - Find an appropriate small ϵ', $0 < \epsilon' < 1$
  - Find weights w_i for each point p_i such that each object has $> \epsilon'$ fraction of the total weight of points
  - ϵ'-net gives a feasible hitting set

Hitting Set using Epsilon Nets

- Finding ϵ' and w_i's

Hitting Set using Epsilon Nets

- Finding ϵ' and w_i's
  - Solving a Linear Program [Evan et al, 2005]

Hitting Set using Epsilon Nets

- Finding ϵ' and w_i's
  - Solving a Linear Program [Evan et al, 2005]
- Suppose Epsilon-nets of size $O(\frac{1}{\epsilon}f(\frac{1}{\epsilon}))$ exists

Hitting Set using Epsilon Nets

- Finding ϵ' and w_i's
  - Solving a Linear Program [Evan et al, 2005]
- Suppose Epsilon-nets of size $O(\frac{1}{\epsilon}f(\frac{1}{\epsilon}))$ exists
- Quality of Solution

Hitting Set using Epsilon Nets

- Finding ϵ' and w_i's
  - Solving a Linear Program [Evan et al, 2005]
- Suppose Epsilon-nets of size $O(\frac{1}{\epsilon}f(\frac{1}{\epsilon}))$ exists
- Quality of Solution
  - Observation: $\epsilon' \leq \frac{1}{OPT}$
  - Solution size: $O(OPT * f(OPT)$
  - $f(OPT)$ approximation

Epsilon Nets: Results

Epsilon Nets: Results

- ϵ-nets of size $O(\frac{1}{\epsilon})$ exist for half spaces in $\mathbb{R}^2$ and $\mathbb{R}^3$.[Komlos, Pach, Woeginger]

Epsilon Nets: Results

- ϵ-nets of size $O(\frac{1}{\epsilon})$ exist for half spaces in $\mathbb{R}^2$ and $\mathbb{R}^3$.[Komlos, Pach, Woeginger]

- ϵ-nets of size $O(\frac{1}{\epsilon})$ exist for squares, disks.[Matousek, Seidel,Welzl]
- $O(1)$-approximation for squares, disks

Epsilon Nets: Results

- ϵ-nets of size $O(\frac{1}{\epsilon})$ exist for half spaces in $\mathbb{R}^2$ and $\mathbb{R}^3$.[Komlos, Pach, Woeginger]

- ϵ-nets of size $O(\frac{1}{\epsilon})$ exist for squares, disks.[Matousek, Seidel,Welzl]
- $O(1)$-approximation for squares, disks

- ϵ-nets of size $O(\frac{1}{\epsilon}\log\log\frac{1}{\epsilon})$ exist for axis parallel rectangles in $\mathbb{R}^2$[Aronov, Ezra, Sharir]
- $\log\log n$-approximation for rectangles [AES09, BG94]

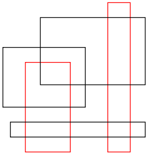
Set Cover using Dual Epsilon Nets

- Set Cover: Dual of Hitting Set
- Set Cover solved using Dual Epsilon Nets [Bronimann, Goodrich '94]
- Set Cover for Disks, Squares
  - Disks, Squares have linear Union Complexity
  - Disks, Squares have linear Dual Epsilon nets [Clarkson, Varadarajan]
  - Set Cover for Disks, Squares have $O(1)$ approximation

Independent Set using Coloring problem

Linear Programming for Independent Set

- S - set of m geometric objects
- Compute maximum sized subset $T \subseteq S$ such that all objects in T are "independent", i.e., $r \cap s = \emptyset, \forall r, s \in T$



- P: Place a point in each distinct region
- Indicator variable: x_i for each object $s_i \in S$

$$Maximize \sum_{i=1}^{n} x_i$$

$$\sum_{j, p_i \in O_j} x_j \leq 1 \forall i = 1 \dots n$$

$$x_i = \{0, 1\} \forall i = 1 \dots m$$

Combinatorial problem related to Independent Set

- Consider the Intersection graph of the geometric objects

Combinatorial problem related to Independent Set

- Consider the Intersection graph of the geometric objects
- Coloring problem: Chromatic number as a function of Clique number

Combinatorial problem related to Independent Set

- Consider the Intersection graph of the geometric objects
- Coloring problem: Chromatic number as a function of Clique number
- Function f: $\chi = \omega * f(\omega)$

Independent Set using Coloring problem

Independent Set using Coloring problem

- Convert the problem to coloring problem using Linear Programming [CP09, C11]

Independent Set using Coloring problem

- Convert the problem to coloring problem using Linear Programming [CP09, C11]
- Algorithm for the coloring problem with good coloring bounds

Independent Set using Coloring problem

- Convert the problem to coloring problem using Linear Programming [CP09, C11]
- Algorithm for the coloring problem with good coloring bounds
- Return the largest color class as the independent set solution

Independent Set using Coloring problem

- Convert the problem to coloring problem using Linear Programming [CP09, C11]
- Algorithm for the coloring problem with good coloring bounds
- Return the largest color class as the independent set solution
- The coloring bound gives the approximation factor

Coloring problem: Results

- Consider the Intersection graph of the geometric objects
- Coloring problem: Chromatic number as a function of Clique number

- Square, Disks: $\chi = O(\omega)$
- Rectangles(no containment): $\chi = O(\omega \log \omega)$
- Rectangles: $\chi = O(\omega^2)$

Local Search based Approximation

Local Search

- Local Search Paradigm
  - Start with any feasible solution
  - Make improvements to the current solution by making local changes
  - Stop when local improvement is not possible

Local Search

- Local Search Paradigm
  - Start with any feasible solution
  - Make improvements to the current solution by making local changes
  - Stop when local improvement is not possible
- Popular heuristic

Local Search

- Local Search Paradigm
  - Start with any feasible solution
  - Make improvements to the current solution by making local changes
  - Stop when local improvement is not possible
- Popular heuristic
- Challenge: Theoretical guarantee on solution quality

Local Search

- Local Search Paradigm
  - Start with any feasible solution
  - Make improvements to the current solution by making local changes
  - Stop when local improvement is not possible
- Popular heuristic
- Challenge: Theoretical guarantee on solution quality
- Recently, many local search based approximation algorithms
  - Innovative analysis techniques

Local Search: Geometric Results Overview

- Clustering
  - Facility location [Charikar et al ’05, Cohen et al ’15]
  - k-median [Korupolu et al ’00, Arya et al ’04]
  - k-means [Kanungo et al ’04, Friggstad et al ’16]
- Packing and Covering
  - Hitting Set, Set Cover [Mustafa et al ’09, Govindarajan et al ’16]
  - Independent Set [Chan et al ’09, Ashner et al ’13]
  - Dominating Set [Gibson et al ’10]
  - Terrain guarding [Krohn et al ’14]

Local Search for Hitting Set

- Initial Solution $T = P$ (all points)

Local Search for Hitting Set

- Initial Solution $T = P$ (all points)
- Make improvements to T by making local changes

Local Search for Hitting Set

- Initial Solution $T = P$ (all points)
- Make improvements to T by making local changes
  - Do a $(k, k-1)$-swap in T if it is feasibile

Local Search for Hitting Set

- Initial Solution $T = P$ (all points)
- Make improvements to T by making local changes
  - Do a $(k, k-1)$-swap in T if it is feasibile
- Stop when none of the $(k, k-1)$-swaps is feasible

Local Search for Hitting Set

- Initial Solution $T = P$ (all points)
- Make improvements to T by making local changes
  - Do a $(k, k-1)$-swap in T if it is feasibile
- Stop when none of the $(k, k-1)$-swaps is feasible
- Running time: $O(n^{O(k)})$
  (k is constant)

Analysis Framework [Aschner et al. '13]

- Objective function - Size of the solution

Analysis Framework [Aschner et al. '13]

- Objective function - Size of the solution
- O - optmial solution, A - local search solution
- Bipartite graph on $O \cup A$ having
  - Balanced, sub-linear separator (Planarity)
  - Local exchange property $((A \setminus A') \cup N(A')$ is feasible soln.)

Analysis Framework [Aschner et al. '13]

- Objective function - Size of the solution
- O - optmial solution, A - local search solution
- Bipartite graph on $O \cup A$ having
  - Balanced, sub-linear separator (Planarity)
  - Local exchange property $((A \setminus A') \cup N(A')$ is feasible soln.)
- Set local search parameter $k = \frac{1}{\epsilon^2}$

Analysis Framework [Aschner et al. '13]

- Objective function - Size of the solution
- O - optmial solution, A - local search solution
- Bipartite graph on $O \cup A$ having
  - Balanced, sub-linear separator (Planarity)
  - Local exchange property ($(A \setminus A') \cup N(A')$ is feasible soln.)
- Set local search parameter $k = \frac{1}{\epsilon^2}$
- $|A| \leq (1+\epsilon)|O|$ (PTAS approximation algorithm)

Local Search for Hitting Set and Independent Set

- PTAS for Hitting Set of Pseudo-Disks [Mustafa and Ray '09]

Local Search for Hitting Set and Independent Set

- PTAS for Hitting Set of Pseudo-Disks [Mustafa and Ray '09]
  - Ashner Framework Graph: Delaunay triangulation on $A \cup O$ (Planar, Induced subgraph within a disk is connected)

Local Search for Hitting Set and Independent Set

- PTAS for Hitting Set of Pseudo-Disks [Mustafa and Ray '09]
  - Ashner Framework Graph: Delaunay triangulation on $A \cup O$ (Planar, Induced subgraph within a disk is connected)
- PTAS for Independent Set of Disks, Sqaures [Chan and Har-Peled '09]
  - Ashner Framework Graph: Intersection graph on $A \cup O$ (Planar)

Open Problems - Local Search Analysis

- Weighted Hitting Set and Independent Set
- Unweighted Demand Hitting Set and Demand Set Cover
- Discrete Independent Set

Questions?