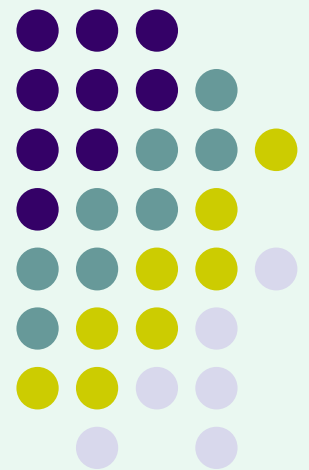


Randomized Techniques in Geometry

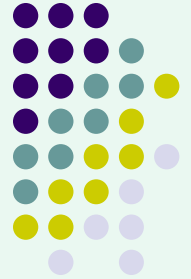
Yogish Sabharwal

IBM India Research Lab

Workshop on Introduction to Geometric Algorithms, IIT Kharagpur, 2008

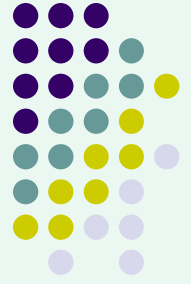


Contents

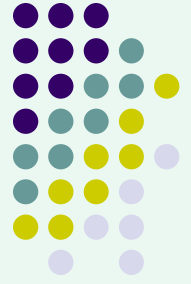


- Introduction
- Randomized Incremental Construction
 - Quick-sort
 - Linear programming
- Random Sampling
 - Skip lists
 - Trapezoidal decompositions
- Clustering Framework

Primary Sources (for this talk)



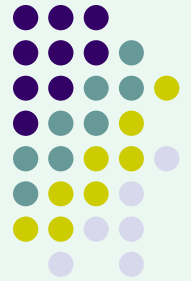
- Randomized Techniques in Computational Geometry, A Survey Talk
Sandeep Sen
<http://www.cse.iitd.ernet.in/~ssen/cs852/Rand.pdf>
- Computational Geometry: An Introduction through Randomized Algorithms
K. Mulmuley



Kinds of Problems

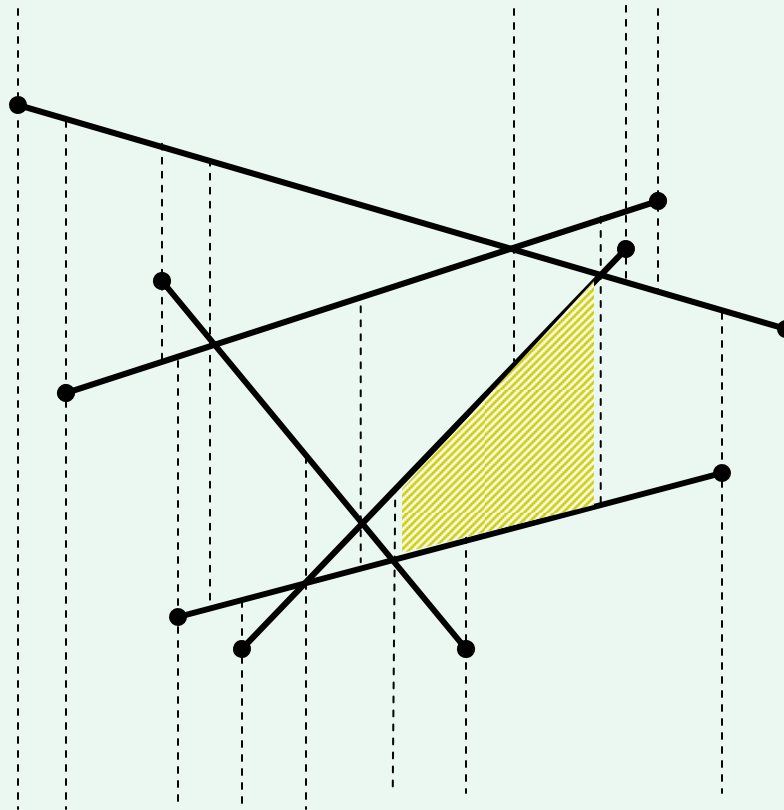
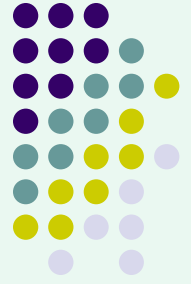
- Range Searching
- Ray Shooting/
Point location
- Planar Partitioning
- Convex hulls
- Linear programming
- Nearest Neighbor
- Triangulation
- Hidden Surface
- Levels of arrangement
- Diameter/Width
- Euclidean minimum
spanning tree

Randomization in Computational Geometry

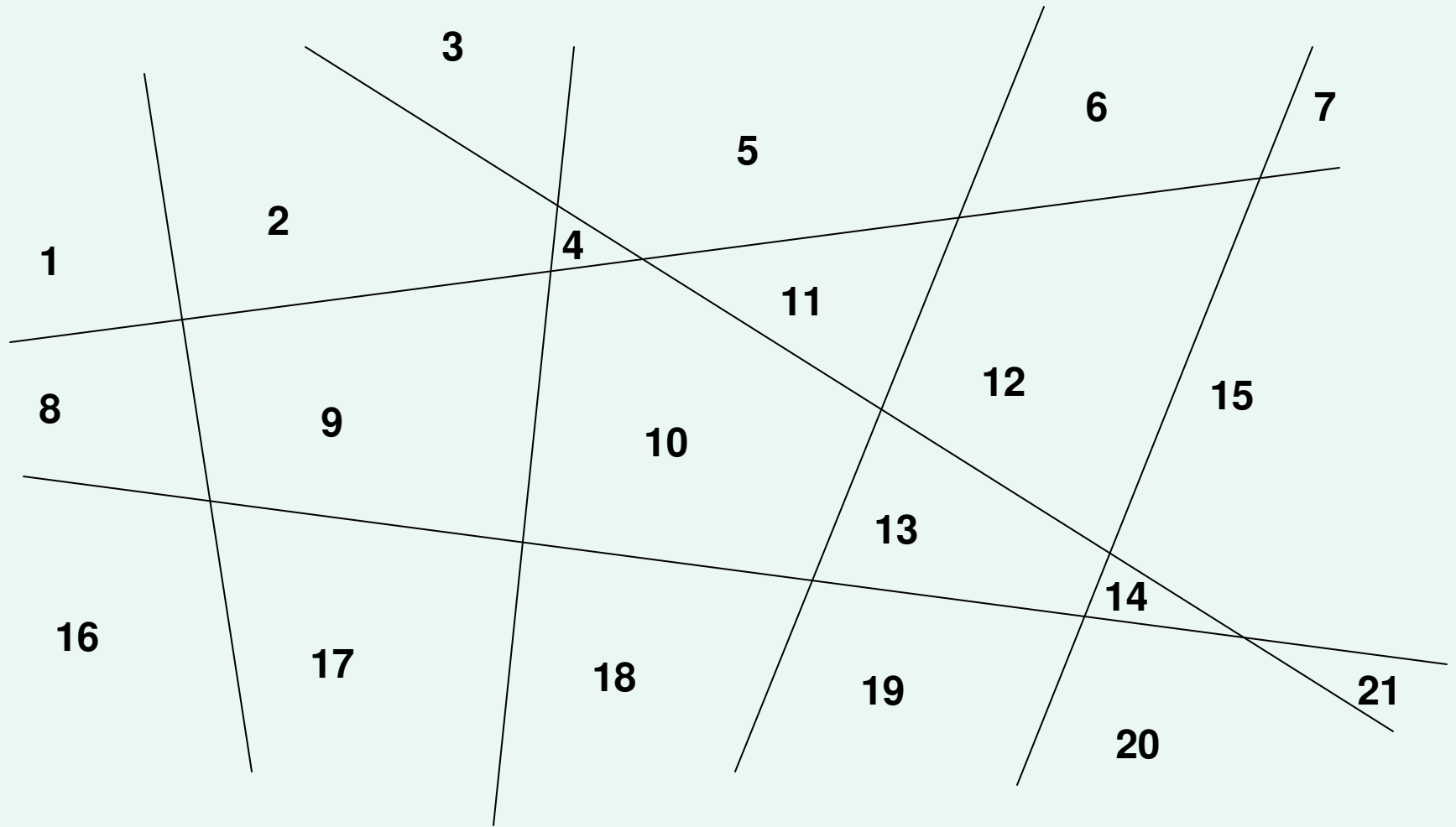
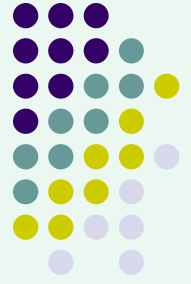


- Improved complexities
 - Range searching
- Simpler Algorithms
 - Convex hulls, triangulation, LP
- Dynamic Algorithms
 - With minimal modifications

Trapezoidal decomposition

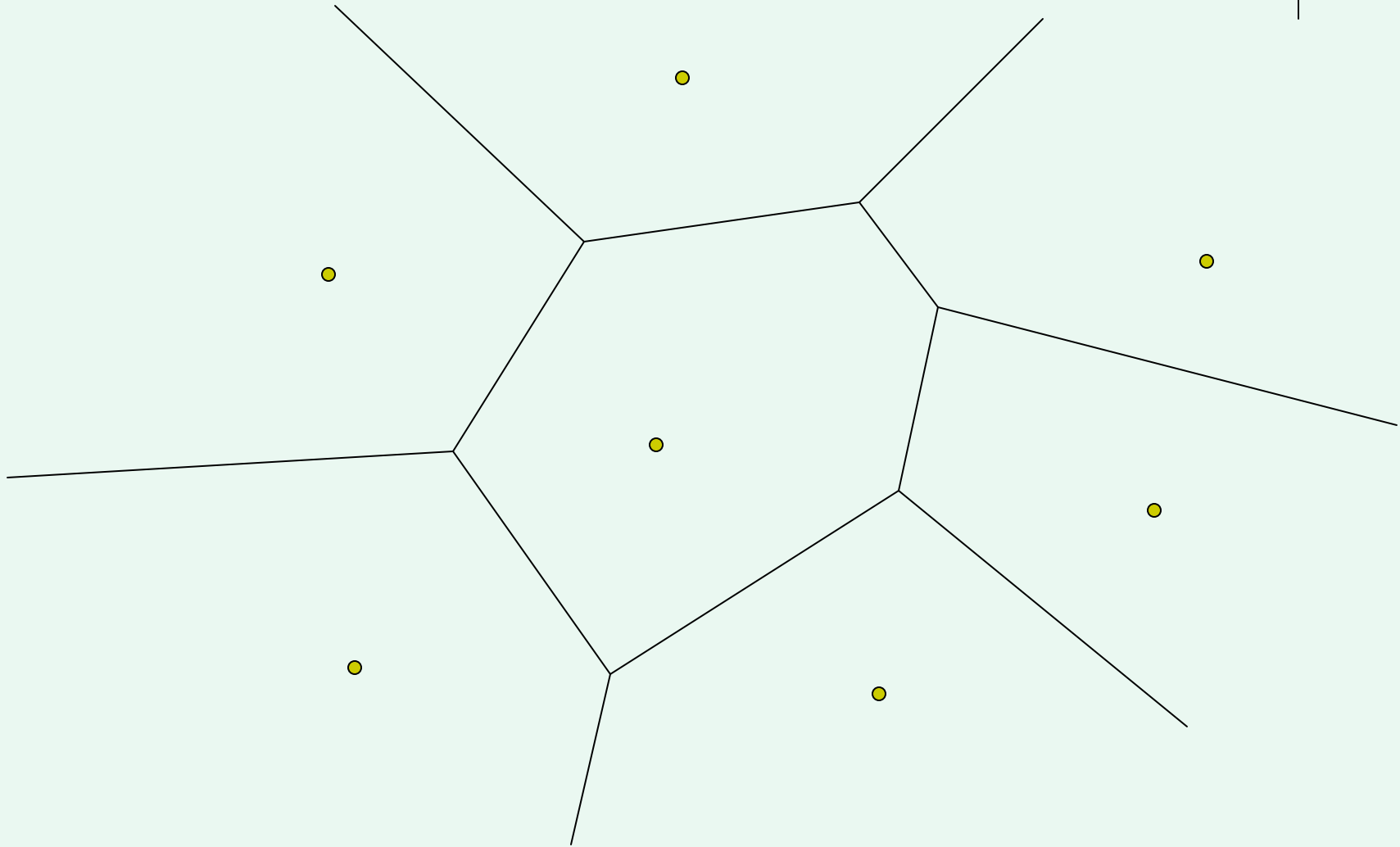
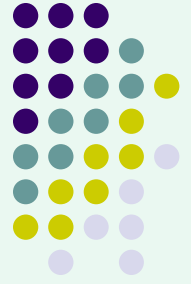


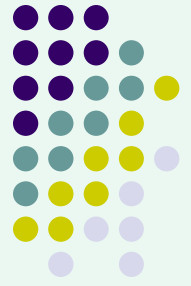
Arrangement Searching



Voronoi Decompositions

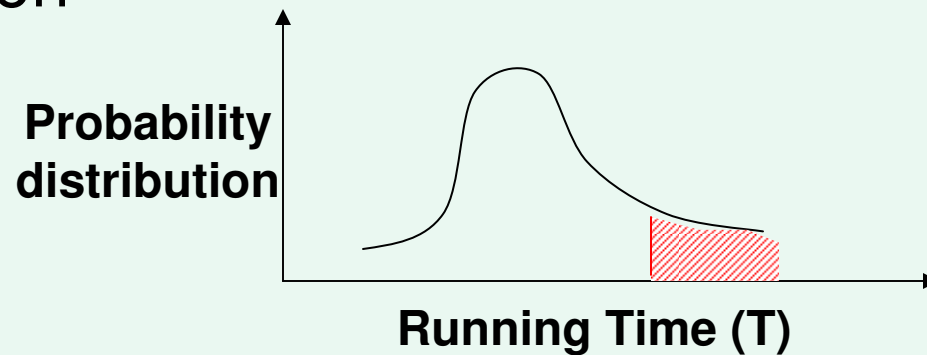
Nearest Neighbor Searching





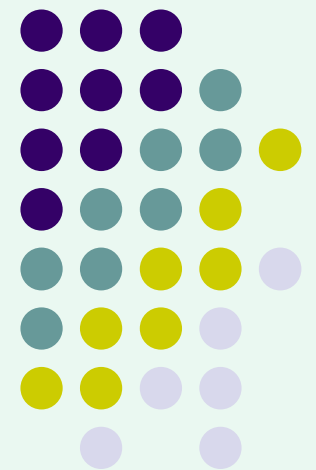
Randomized Algorithms

- RNG in constant time
- Assumes no input distribution
- Halts with correct output
- Running time is bounded by some probability distribution



- Expectation[T]
- Tail Estimates: $Prob[T_n > f(n) \leq \varepsilon]$

Randomized Incremental Construction (R.I.C.)



Randomized Incremental Construction



- **Algorithm**

1. Start with an empty set
2. Insert next object (one at a time)
3. Update the partial construction (data structures)

- Total time = \sum_i (Time to insert the i^{th} object)
- $T_s(N)$ = Total time to insert a sequence s
 - s is good if total time is less
- Expected total time
 - Expected time for a Random Insertion Sequence

Quick-sort as R.I.C.

(Conflict graph)



● ● ● ● ● ● ● ●
X₁ X₂ X₃ X₄ X₅ X₆ X₇ X₈

-inf

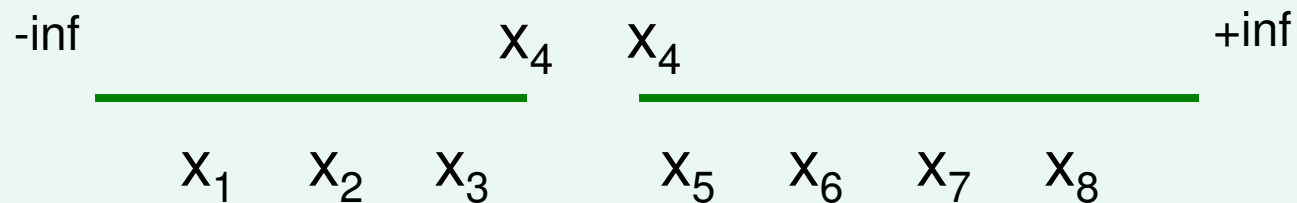
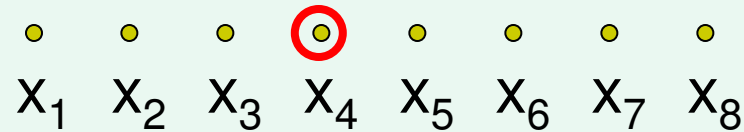
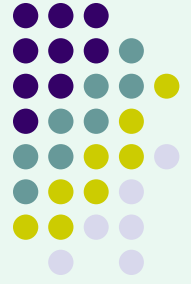
+inf

X₁ X₂ X₃ X₄ X₅ X₆ X₇ X₈

Conflict list (*unordered*)

Quick-sort as R.I.C.

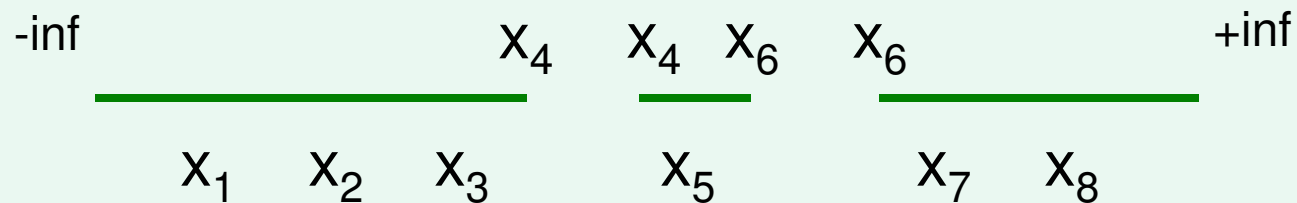
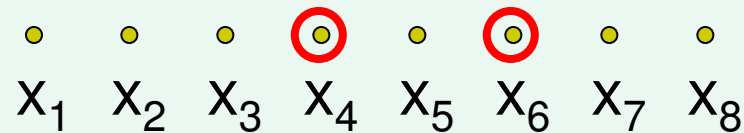
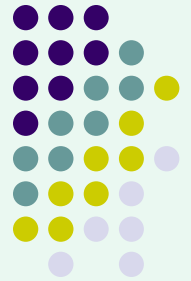
(Conflict graph)



Conflict lists (*unordered*)

Quick-sort as R.I.C.

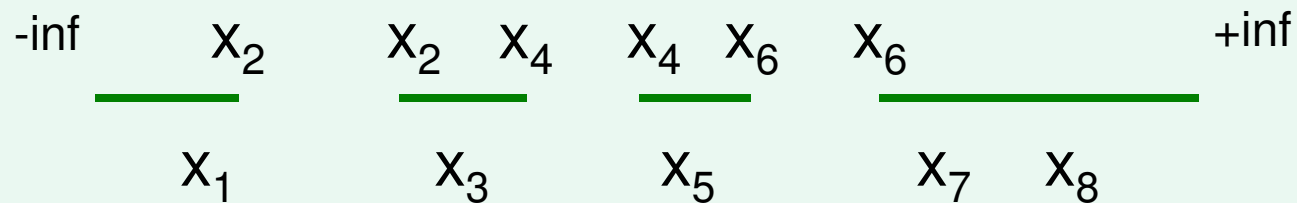
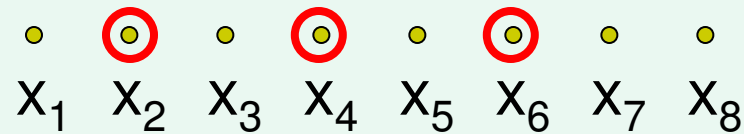
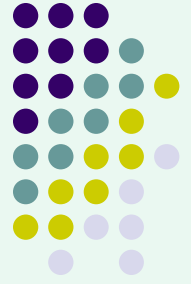
(Conflict graph)



Conflict lists (*unordered*)

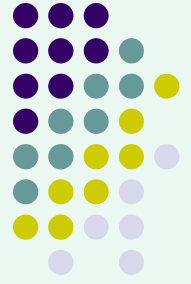
Quick-sort as R.I.C.

(Conflict graph)



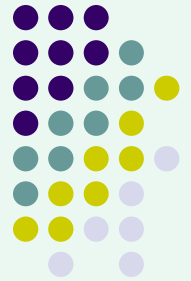
Conflict lists (*unordered*)

Quick-sort as R.I.C. (Analysis)



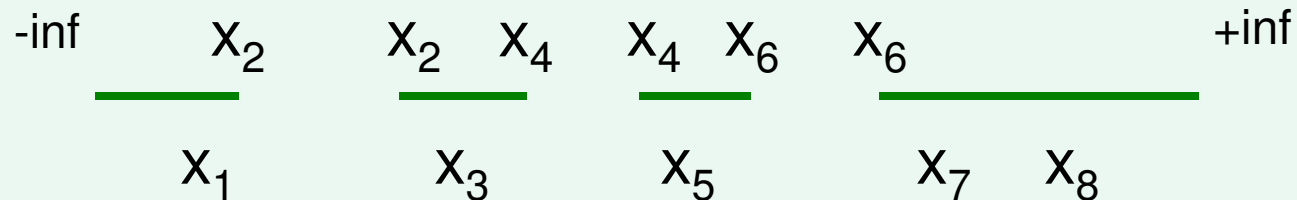
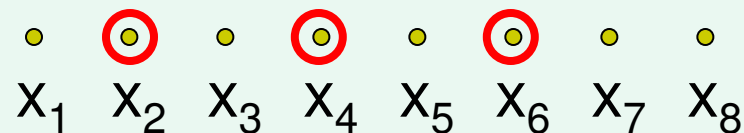
- Notation
 - N_i = Set of selected objects at stage i
 - $H(N_i)$ = complex (partitions/intervals/faces) induced by N_i
 - $I(\sigma)$ = Conflict list for a configuration $\sigma \in H(N_i)$

Quick-sort as R.I.C. (Analysis)



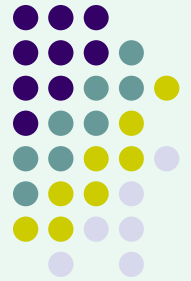
- Notation

- N_i = Set of selected objects at stage i
- $H(N_i)$ = complex (partitions/intervals/faces) induced by N_i
- $I(\sigma)$ = Conflict list for a configuration $\sigma \in H(N_i)$



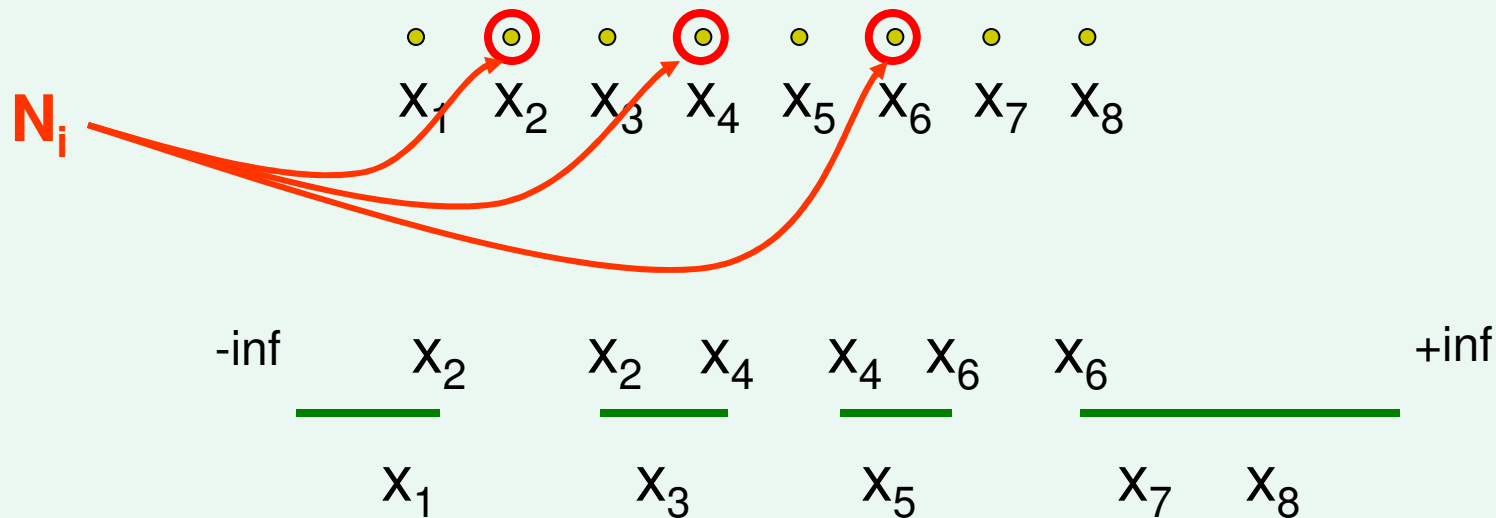
Quick-sort as R.I.C.

(Analysis)



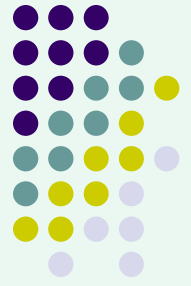
- Notation

- N_i = Set of selected objects at stage i
- $H(N_i)$ = complex (partitions/intervals/faces) induced by N_i
- $I(\sigma)$ = Conflict list for a configuration $\sigma \in H(N_i)$



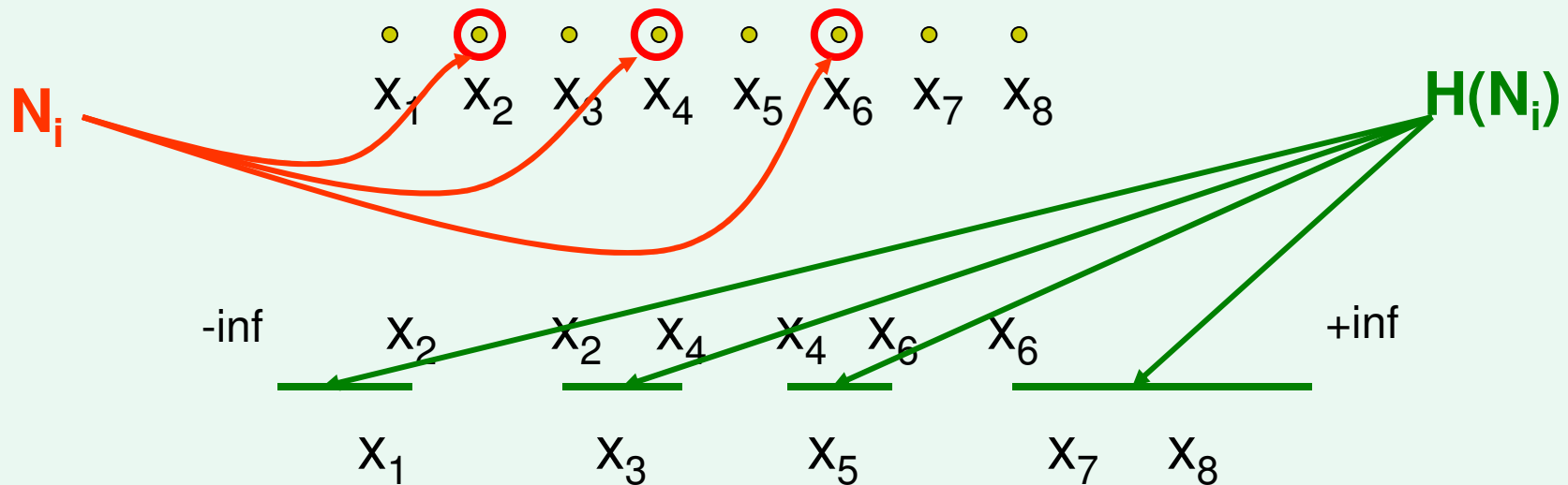
Quick-sort as R.I.C.

(Analysis)



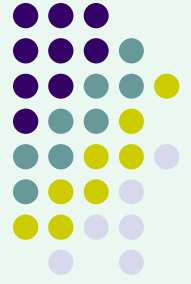
- Notation

- N_i = Set of selected objects at stage i
- $H(N_i)$ = complex (partitions/intervals/faces) induced by N_i
- $I(\sigma)$ = Conflict list for a configuration $\sigma \in H(N_i)$



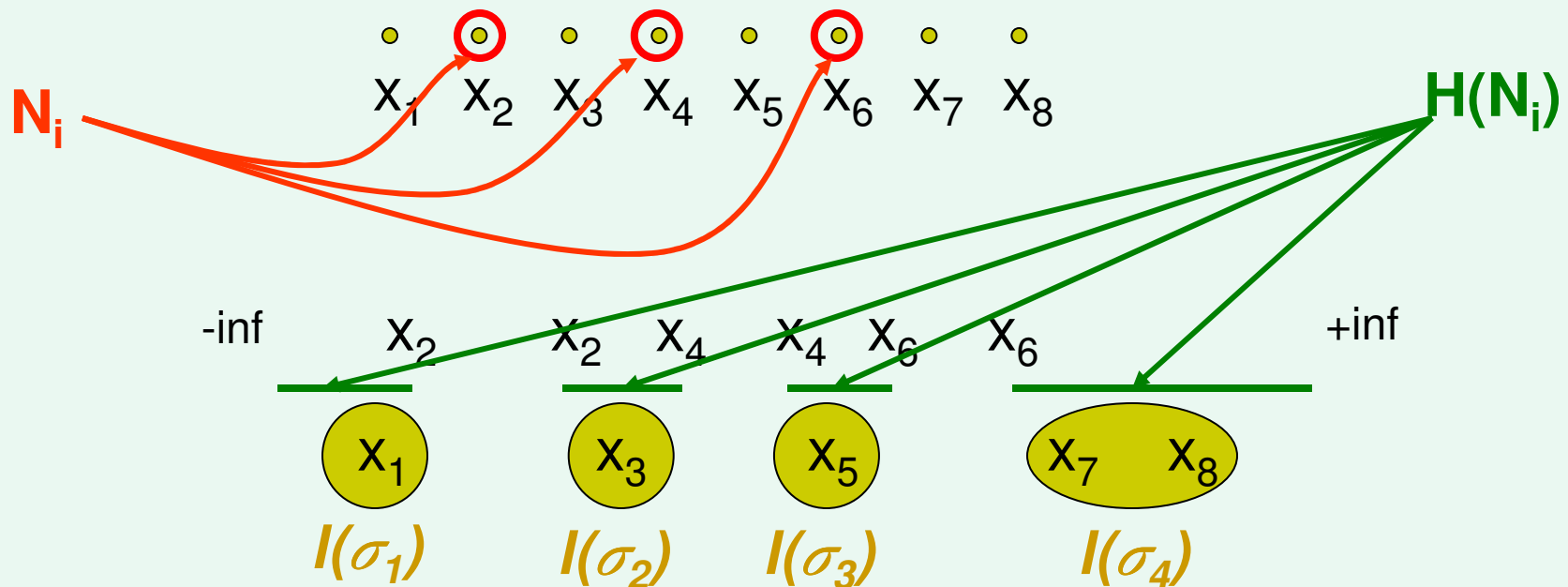
Quick-sort as R.I.C.

(Analysis)

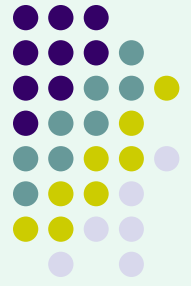


- Notation

- N_i = Set of selected objects at stage i
- $H(N_i)$ = complex (partitions/intervals/faces) induced by N_i
- $I(\sigma)$ = Conflict list for a configuration $\sigma \in H(N_i)$



Quick-sort as R.I.C. (Analysis)

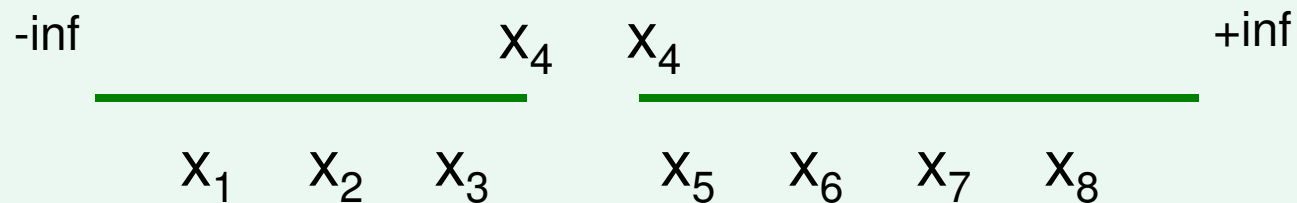
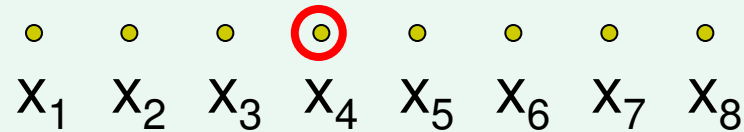
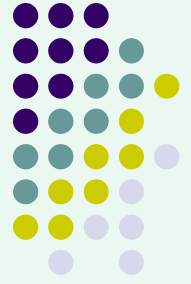


- Cost of R.I.C.
 - c_i = expected cost of adding i^{th} object
 - Total expected cost = $\sum_i c_i$
- Consider a fixed N_{i+1}
 - Every object is equally likely to be the $(i+1)^{\text{th}}$ insertion

$$c_i = \frac{1}{i+1} \sum_{S \in N_{i+1}} [l(I_1(S)) + l(I_2(S)) + 1]$$

Quick-sort as R.I.C.

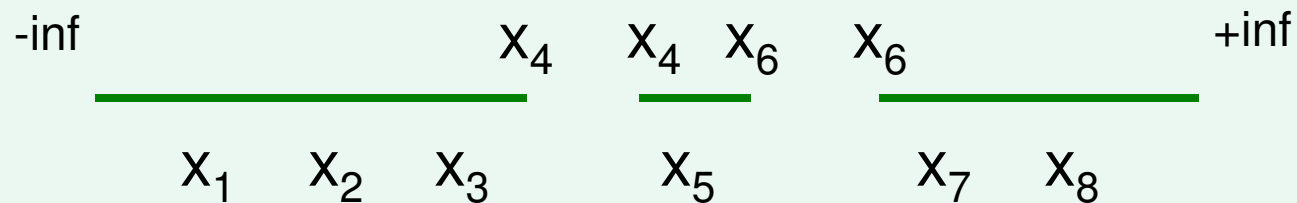
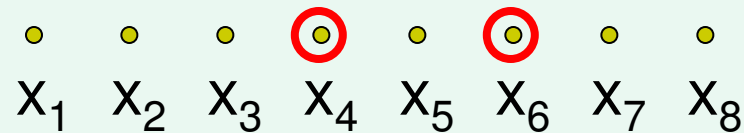
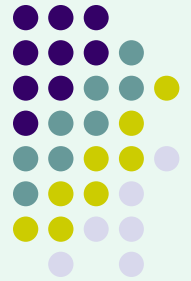
(Conflict graph)



Conflict lists (*unordered*)

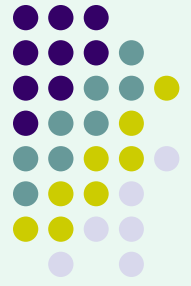
Quick-sort as R.I.C.

(Conflict graph)



Conflict lists (*unordered*)

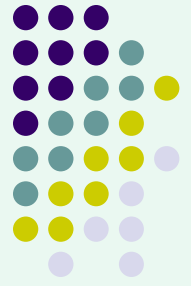
Quick-sort as R.I.C. (Analysis)



- Cost of R.I.C.
 - c_i = expected cost of adding i^{th} object
 - Total expected cost = $\sum_i c_i$
- Consider a fixed N_{i+1}
 - Every object is equally likely to be the $(i+1)^{\text{th}}$ insertion

$$c_i = \frac{1}{i+1} \sum_{S \in N_{i+1}} [l(I_1(S)) + l(I_2(S)) + 1]$$

Quick-sort as R.I.C. (Analysis)



- Cost of R.I.C.
 - c_i = expected cost of adding i^{th} object
 - Total expected cost = $\sum_i c_i$
- Consider a fixed N_{i+1}
 - Every object is equally likely to be the $(i+1)^{\text{th}}$ insertion

$$\begin{aligned} c_i &= \frac{1}{i+1} \sum_{S \in N_{i+1}} [l(I_1(S)) + l(I_2(S)) + 1] \\ &\leq \frac{2}{i+1} \sum_{J \in H(N_{i+1})} [l(J) + 1] = O\left(\frac{n}{i+1}\right) \end{aligned}$$

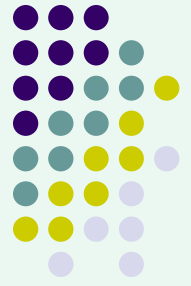
Quick-sort as R.I.C. (Analysis)



- Cost of R.I.C.
 - c_i = expected cost of adding i^{th} object
 - Total expected cost = $\sum_i c_i$
- Consider a fixed N_{i+1}
 - Every object is equally likely to be the $(i+1)^{\text{th}}$ insertion

$$\begin{aligned} c_i &= \frac{1}{i+1} \sum_{S \in N_{i+1}} [l(I_1(S)) + l(I_2(S)) + 1] \\ &\leq \frac{2}{i+1} \sum_{J \in H(N_{i+1})} [l(J) + 1] = O\left(\frac{n}{i+1}\right) \end{aligned}$$

$$\sum_i c_i = O(n \log n)$$



Backward Analysis

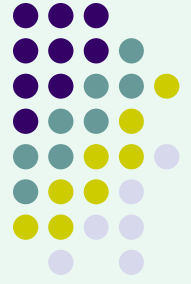
- For a fixed set of $i+1$ objects, what is the probability that the $(i+1)^{\text{st}}$ insertion affects σ ?
- Because of random insertion sequence, any one of the fixed $(i+1)$ objects is equally likely to be the last inserted object (by symmetry)

What is the probability that a random deletion from $(i+1)$ objects defines σ ? (pretending to run backwards)

$$\frac{d(\sigma)}{i+1}$$

- Since conditional expected cost depends only on i (independent of the actual set of objects)

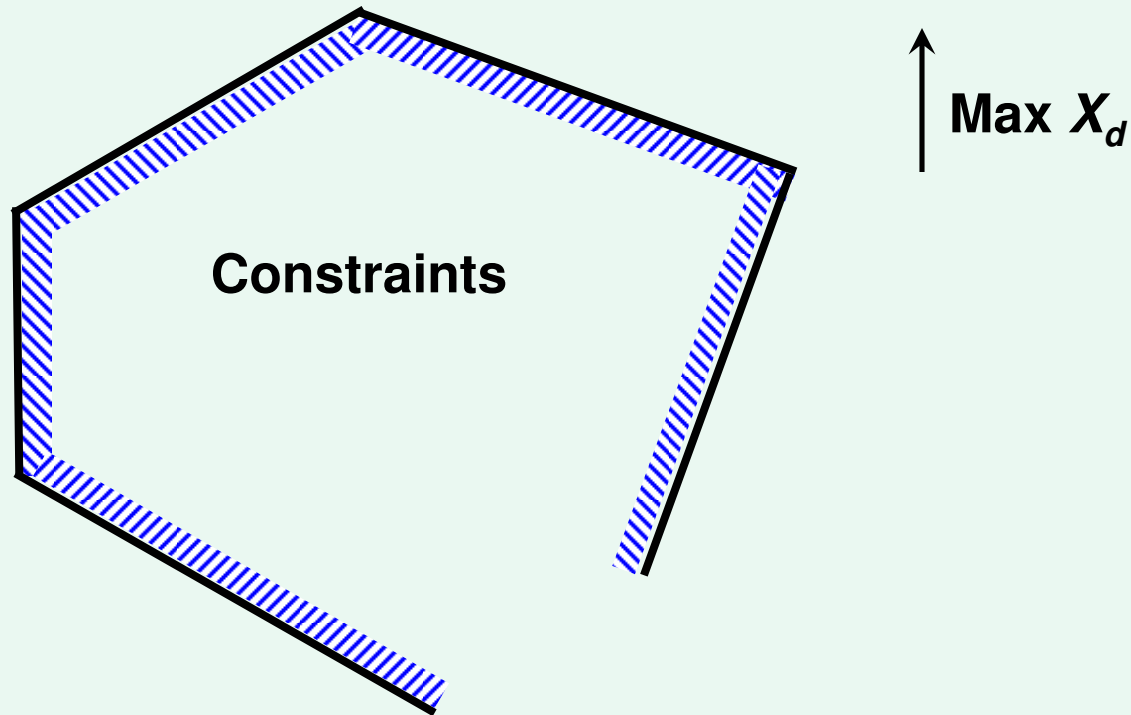
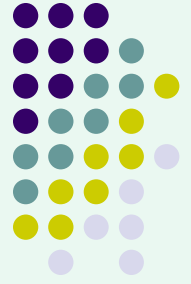
Conditional Expected cost = (Unconditional) Expected cost



R.I.C. – Some results

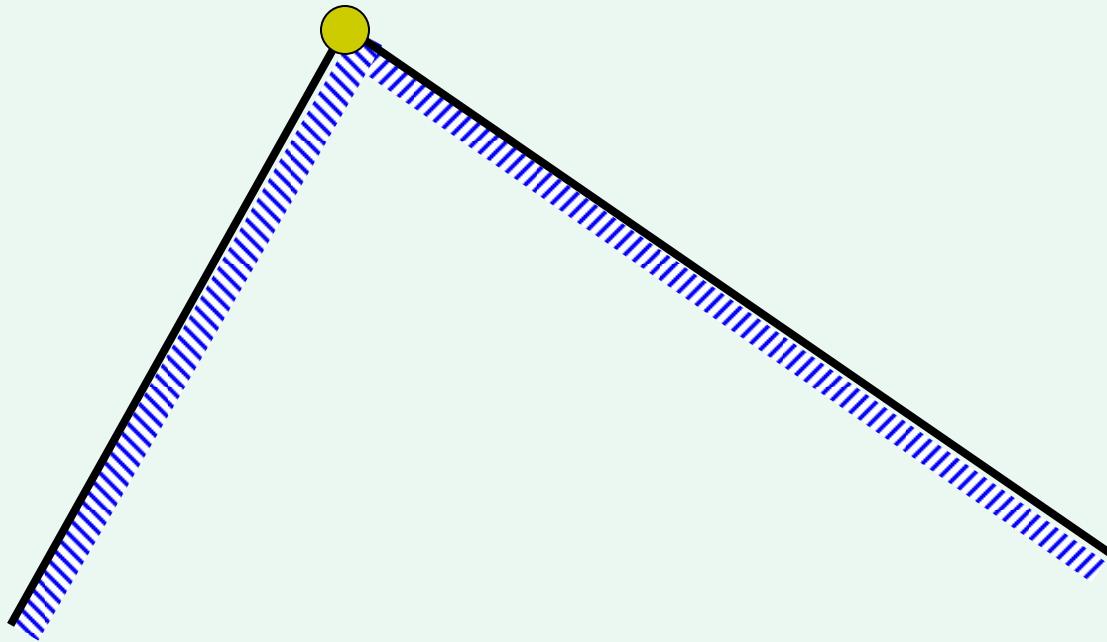
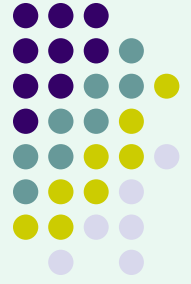
- Trapezoidal Decompositions
 - $E[T] = O(n + k \log n)$
- *Voronoi Diagrams*
 - $E[T] = O(n \log n)$

Linear Programming (Fixed dimensions)

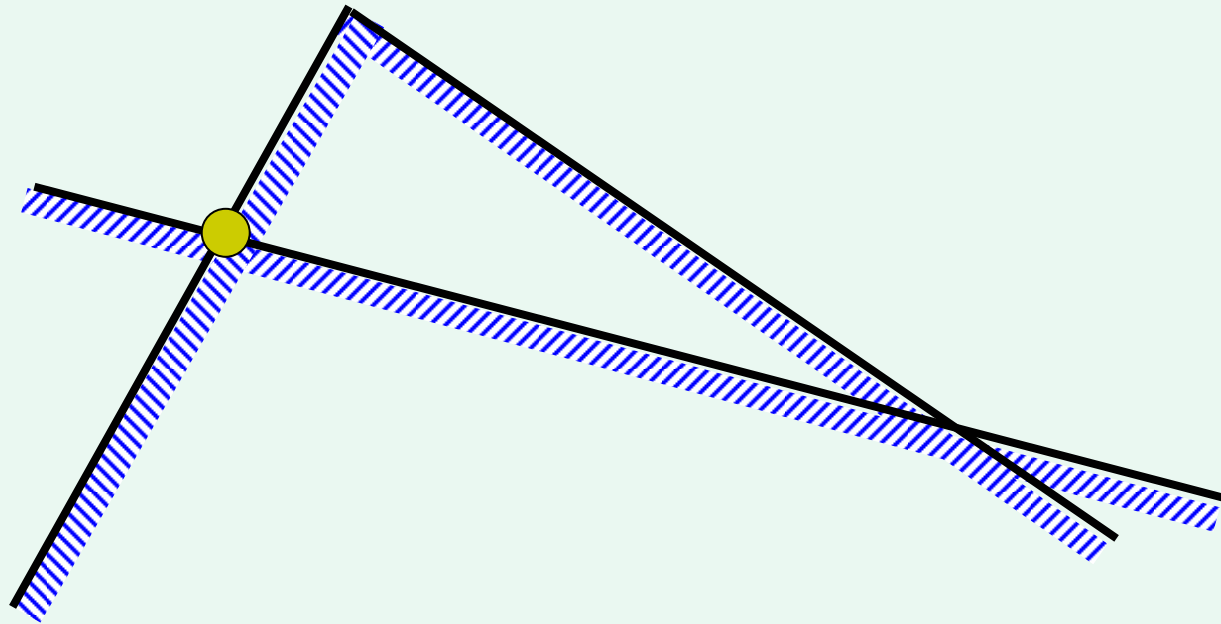


- Non-degenerate : Exactly d constraints define the optimum

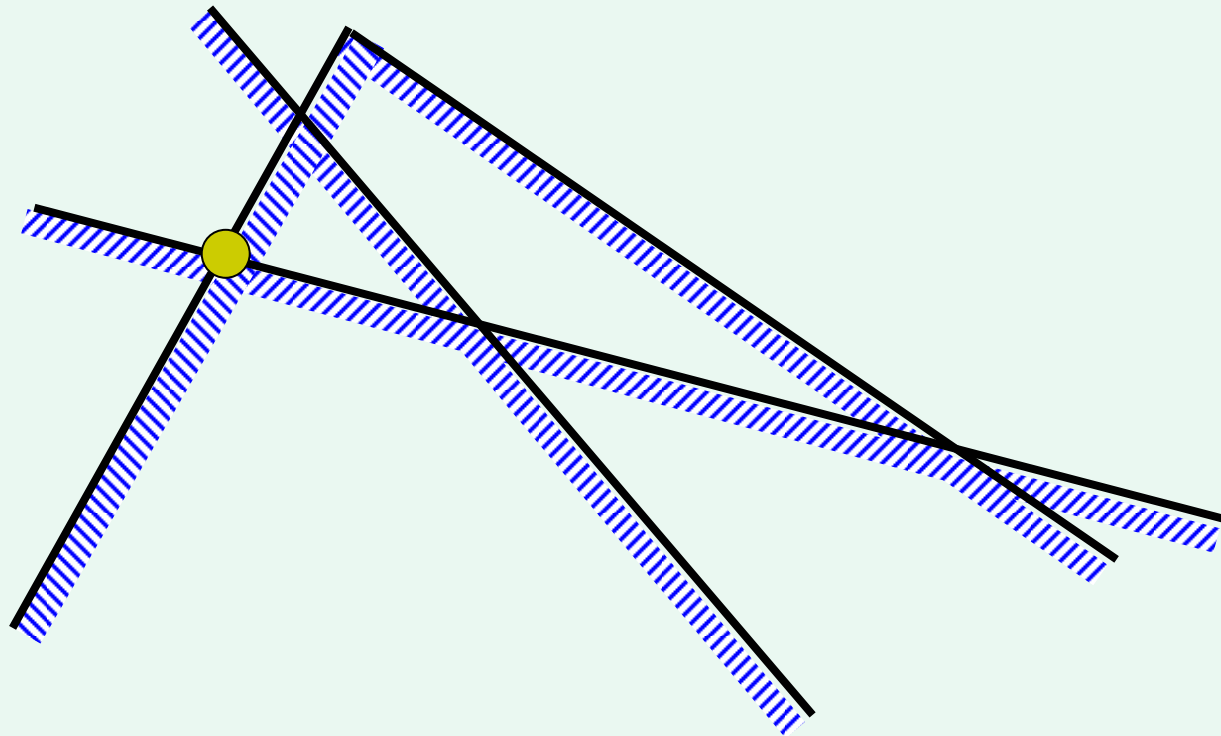
Linear Programming



Linear Programming

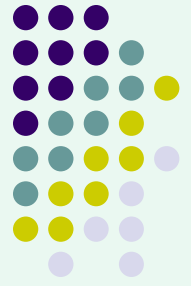


Linear Programming



Linear Programming

Expected Running Time Analysis



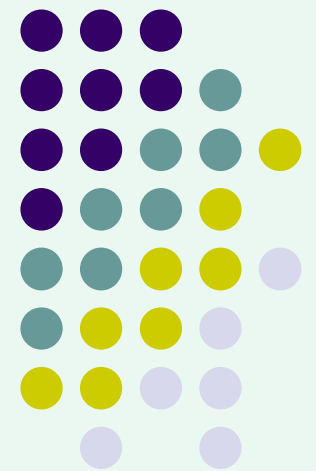
- $\overline{T_d(n)}$ = Expected running time in d dimensions for n constraints
- From backward analysis, probability that i^{th} insertion changes optimum is d/i (d constraints define optimum)

$$\overline{T_d(i)} = \overline{T_d(i-1)} + \overline{T_{d-1}(i-1)} \cdot \frac{d}{i} + O(d)$$

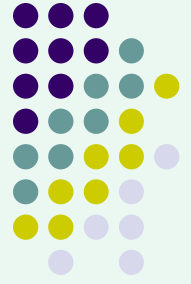
- By induction [Seidel]

$$\overline{T_d(n)} = O(d! n)$$

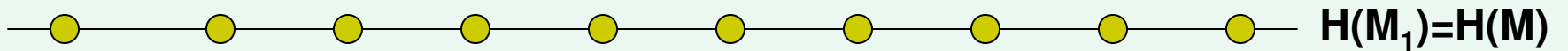
Random Sampling

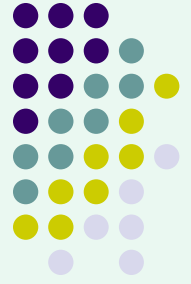


Skip Lists



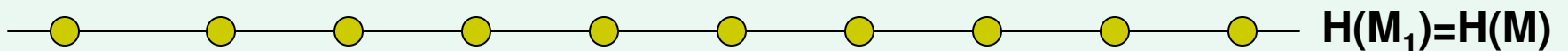
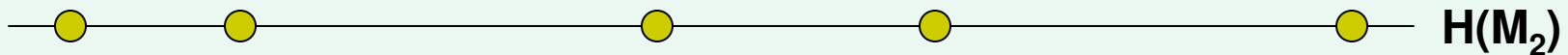
- Start with complete set of points = M

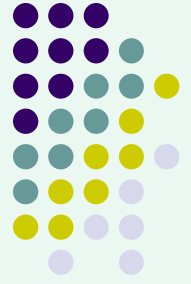




Skip Lists

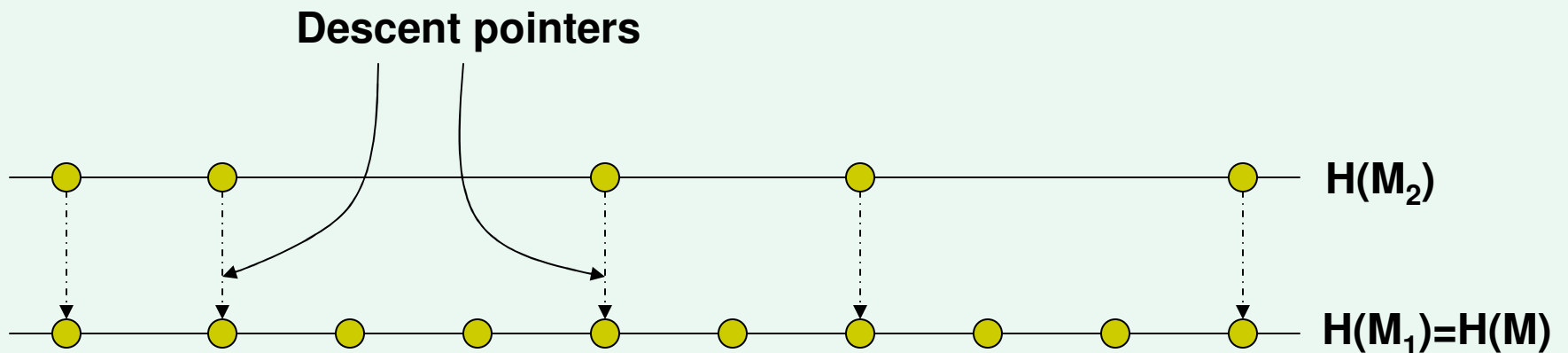
- Start with complete set of points = M
- Promote each point with prob. $\frac{1}{2}$ to the next level

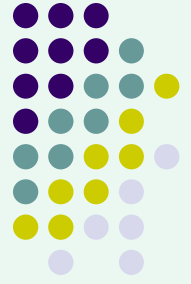




Skip Lists

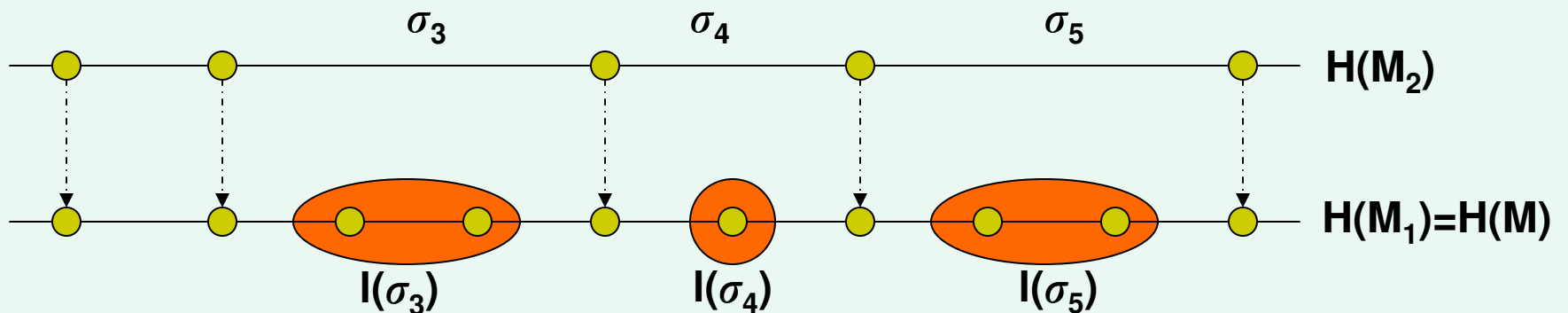
- Start with complete set of points = M
- Promote each point with prob. $\frac{1}{2}$ to the next level
- Create descent pointers

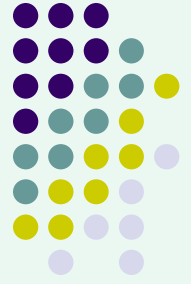




Skip Lists

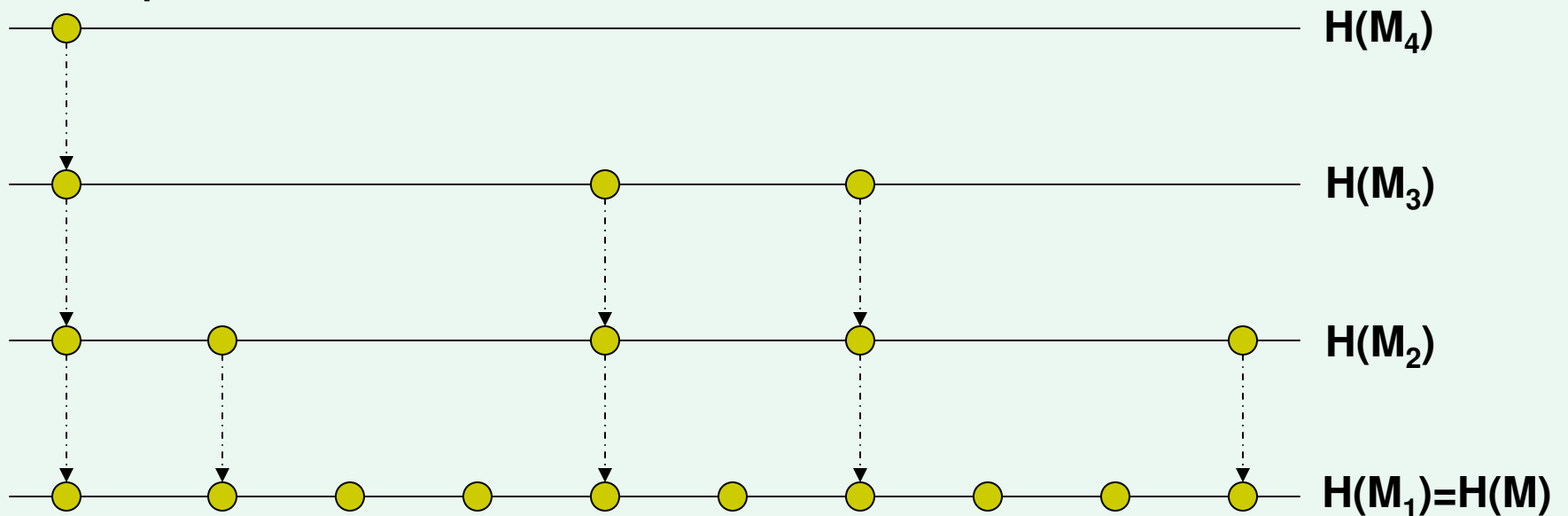
- Start with complete set of points = M
- Promote each point with prob. $\frac{1}{2}$ to the next level
- Create descent pointers

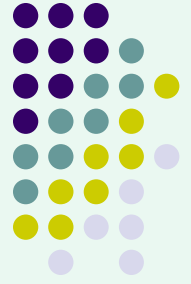




Skip Lists

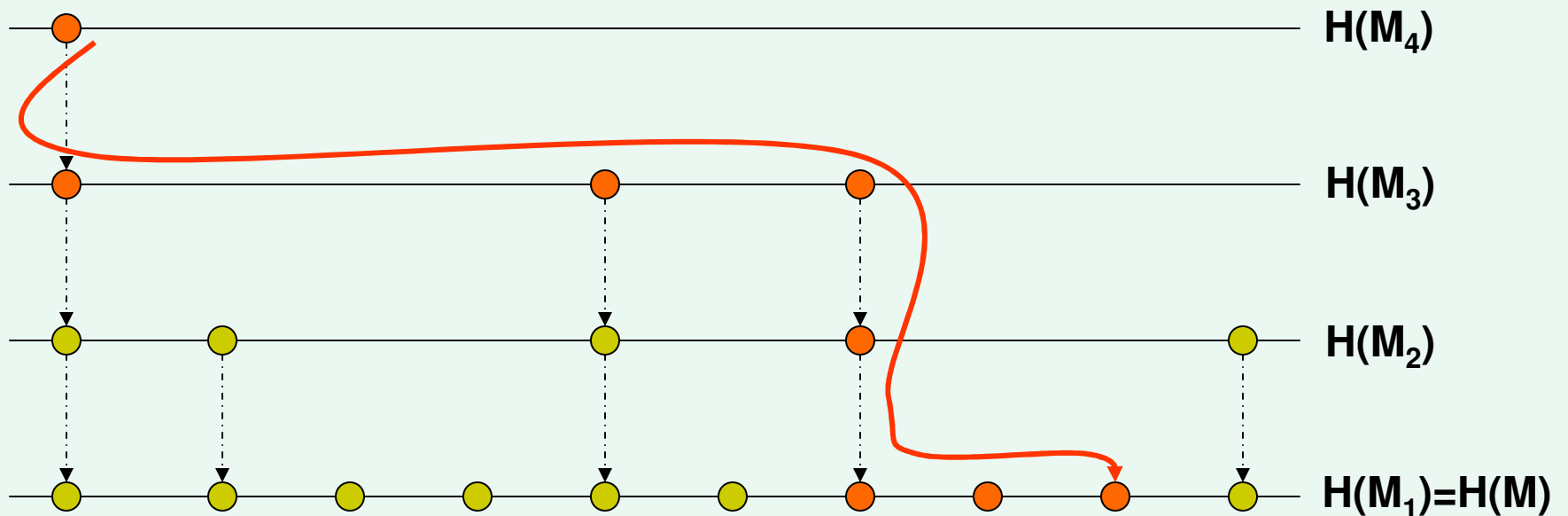
- Start with complete set of points = M
- Promote each point with prob. $\frac{1}{2}$ to the next level
- Create descent pointers
- Repeat

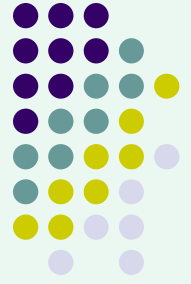




Skip Lists

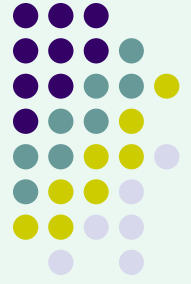
- Search - recursively
 - Follow descent pointers
 - Traverse conflict list to locate interval containing query





Skip Lists

- Analysis
 - # Levels = $O(\log m)$ w.h.p.
 - $P[\text{level} > k] < 1/2^k$ for any element
 - $P[\text{level} > k] < m/2^k$ for every element
 - Space = $O(m)$
 - $\text{Exp}[\text{levels}] = \sum_i (i/2^i) = O(1)$ for each element
 - Construction Time = $O(m \log m)$



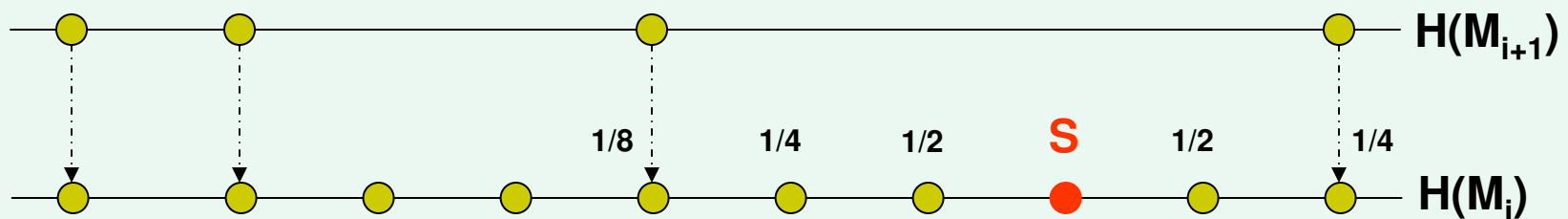
Skip Lists

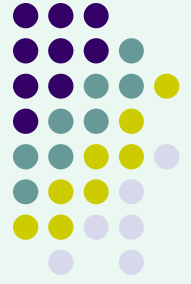
- Analysis

- Search Cost = $O(\log m)$

- $O(\sum_i 1 + l(\Delta_i))$ summing over all intervals in search path

- $Exp[l(\Delta_i)] = 2 \times \sum_t t/2^t = O(1)$





Bottom-up Sampling

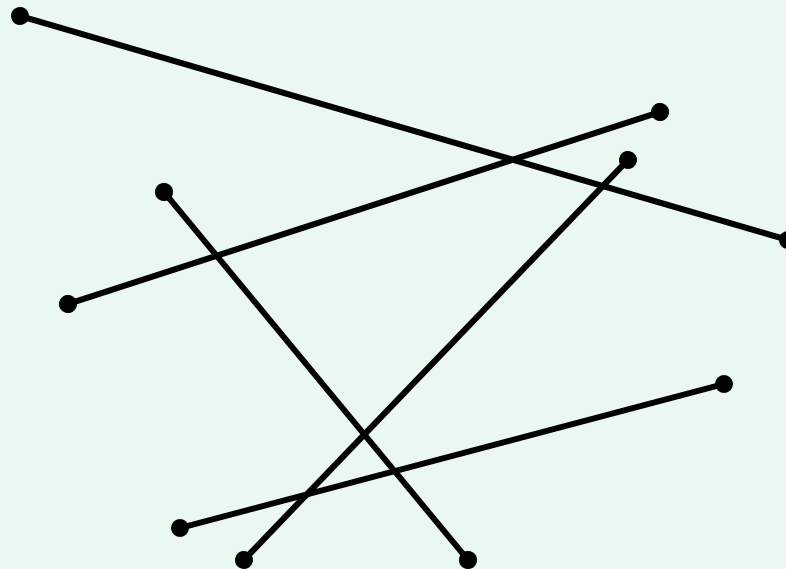
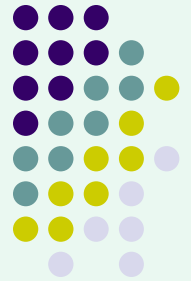
- **Algorithm for constructing the complex**

1. Start with set M
2. Obtain a sequence of sets $M = M_1 \subseteq M_2 \subseteq \dots \subseteq M_r = \phi$
 M_{i+1} is obtained from M_i by selecting each object with probability $1/2$
3. Build the complex for each set M_i
4. Build a descent structure to descend from M_{i+1} to M_i

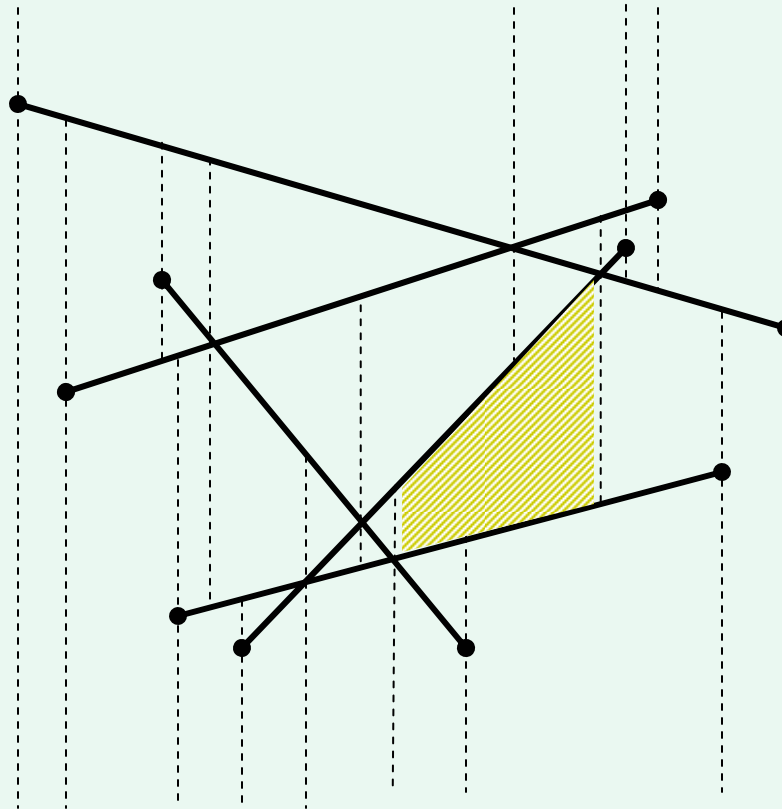
- **Algorithm for searching the complex**

1. Search query point recursively from M_r down to M_1
2. In stage $r-i$ locate point in M_i from M_{i+1} using descent structure
3. In stage $r-1$, the point is located in M

Trapezoidal decomposition

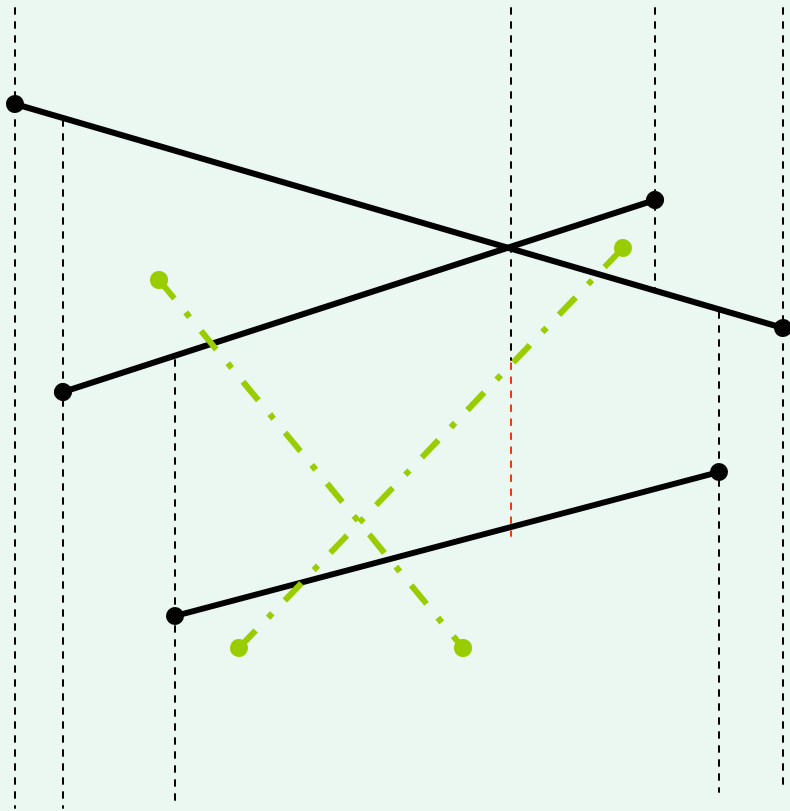
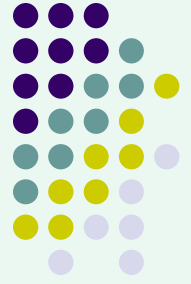


Trapezoidal decomposition

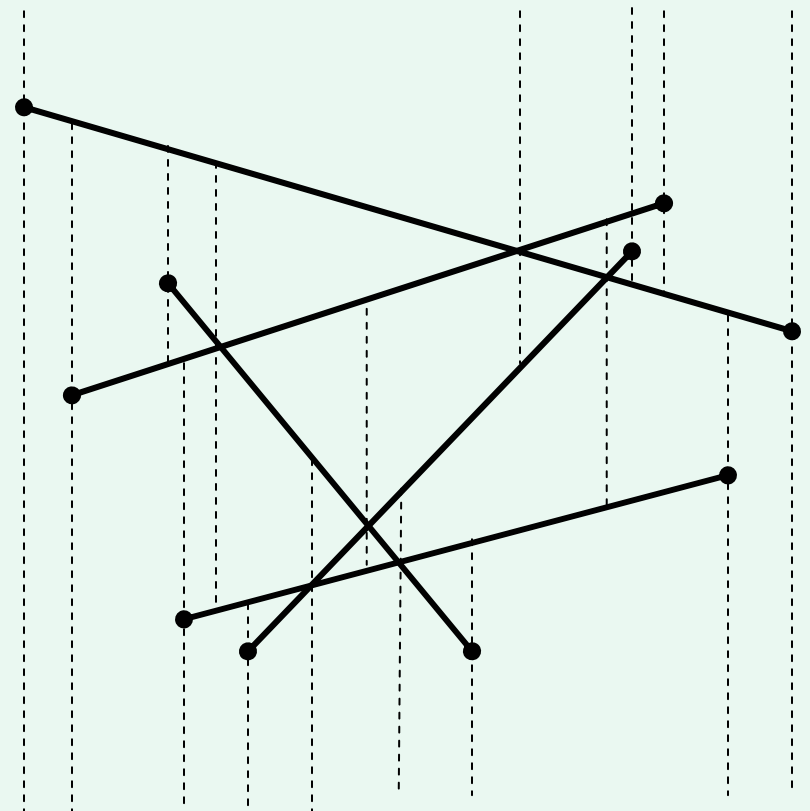


Trapezoidal decomposition

Random Sampling



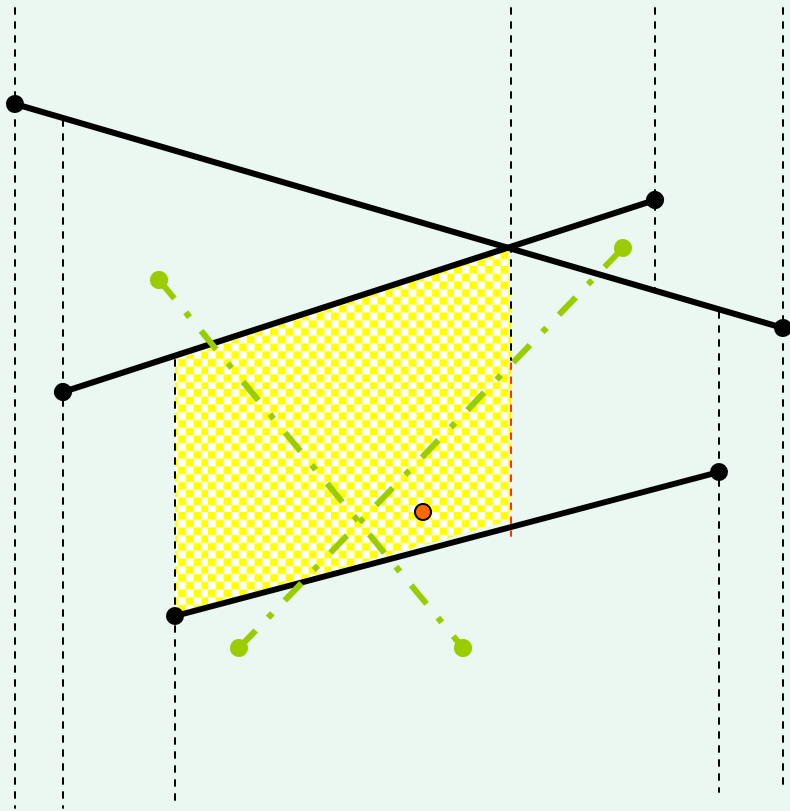
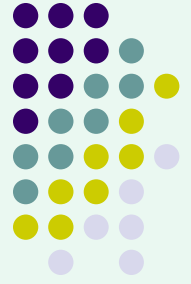
$H(M_{i+1})$



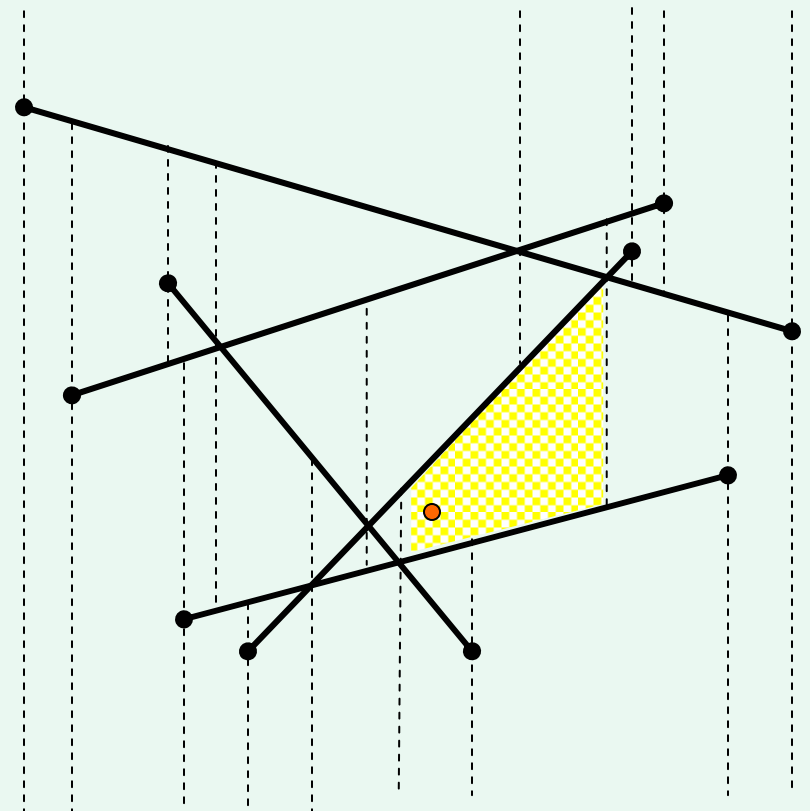
$H(M_i)$

Trapezoidal decomposition

Point Location



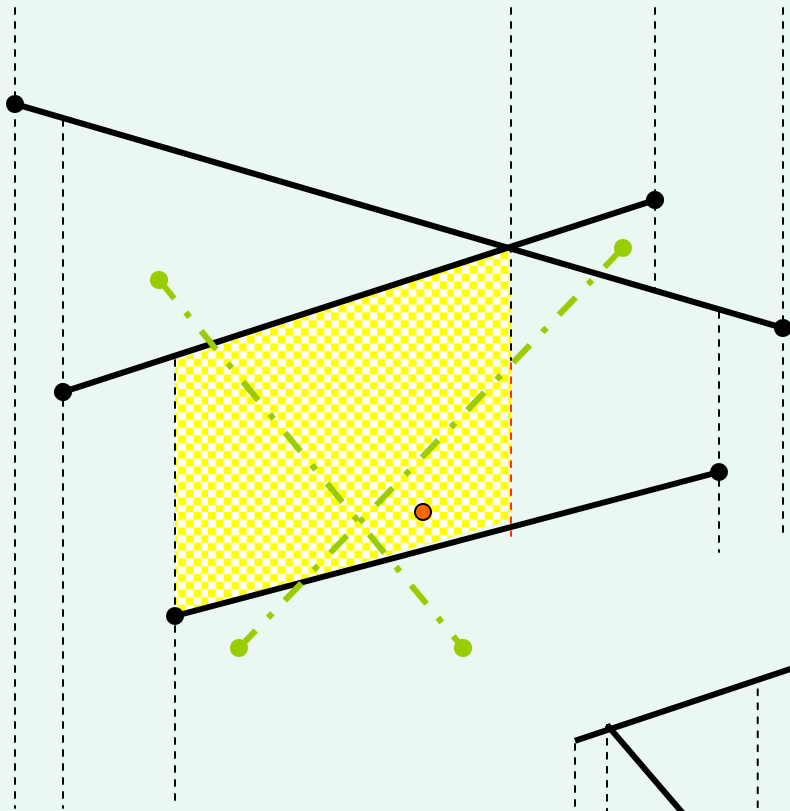
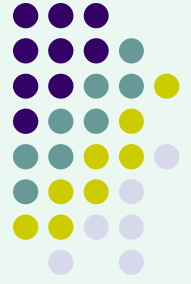
$H(M_{i+1})$



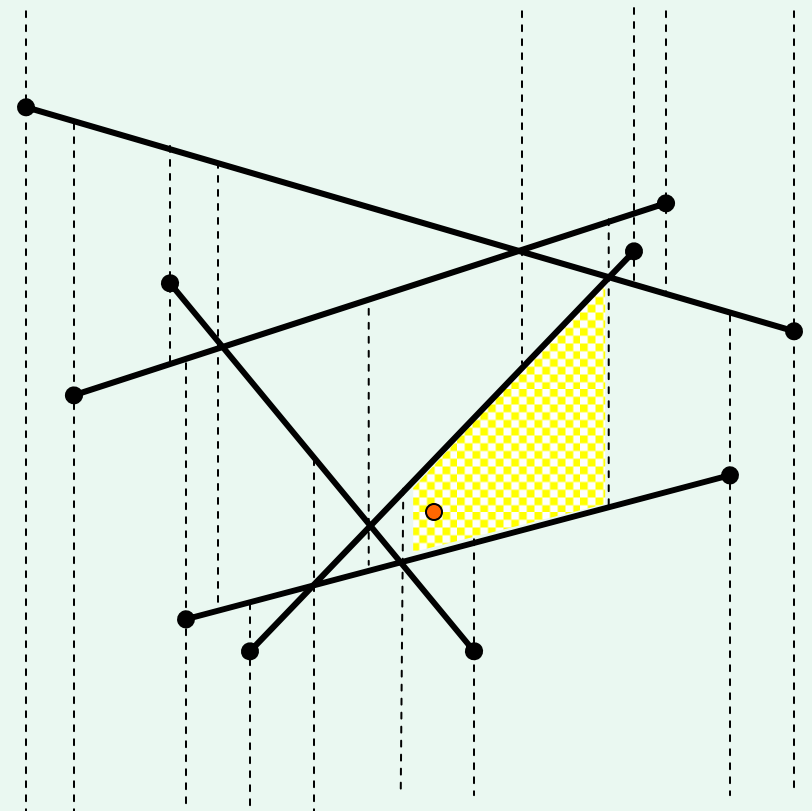
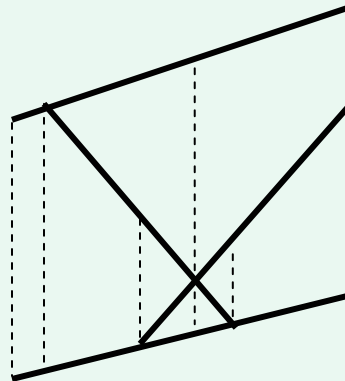
$H(M_i)$

Trapezoidal decomposition

Descent Structure



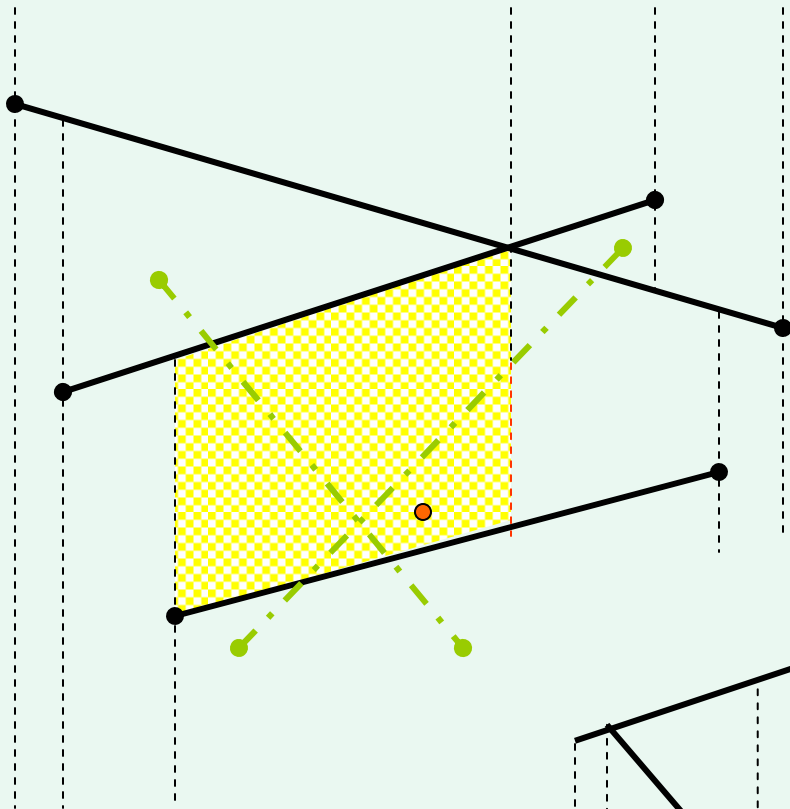
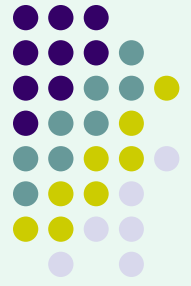
$H(M_{i+1})$



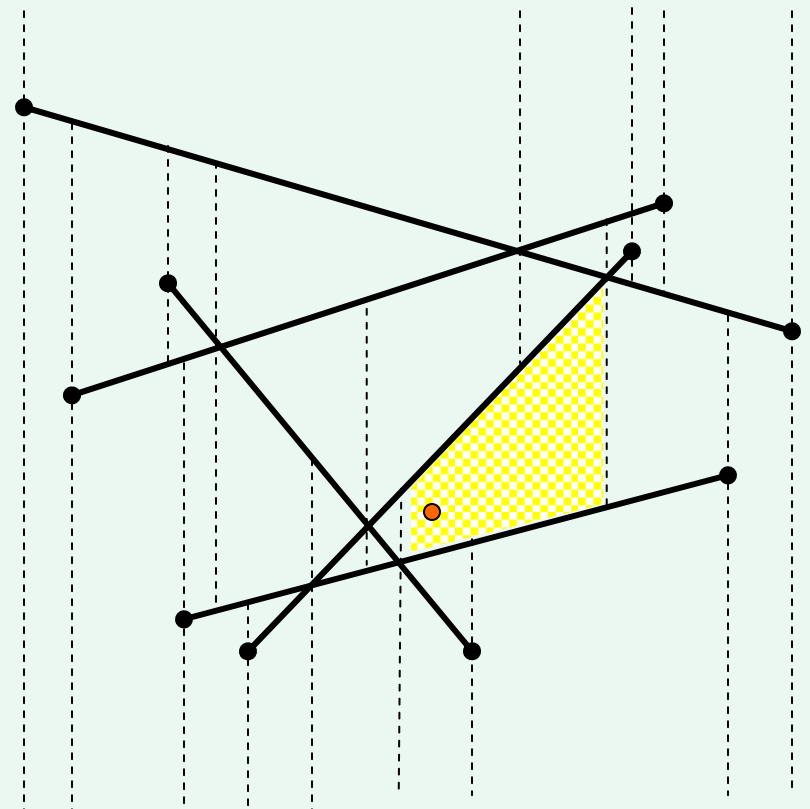
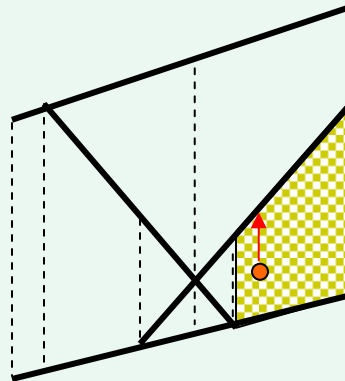
$H(M_i)$

Trapezoidal decomposition

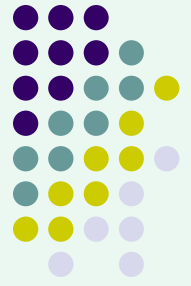
Descent Structure



$H(M_{i+1})$



$H(M_i)$



Bounding the conflict sizes

Claim:

$$\Pr \{ \max_{\sigma \in H(R)} l(\sigma) \geq c (n/r) \log r \} \leq 1/2$$

$|R| = r$ by Bernoulli sampling

$p(\sigma, r)$: conditional prob. that none of the k conflicting elements are selected given σ is feasible

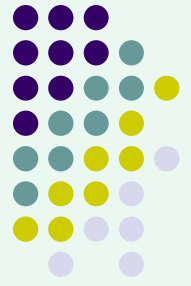
$$\leq (1-r/n)^k$$

$$\leq e^{-c \log r}$$

for $k \geq cn/r \ln r$

BAD σ

$$= 1/r^c$$



Bounding the conflict sizes

$q(\sigma, r)$: prob. that $D(\sigma) \subset R$

Prob. that $\sigma \in H(R) = p(\sigma, r) \times q(\sigma, r)$

Prob. that some $\sigma \in H(R)$ is *BAD* ($l(\sigma) \geq c(n/r) \log r$)

$$\leq 1/r^c \sum_{\sigma} q(\sigma, r)$$

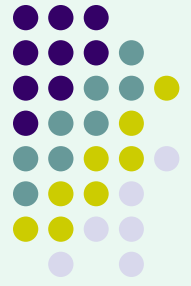
$$= 1/r^c E(H(R))$$

usually $H(R) = r^{O(1)}$

$$\leq 1/2 \quad \text{for appropriate } c$$

Trapezoidal Decomposition

Point Location



Claim:

$$\Pr \{ \max_{\sigma \in H(R)} l(\sigma) \geq c (n/r) \log r \} \leq 1/2$$

$|R| = r$ by Bernoulli sampling

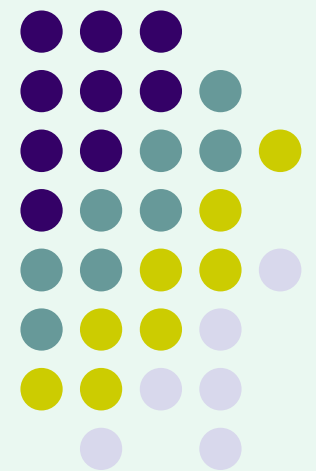
$r = n/2$ (bottom-up sampling) \rightarrow

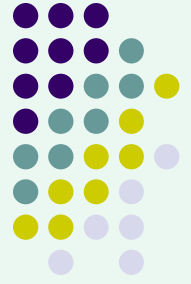
$$\Pr \{ \max_{\sigma \in H(R)} l(\sigma) \geq c \log n \} \leq 1/2$$

Point location in Trapezoidal decomposition:

$O(\log^2 n)$ w.h.p.

Randomized Framework for Clustering problems





Clustering Problems

- Select $X = k$ points in \mathbb{R}^d , that minimize

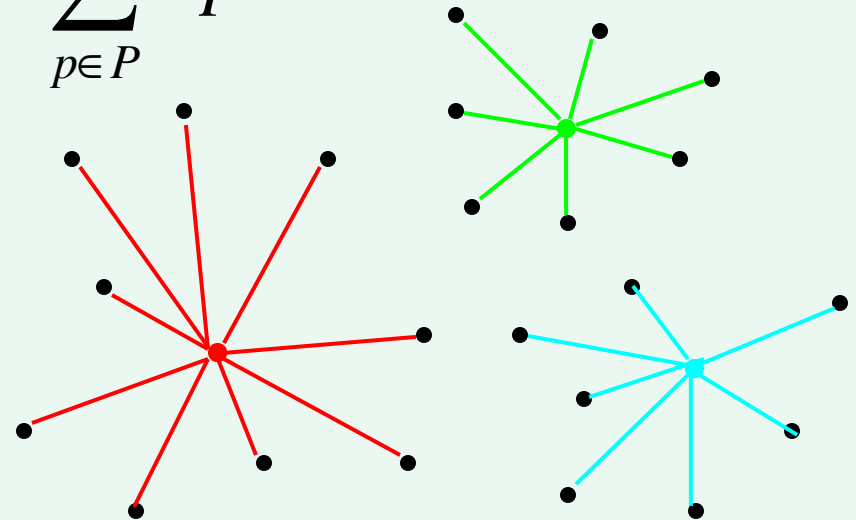
- K-Medians

$$\sum_{p \in P} \| p - X \|$$

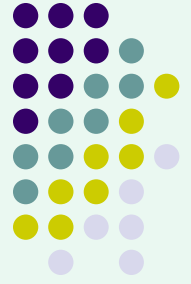
- K-Means

$$\sum_{p \in P} \| p - X \|^2$$

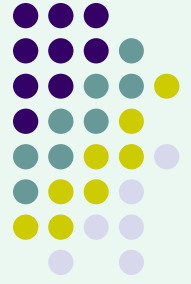
- Each point is *assigned* to closest center in X
- *Cluster* : Set of points assigned to a center



Class of Clustering Problems

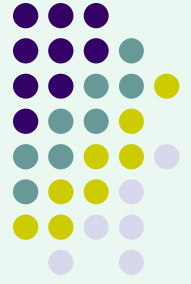


- Conditions
 - ➔ *Existence of Random Sampling Procedure*
 - ➔ *Tightness Property*
- Any clustering problem satisfying these 2 conditions will have a linear time algorithm
- k-means, k-median and discrete k-means clustering problems satisfy these properties



Results : k-means

Matousek	$O\left(n\varepsilon^{-2k^2d} \log^k n\right)$
de la Vega, Karpinski, Kenyon, Rabani	$O\left(2^{(k/\varepsilon)^{O(1)}} d^{O(1)} n \log^{O(k)} n\right)$
Har-Peled, Mazumdar (Discrete also)	$O\left(n+k^{k+2} \varepsilon^{-(2d+1)k} \log^{k+1} n \log^k(1/\varepsilon)\right)$
Sabharwal, Sen, CGTA (2-means)	$O\left((1/\varepsilon)^d n\right)$
Kumar, Sabharwal, Sen, FOCS 2004 (Discrete also)	$O\left(2^{(k/\varepsilon)^{O(1)}} nd\right)$

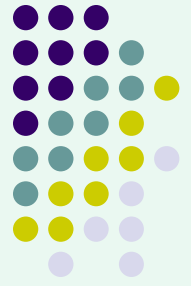


Results : k-medians

Arora, Raghavan, Rao	$O(n^{O(1/\varepsilon)})$ for fixed d
Kolliopoulos, Rao (Discrete only)	$O(\partial n \log n \log k)$
Badoiu, Har-Peled, Indyk	$O\left(2^{(k/\varepsilon)^{O(1)}} d^{O(1)} n \log^{O(k)} n\right)$
Har-Peled, Mazumdar (Discrete also)	$O\left(n + \partial k^{O(1)} \log^{O(1)} n\right)$
Kumar, Sabharwal, Sen, ICALP 2005 (non-discrete only)	$O\left(2^{(k/\varepsilon)^{O(1)}} nd\right)$

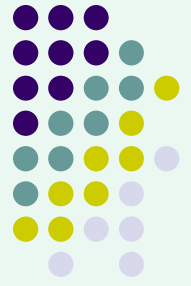
$$\partial = \exp\left[O\left(\frac{(1 + \log 1/\varepsilon)}{\varepsilon}\right)^{d-1}\right]$$

Class of Clustering Problems



- Random Sampling Procedure
 - Generates candidates, C , that approximate the 1-center of a set of points P with constant probability
 - Size of sample must be independent of $|P|$ & d .
 - Size of C must be independent of $|P|$ & d .

k-Means: Random Sampling Procedure

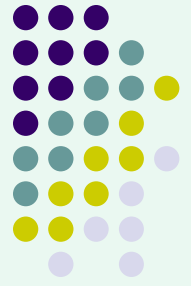


- Optimal 1-Mean is

$$\bar{x}(S) = \frac{1}{|S|} \sum_{p \in S} p$$

- Approximation of Centroid (Inaba et al)
 - T = random sample of size $m=O(1/\epsilon)$
 - Centroid of T is a $(1+\epsilon)$ -approx centroid of S with constant probability

k-Means: Random Sampling Procedure



$$E(\bar{x}(T)) = E(\bar{x}(S)) \quad E(\|\bar{x}(T) - \bar{x}(S)\|^2) = \frac{1}{|T|} \text{Var}(S)$$

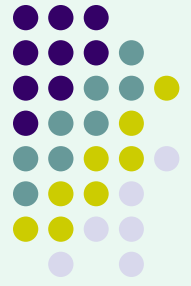
$$\text{Var}(S) = \frac{1}{|S|} \sum_{x_i \in S} \|x_i - \bar{x}(S)\|^2$$

$$\Pr\left(\|\bar{x}(T) - \bar{x}(S)\|^2 > \frac{2}{|T|} \text{Var}(S)\right) < 1/2$$

$$\text{Also, } \sum_{x_i \in S} \|x_i - \bar{x}(T)\|^2 = |S| \text{Var}(S) + |S| \cdot \|\bar{x}(T) - \bar{x}(S)\|^2$$

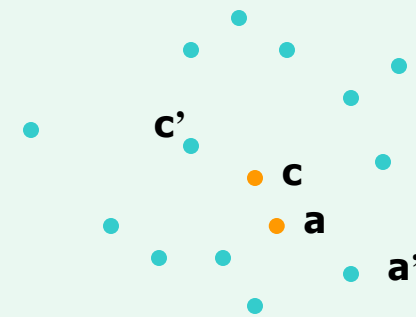
$$\therefore \sum_{x_i \in S} \|x_i - \bar{x}(T)\|^2 < \left(1 + \frac{2}{|T|}\right) |S| \text{Var}(S) = (1 + \varepsilon) |S| \text{Var}(S) \quad \left(T = \frac{2}{\varepsilon}\right)$$

Discrete K-Means: Random Sampling Procedure

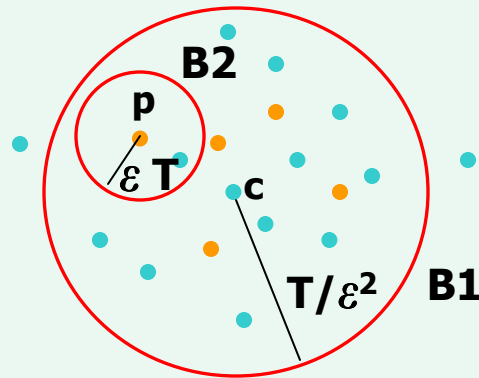
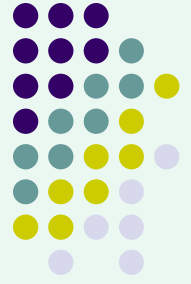


- The closest discrete point, c' , to the non-discrete 1-mean, c , is the discrete 1-mean.
- Randomly Sample $O(1/\epsilon)$ points to obtain approximate non-discrete 1-mean, a .
- Is the closest discrete point, a' , of a $(1+\epsilon)$ -approx. non-discrete 1-mean a good approx to the discrete 1-mean?
 - May be far away from the actual discrete 1-mean
 - $\text{dist}(c, a')$ is comparable to $\text{dist}(c, c')$
 - a' is a $(1+\sqrt{\epsilon})$ approximation to the discrete 1-mean

$$\Delta(P, x) = \Delta(P, c) + |P| \cdot \|c - x\|^2$$



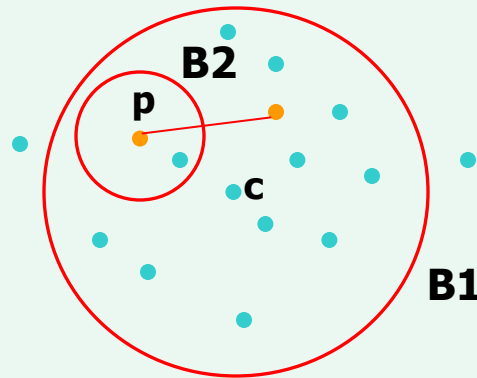
K-Median: Random Sampling Procedure



$$T = \frac{\sum_{x \in P} \|x - c\|}{|P|}$$

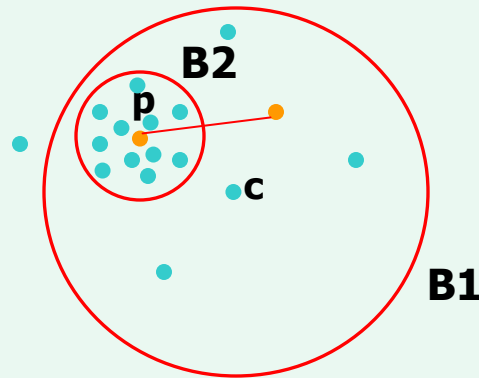
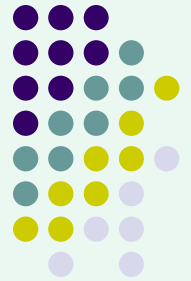
- Sample a set of $1/\varepsilon + 1$ points from P
 - Let $Q =$ first $1/\varepsilon$ points, $p =$ last point
- Let $T =$ Avg. 1-Median cost of P , $c =$ 1-Median
- Let $B1 = B(c, T/\varepsilon^2)$, $B2 = B(p, \varepsilon T)$
- Let $P' =$ points in $B1$

K-Median: Random Sampling Procedure



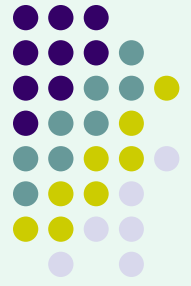
- If there are $\geq 2\epsilon|P'|$ points of P' outside $B2$
 - With const prob., Q contains one of these points
 - $\sum_{q \in Q} d(p, q)$ is an $O(1/\epsilon^{O(1)})$ -approximation of T
 - Use techniques of Badoiu et al to generate candidate center set. Since we have approximation to T , size can be limited to $O(2^{1/\epsilon^{O(1)}})$

K-Median: Random Sampling Procedure



- If there are $< 2\epsilon|P'|$ points of P' outside $B2$
 - Most points are concentrated around p
 - Infact, p is itself a good approximation to the center
 - Add the random sample to the candidate center set

Other Results



- 1-Median [*Kumar, Sabharwal, Sen, ICALP 2005*]
 - With constant probability we can determine a $(1+\varepsilon)$ -approximate 1-median in time $O(2^{1/\varepsilon^{O(1)}} \cdot d)$
 - Running time independent of n
- Weighted Cases
 - Algorithm extends for weighted point sets
 - Running time $O(2^{(k/\varepsilon)^{O(1)}} \cdot n \log^k W \cdot d)$
 - n = no. of distinct points, W = total weight

Thank You

