Fundamental of Data Structures: Design and analysis

Sudebkumar Prasant Pal,
Department of Computer Science and Engineering, IIT Kharagpur, 721302.
ACM Summer School on Cryptology Research, ISI Kolkata

June 13, 2018

Scope of the lecture

- **Binary search trees, Range trees and Kd-trees**
  We consider 1-d and 2-d range queries for point sets.

Scope of the lecture

- **Binary search trees, Range trees and Kd-trees**
  We consider 1-d and 2-d range queries for point sets.
- **Interval trees and Segment tree**
  Interval trees for reporting all intervals on a line containing a given query point on the line.

Scope of the lecture

- **Binary search trees, Range trees and Kd-trees**
  We consider 1-d and 2-d range queries for point sets.
- **Interval trees and Segment tree**
  Interval trees for reporting all intervals on a line containing a given query point on the line.
- **Paradigm of Sweep algorithms**
  For reporting intersections of line segments, and for computing visible regions.

Scope of the lecture

- **Binary search trees, Range trees and Kd-trees**
  We consider 1-d and 2-d range queries for point sets.
- **Interval trees and Segment tree**
  Interval trees for reporting all intervals on a line containing a given query point on the line.
- **Paradigm of Sweep algorithms**
  For reporting intersections of line segments, and for computing visible regions.
- **Finger searching**
  Computing shortest path trees in linear time.

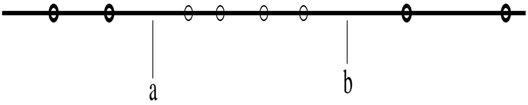
Scope of the lecture

- **Binary search trees, Range trees and Kd-trees**
  We consider 1-d and 2-d range queries for point sets.
- **Interval trees and Segment tree**
  Interval trees for reporting all intervals on a line containing a given query point on the line.
- **Paradigm of Sweep algorithms**
  For reporting intersections of line segments, and for computing visible regions.
- **Finger searching**
  Computing shortest path trees in linear time.
- **Hierarchical searching**
  Planar point location

Scope of the lecture

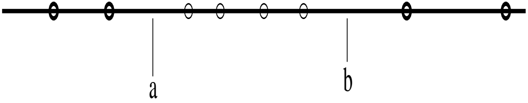
- **Binary search trees, Range trees and Kd-trees**
  We consider 1-d and 2-d range queries for point sets.
- **Interval trees and Segment tree**
  Interval trees for reporting all intervals on a line containing a given query point on the line.
- **Paradigm of Sweep algorithms**
  For reporting intersections of line segments, and for computing visible regions.
- **Finger searching**
  Computing shortest path trees in linear time.
- **Hierarchical searching**
  Planar point location
- **$\frac{1}{r}$-cuttings, many faces complexity, incidences**
  Planar point location

1-dimensional Range searching



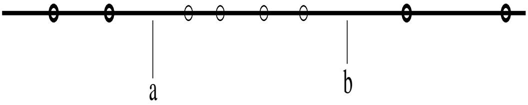
- Problem: Given a set P of n points $\{p_1, p_2, \cdots, p_n\}$ on the real line, report points of P that lie in the range $[a, b]$, $a \leq b$.

1-dimensional Range searching



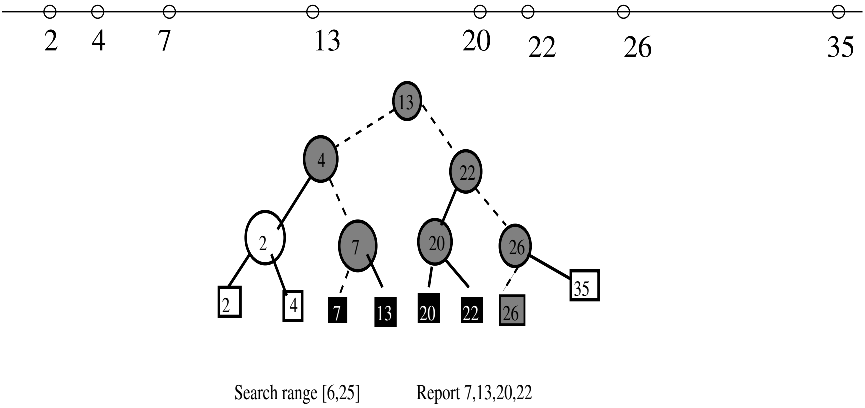
- Problem: Given a set P of n points $\{p_1, p_2, \cdots, p_n\}$ on the real line, report points of P that lie in the range $[a, b]$, $a \leq b$.
- Using binary search on an array we can answer such a query in $O(\log n + k)$ time where k is the number of points of P in $[a, b]$.

1-dimensional Range searching



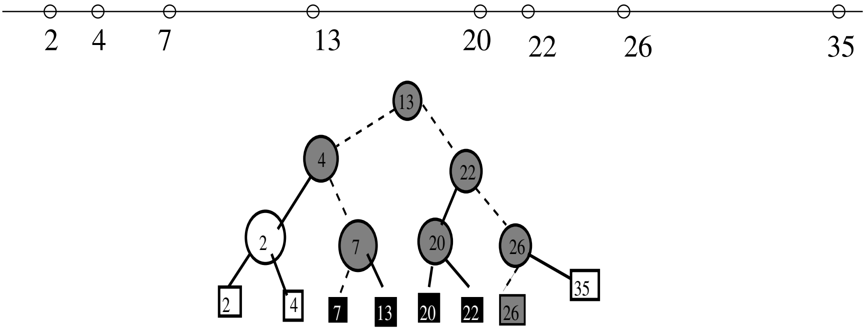
- Problem: Given a set P of n points $\{p_1, p_2, \cdots, p_n\}$ on the real line, report points of P that lie in the range $[a, b]$, $a \leq b$.
- Using binary search on an array we can answer such a query in $O(\log n + k)$ time where k is the number of points of P in $[a, b]$.
- However, when we permit insertion or deletion of points, we cannot use an array answering queries so efficiently.

1-dimensional Range searching



- We use a *binary leaf search tree* where leaf nodes store the points on the line, sorted by x-coordinates.

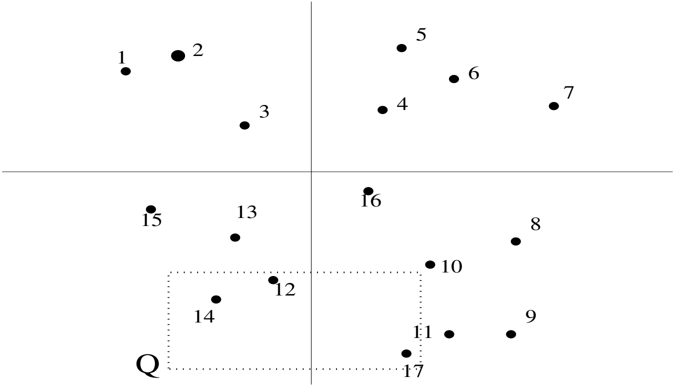
1-dimensional Range searching



Search range [6,25] Report 7,13,20,22

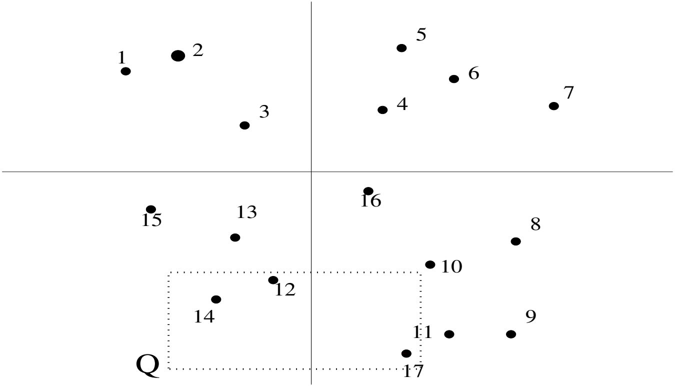
- We use a *binary leaf search tree* where leaf nodes store the points on the line, sorted by x-coordinates.
- Each internal node stores the x-coordinate of the rightmost point in its left subtree for guiding search.

2-dimensional Range Searching



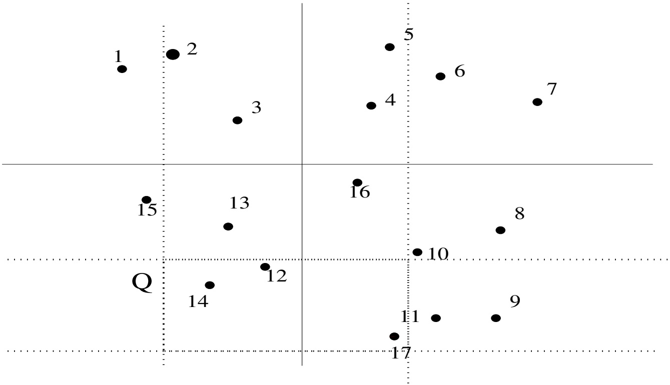
- Problem: Given a set P of n points in the plane, report points inside a query rectangle Q whose sides are parallel to the axes.

2-dimensional Range Searching



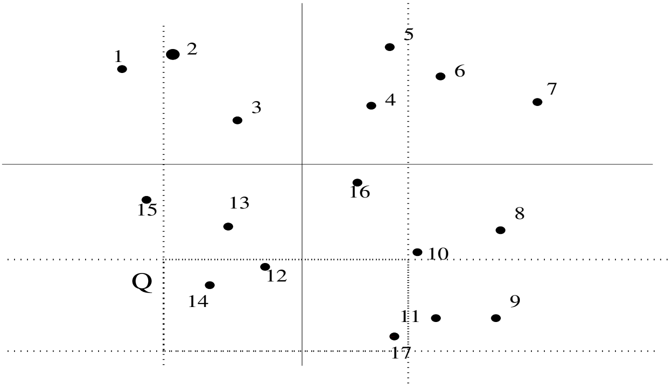
- Problem: Given a set P of n points in the plane, report points inside a query rectangle Q whose sides are parallel to the axes.
- Here, the points inside R are 14, 12 and 17.

2-dimensional Range Searching



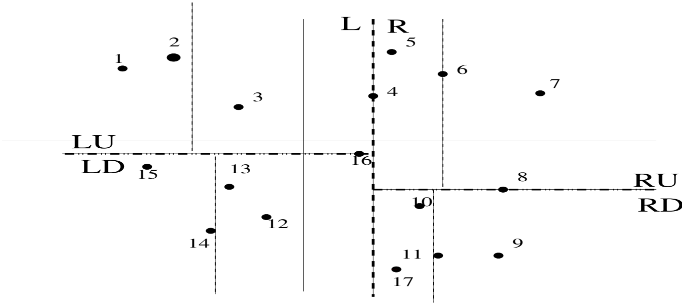
- Using two 1-d range queries, one along each axis, solves the 2-d range query.

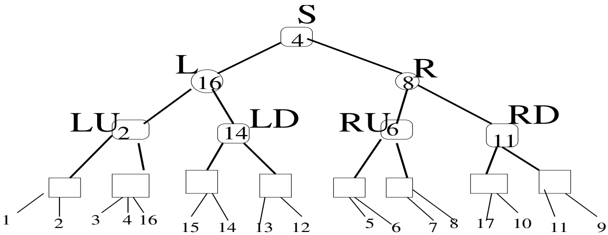
2-dimensional Range Searching



- Using two 1-d range queries, one along each axis, solves the 2-d range query.
- The cost incurred may exceed the actual output size of the 2-d range query.

2-dimensional Range Searching: Kd-trees





2-dimensional Range Searching

- The tree T is a perfectly height-balanced binary search tree with alternate layers of nodes spitting subsets of points in P using x- and y- coordinates, respectively as follows.

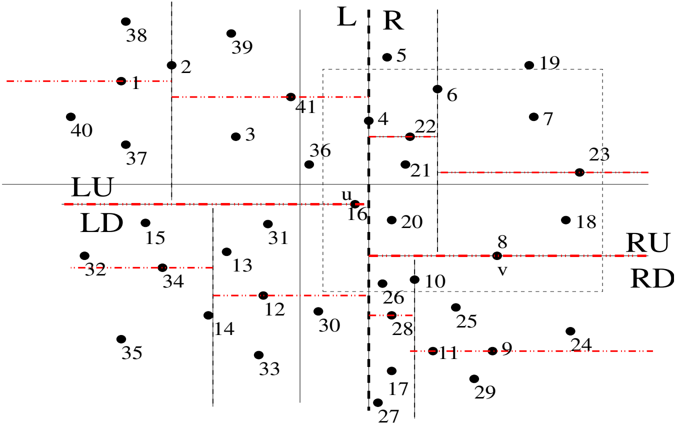
2-dimensional Range Searching

- The tree T is a perfectly height-balanced binary search tree with alternate layers of nodes spitting subsets of points in P using x- and y- coordinates, respectively as follows.
- The point r stored in the root vertex T splits the set S into two roughly equal sized sets L and R using the median x-cooordinate $xmedian(S)$ of points in S, so that all points in L (R) have abscissae less than or equal to (strictly greater than) $xmedian(S)$.

2-dimensional Range Searching

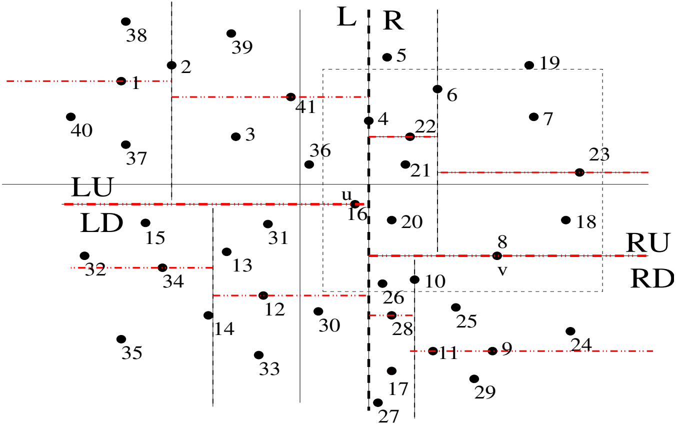
- The tree T is a perfectly height-balanced binary search tree with alternate layers of nodes spitting subsets of points in P using x- and y- coordinates, respectively as follows.
- The point r stored in the root vertex T splits the set S into two roughly equal sized sets L and R using the median x-cooordinate $xmedian(S)$ of points in S, so that all points in L (R) have abscissae less than or equal to (strictly greater than) $xmedian(S)$.
- The entire plane is called the $region(r)$.

Answering rectangle queries



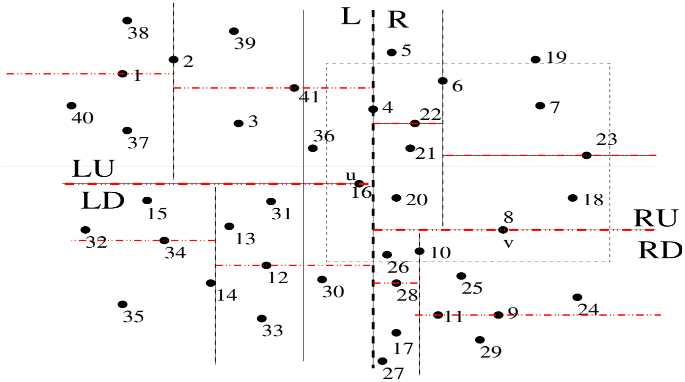
- A query rectangle Q may overlap a region or completely contain a region.

Answering rectangle queries



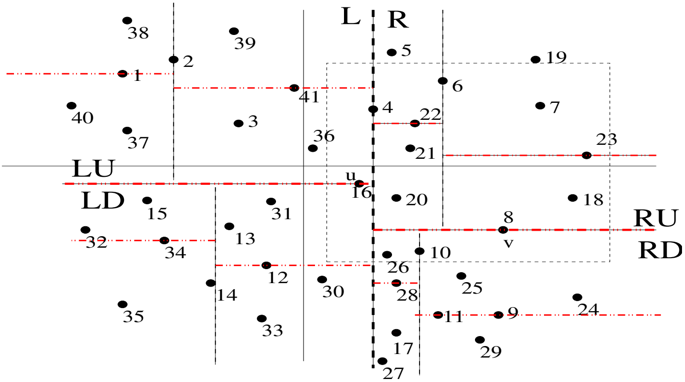
- A query rectangle Q may overlap a region or completely contain a region.
- If R contains the entire bounded $region(p)$ of a point p defining a node of T then report all points in $region(p)$.

2-dimensional Range Searching: Kd-trees [1]



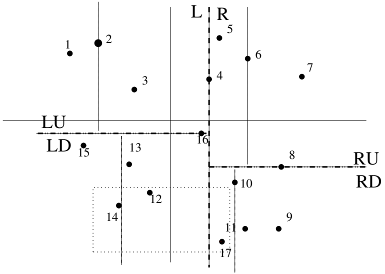
- The set L (R) is split into two roughly equal sized subsets LU and LD (RU and RD), using point u (v) that has the median y-coordinate in the set L (R), and including u in LU (RU).

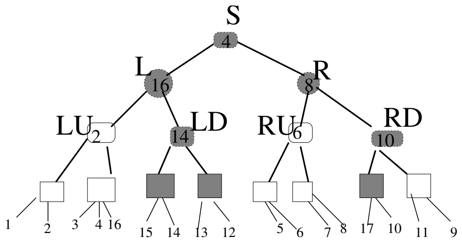
2-dimensional Range Searching: Kd-trees [1]



- The set L (R) is split into two roughly equal sized subsets LU and LD (RU and RD), using point u (v) that has the median y-coordinate in the set L (R), and including u in LU (RU).
- The entire halfplane containing set L (R) is called the $region(u)$ ($region(v)$).

Time complexity of rectangle queries





Time complexity of output point reporting

- Reporting points within R contributes to the output size k for the query.

Time complexity of output point reporting

- Reporting points within R contributes to the output size k for the query.
- No leaf level region in T has more than 2 points.

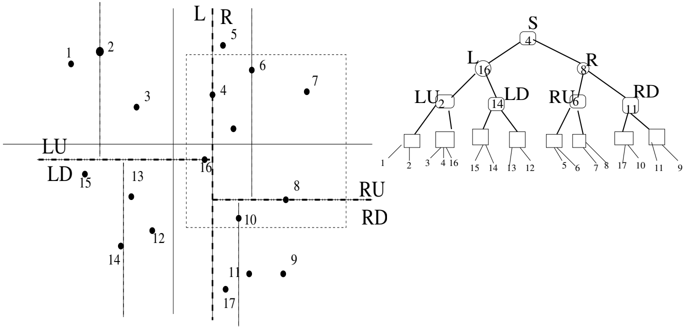
Time complexity of output point reporting

- Reporting points within R contributes to the output size k for the query.
- No leaf level region in T has more than 2 points.
- So, the cost of inspecting points outside R but within the intersection of leaf level regions of T can be charged to the internal nodes traversed in T.

Time complexity of output point reporting

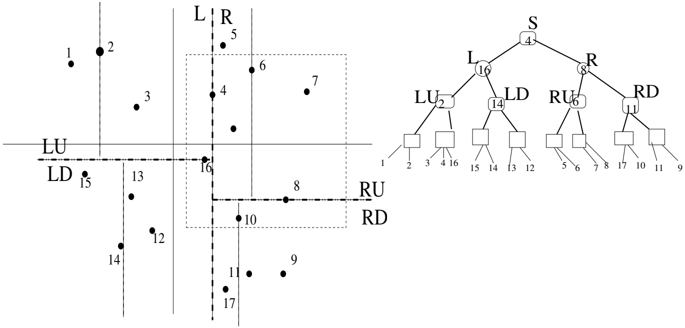
- Reporting points within R contributes to the output size k for the query.
- No leaf level region in T has more than 2 points.
- So, the cost of inspecting points outside R but within the intersection of leaf level regions of T can be charged to the internal nodes traversed in T.
- This cost is borne for all leaf level regions intersected by R.

Time complexity of traversing the tree



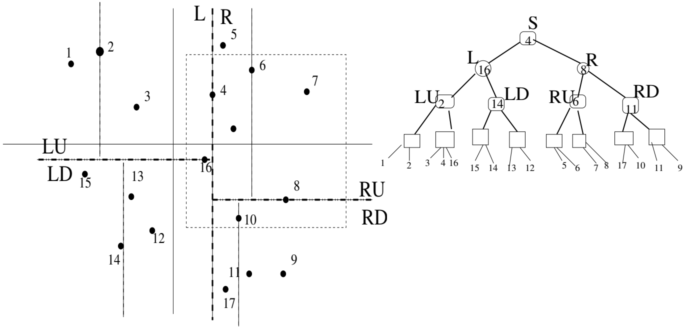
- It is sufficient to determine the upper bound on the number of (internal) nodes whose regions are intersected by a single vertical (horizontal) line.

Time complexity of traversing the tree



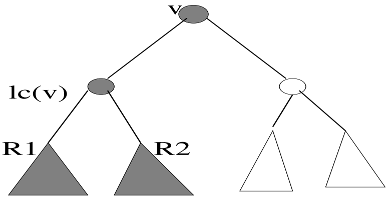
- It is sufficient to determine the upper bound on the number of (internal) nodes whose regions are intersected by a single vertical (horizontal) line.
- Any vertical line intersecting S can intersect either L or R but not both, but it can meet both RU and RD (LU and LD).

Time complexity of traversing the tree



- It is sufficient to determine the upper bound on the number of (internal) nodes whose regions are intersected by a single vertical (horizontal) line.
- Any vertical line intersecting S can intersect either L or R but not both, but it can meet both RU and RD (LU and LD).
- Any horizontal line intersecting R can intersect either RU or RD but not both, but it can meet both children of RU (RD).

Time complexity of rectangle queries

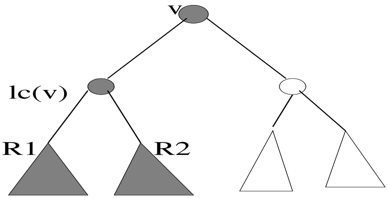


- Therefore, the time complexity $T(n)$ for an n-vertex Kd-tree obeys the recurrence relation

$$T(n) = 2 + 2T(\frac{n}{4})$$

$$T(1) = 1$$

Time complexity of rectangle queries



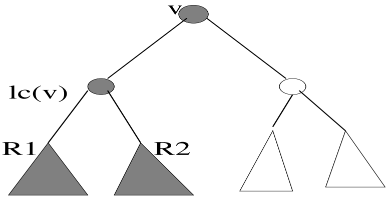
- Therefore, the time complexity $T(n)$ for an n-vertex Kd-tree obeys the recurrence relation

$$T(n) = 2 + 2T(\frac{n}{4})$$

$$T(1) = 1$$

- The solution for $T(n) = O(\sqrt{(}n))$.

Time complexity of rectangle queries



- Therefore, the time complexity $T(n)$ for an n-vertex Kd-tree obeys the recurrence relation

$$T(n) = 2 + 2T(\frac{n}{4})$$

$$T(1) = 1$$

- The solution for $T(n) = O(\sqrt(n))$.
- The total cost of reporting k points in R is therefore $O(\sqrt(n) + k)$.

Range searching with Kd-trees and range trees

- Given a set S of n points in the plane, we can construct a Kd-tree in $O(n \log n)$ time and $O(n)$ space, so that rectangle queries can be executed in $O(\sqrt{n} + k)$ time. Here, the number of points in the query rectangle is k.

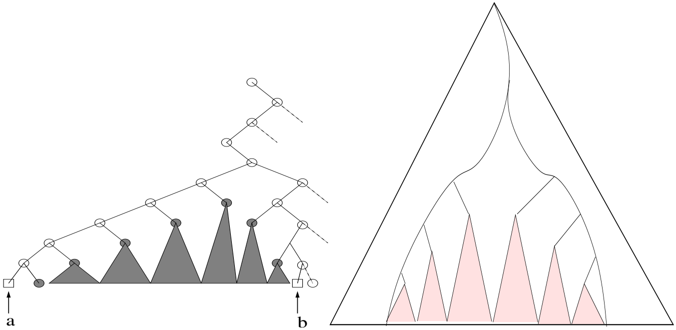
Range searching with Kd-trees and range trees

- Given a set S of n points in the plane, we can construct a Kd-tree in $O(n \log n)$ time and $O(n)$ space, so that rectangle queries can be executed in $O(\sqrt{n} + k)$ time. Here, the number of points in the query rectangle is k.
- Given a set S of n points in the plane, we can construct a range tree in $O(n \log n)$ time and space, so that rectangle queries can be executed in $O(\log^2 n + k)$ time.

Range searching with Kd-trees and range trees

- Given a set S of n points in the plane, we can construct a Kd-tree in $O(n \log n)$ time and $O(n)$ space, so that rectangle queries can be executed in $O(\sqrt{n} + k)$ time. Here, the number of points in the query rectangle is k.
- Given a set S of n points in the plane, we can construct a range tree in $O(n \log n)$ time and space, so that rectangle queries can be executed in $O(\log^2 n + k)$ time.
- The query time can be improved to $O(\log n + k)$ using the technique of *fractional cascading*.

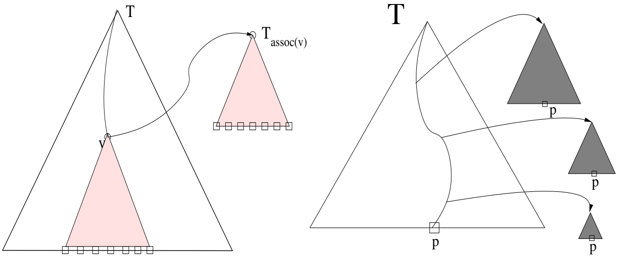
Range searching in the plane using range trees



Given a 2-d rectangle query $[a, b] X [c, d]$, we can identify subtrees whose leaf nodes are in the range $[a, b]$ along the X-direction.

Only a subset of these leaf nodes lie in the range $[c, d]$ along the Y-direction.

Range searching in the plane using range trees

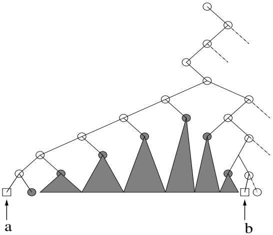


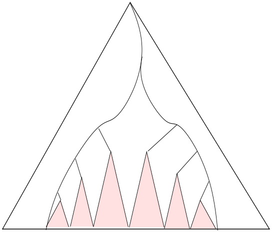
$T_{assoc(v)}$ is a binary search tree on y-coordinates for points in the leaf nodes of the subtree tooted at v in the tree T.

The point p is duplicated in $T_{assoc(v)}$ for each v on the search path for p in tree T.

The total space requirements is therefore $O(n \log n)$.

Range searching in the plane using range trees

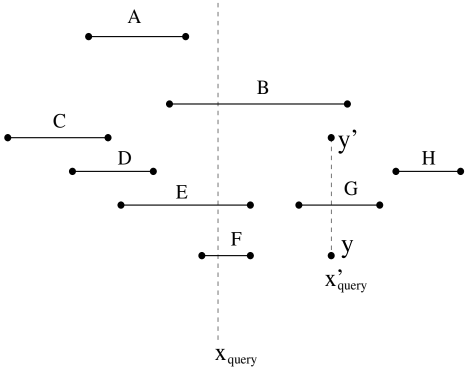




We perform 1-d range queries with the y-range $[c, d]$ in each of the subtrees adjacent to the left and right search paths for the x-range $[a, b]$ in the tree T.

Since the search path is $O(\log n)$ in size, and each y-range query requires $O(\log n)$ time, the total cost of searching is $O(\log^2 n)$. The reporting cost is $O(k)$ where k points lie in the query rectangle.

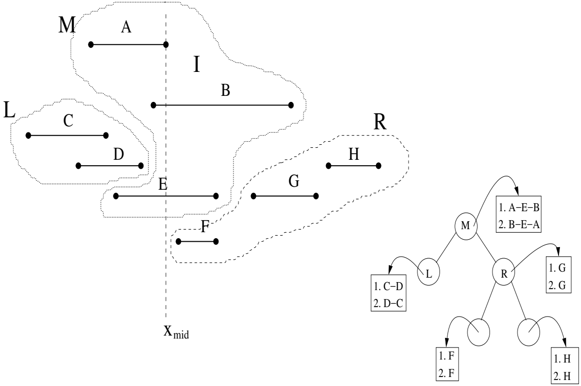
Finding intervals containing a query point



Simpler queries ask for reporting all intervals intersecting the vertical line $X = x_{query}$.

More difficult queries ask for reporting all intervals intersecting a vertical segment joining (x'_{query}, y) and (x'_{query}, y').

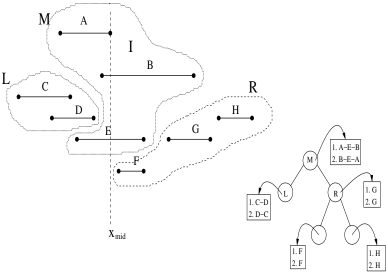
Computing the interval tree



The set M has intervals intersecting the vertical line $X = x_{mid}$, where x_{mid} is the median of the x-coordinates of the $2n$ endpoints.

The root node has intervals M sorted in two independent orders (i) by right end points (B-E-A), and (ii) left end points (A-E-B).

Answering queries using an interval tree



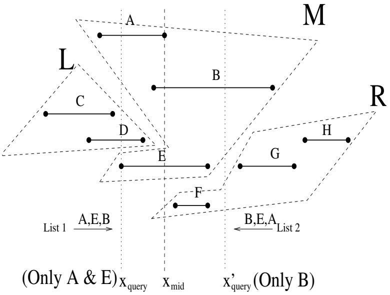
The set L and R have at most n endpoints each.

So they have at most $\frac{n}{2}$ intervals each.

Clearly, the cost of (recursively) building the interval tree is $O(n \log n)$.

The space required is linear.

Answering queries using an interval tree

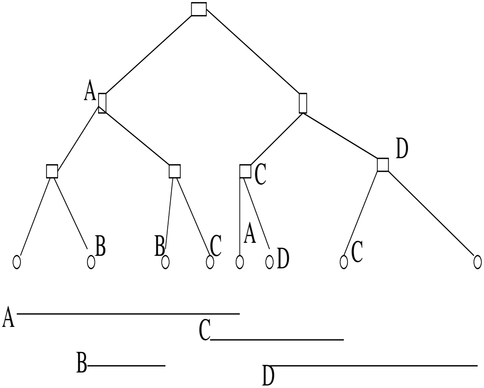


For $x_{query} < x_{mid}$, we do not traverse subtree for subset R.

For $x'_{query} > x_{mid}$, we do not traverse subtree for subset L.

Clearly, the cost of reporting the k intervals is $O(\log n + k)$.

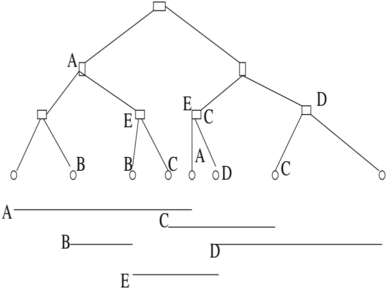
Introducing the segment tree



For an interval which spans the entire range $inv(v)$, we mark only internal node v in the segment tree, and not any descendant of v.

We never mark any ancestor of a marked node.

Representing intervals in the segment tree

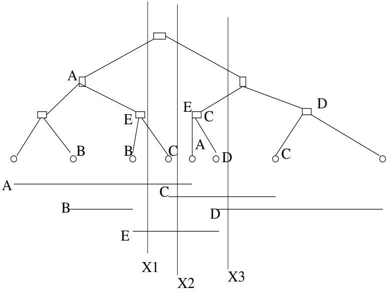


At each level, at most two internal nodes are marked for any given interval.

Along a root to leaf path an interval is stored only once.

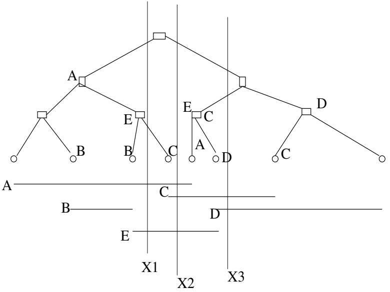
The space requirement is therefore $O(n \log n)$.

Reporting intervals containing a given query point



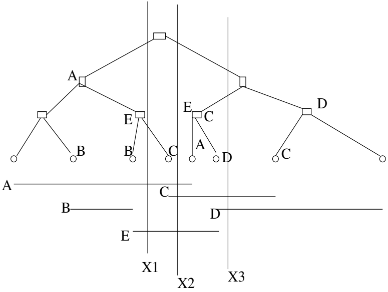
- Search the path in the tree reaching the leaf for the given query point.

Reporting intervals containing a given query point



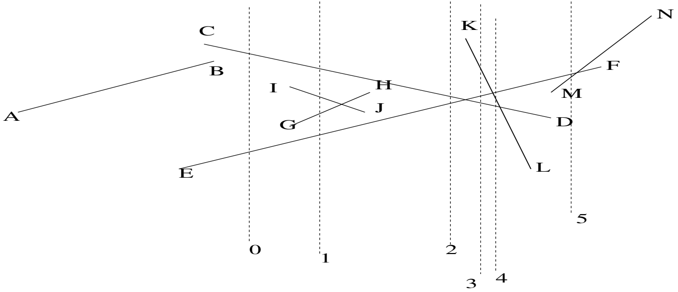
- Search the path in the tree reaching the leaf for the given query point.
- Report all intervals that appear stored on the search path.

Reporting intervals containing a given query point



- Search the path in the tree reaching the leaf for the given query point.
- Report all intervals that appear stored on the search path.
- If k intervals contain the query point then the cost incurred is $O(\log n + k)$.

Reporting segments intersections



Problem: Given a set S of n line segments in the plane, report all intersections between the segments.

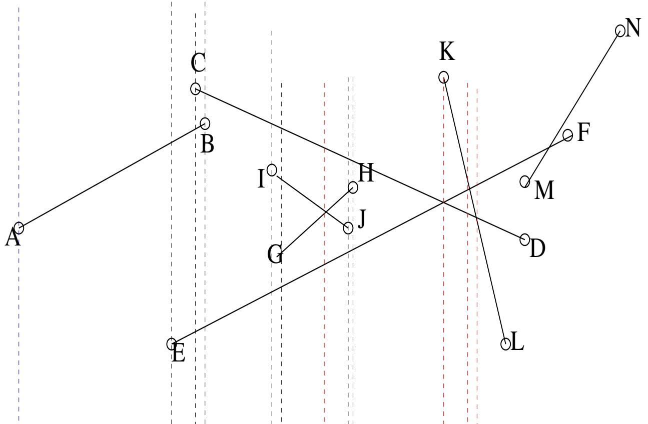
Check all pairs in $O(n^2)$ time.

A vertical line just before any intersection meets intersecting segments in an empty, intersection-free segment.

Detect intersections by checking consecutive pairs of segments along a vertical line.

This way, each intersection point can be detected.

Sweeping steps: Endpoints and intersection points



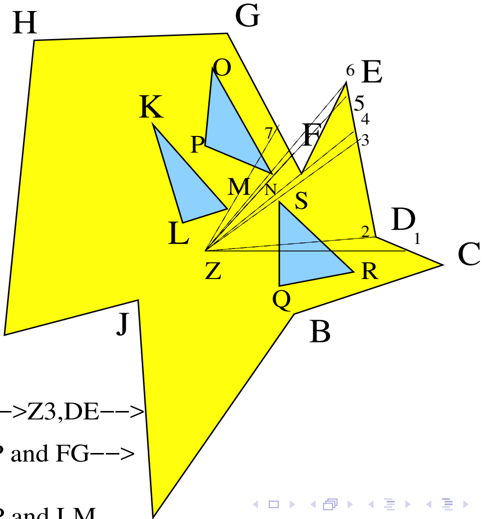
AB−>AB,EF−>CD,AB,EF−>CD,EF−>CD,IJ,EF−>CD,IJ,GH,EF−>CD,GH,IJ,EF
CD,GH,EF−>CD,EF−>EF,CD

Step 1

SQ,SR,DC,1-->SQ,SR,DE,2-->DE,3--

FG,FE,DE,4-->NP,NO,FG,FE,DE,5-->

NP,NO,FG,FE,DE,6-->LM,MK,NP,NO,FG,7



Z1 ,SQ-->Z2,SQ-->Z3,DE-->

Z4,FG and DE-->Z5,NP and FG-->

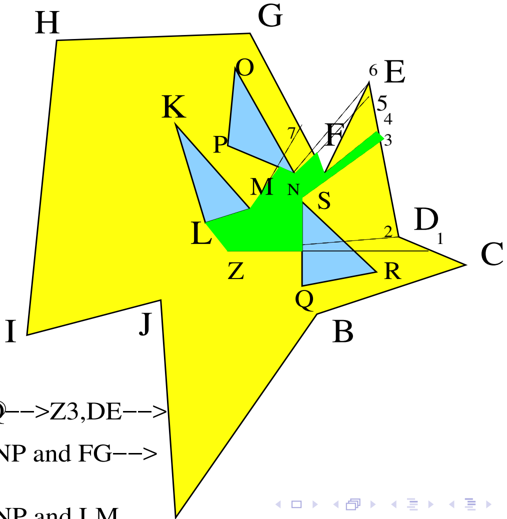
Z6,NP-->Z7, NP and LM

Step 2

SQ,SR,DC,1−−>SQ,SR,DE,2−−>DE,3−-

FG,FE,DE,4−−>NP,NO,FG,FE,DE,5−−>

NP,NO,FG,FE,DE,6−−>LM,MK,NP,NO,FG,7

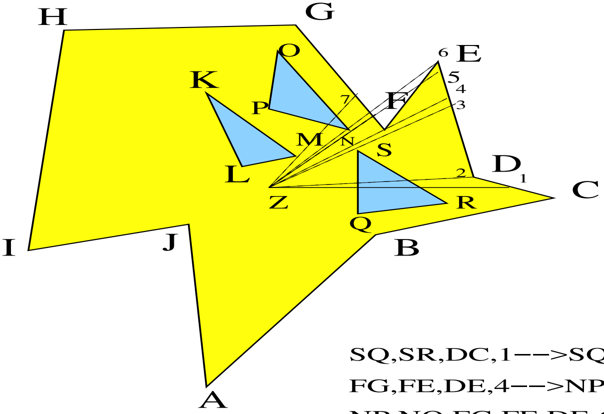


Z1 ,SQ−−>Z2,SQ−−>Z3,DE−−>

Z4,FG and DE−−>Z5,NP and FG−−>

Z6,NP−−>Z7, NP and LM

STEP 3



SQ,SR,DC,1−−>SQ,SR,DE
FG,FE,DE,4−−>NP,NO,FG
NP,NO,FG,FE,DE,6−−>LM

Z1 ,SQ−−>Z2,SQ−−>Z3,DE−−>Z4,FG and DE−−>Z5,NP and FG−−>
Z6,NP−−>Z7, NP and LM

Many faces complexity in an arangement of lines in the plane.

- We consider the problem of estimating the number $K(m, n)$, the *many faces complexity* of edges of m faces in an *arrangement* of n lines.

Many faces complexity in an arangement of lines in the plane.

- We consider the problem of estimating the number $K(m, n)$, the *many faces complexity* of edges of m faces in an *arrangement* of n lines.
- One way to visualize is to consider a set P of m points in the plane, and a set L of n lines in the plane. The (at most) m faces are determined by the m points in the arrangement $A(L)$ of lines in L.

Many faces complexity in an arangement of lines in the plane.

- We consider the problem of estimating the number $K(m,n)$, the *many faces complexity* of edges of m faces in an *arrangement* of n lines.
- One way to visualize is to consider a set P of m points in the plane, and a set L of n lines in the plane. The (at most) m faces are determined by the m points in the arrangement $A(L)$ of lines in L.
- We get the inferior upper bound (known as the Canham bound) of $O(m\sqrt{n}+n)$ using the *forbidden subgraph* property of the *bipartite incidence graph* of lines and faces in an arrangement of lines.

Many faces complexity in an arangement of lines in the plane.

- We consider the problem of estimating the number $K(m, n)$, the *many faces complexity* of edges of m faces in an *arrangement* of n lines.
- One way to visualize is to consider a set P of m points in the plane, and a set L of n lines in the plane. The (at most) m faces are determined by the m points in the arrangement $A(L)$ of lines in L.
- We get the inferior upper bound (known as the Canham bound) of $O(m\sqrt{n} + n)$ using the *forbidden subgraph* property of the *bipartite incidence graph* of lines and faces in an arrangement of lines.
- The forbidden subgraph is $K_{2,5}$. Using the result by Kovari, Sos and Turan (Theorem 9.6 in [4]) for such forbidden component subgraphs, we get the above loose upper bound. See Pach and Agarwal [4], for a proof of the Kovari, Sos and Turan result.

- We proceed to use a divide-and-conquer approach as follows, in order to derive a much better bound that also asymptotically matches the best known lower bounds (see Theorem 11.9 of [4]).

- We proceed to use a divide-and-conquer approach as follows, in order to derive a much better bound that also asymptotically matches the best known lower bounds (see Theorem 11.9 of [4]).
- Suppose we form an arrangement with a subset R of size r of the set L of n lines. The arrangement $A(L)$ is of our interest.

- We proceed to use a divide-and-conquer approach as follows, in order to derive a much better bound that also asymptotically matches the best known lower bounds (see Theorem 11.9 of [4]).
- Suppose we form an arrangement with a subset R of size r of the set L of n lines. The arrangement $A(L)$ is of our interest.
- However, we may first convert $A(R)$ into a trapezoidal map $A^*(R)$ with $k = s \leq 3r^2$ *trapezoids/triangles* as faces, by dropping *plumbline* vertical segments from vertices and intersection points of $A(R)$.

- We proceed to use a divide-and-conquer approach as follows, in order to derive a much better bound that also asymptotically matches the best known lower bounds (see Theorem 11.9 of [4]).
- Suppose we form an arrangement with a subset R of size r of the set L of n lines. The arrangement $A(L)$ is of our interest.
- However, we may first convert $A(R)$ into a trapezoidal map $A^*(R)$ with $k = s \leq 3r^2$ *trapezoids/triangles* as faces, by dropping *plumbline* vertical segments from vertices and intersection points of $A(R)$.
- It is nice if not too many lines from $L \setminus R$ intersect an arbitrary trapezoid Δ_j of $A^*(R)$, where the (fixed) point $p_j \in P$ lies in the (unique) trapezoid Δ_j.

- We proceed to use a divide-and-conquer approach as follows, in order to derive a much better bound that also asymptotically matches the best known lower bounds (see Theorem 11.9 of [4]).
- Suppose we form an arrangement with a subset R of size r of the set L of n lines. The arrangement $A(L)$ is of our interest.
- However, we may first convert $A(R)$ into a trapezoidal map $A^*(R)$ with $k = s \leq 3r^2$ *trapezoids/triangles* as faces, by dropping *plumbline* vertical segments from vertices and intersection points of $A(R)$.
- It is nice if not too many lines from $L \setminus R$ intersect an arbitrary trapezoid Δ_j of $A^*(R)$, where the (fixed) point $p_j \in P$ lies in the (unique) trapezoid Δ_j.
- Even if this trapezoid is intersected by q_j lines, we wish to have the expectation $E(q_j) = O(\frac{n}{r})$, where the expectation is over all the $\binom{n}{r}$ random samples $R \subset L$.

- We proceed to use a divide-and-conquer approach as follows, in order to derive a much better bound that also asymptotically matches the best known lower bounds (see Theorem 11.9 of [4]).
- Suppose we form an arrangement with a subset R of size r of the set L of n lines. The arrangement $A(L)$ is of our interest.
- However, we may first convert $A(R)$ into a trapezoidal map $A^*(R)$ with $k = s \leq 3r^2$ *trapezoids/triangles* as faces, by dropping *plumbline* vertical segments from vertices and intersection points of $A(R)$.
- It is nice if not too many lines from $L \setminus R$ intersect an arbitrary trapezoid Δ_j of $A^*(R)$, where the (fixed) point $p_j \in P$ lies in the (unique) trapezoid Δ_j.
- Even if this trapezoid is intersected by q_j lines, we wish to have the expectation $E(q_j) = O(\frac{n}{r})$, where the expectation is over all the $\binom{n}{r}$ random samples $R \subset L$.
- This is indeed possible and we show this later using combinatorial arguments; this is a technical result of independent and deep import.

- Let the face Δ_i of $A^*(R)$ intersect n_i lines of $L \setminus R$ and contain m_i of the m points from the point set P.

- Let the face Δ_i of $A^*(R)$ intersect n_i lines of $L \setminus R$ and contain m_i of the m points from the point set P.
- Here, the set L_i of lines from $L \setminus R$ that intersect Δ_i, form an arrangement $A(L_i)$; the convex faces (cells) in $A(L_i)$ are just the faces of arrangements $A(L)$ or $A(R)$.

- Let the face Δ_i of $A^*(R)$ intersect n_i lines of $L \setminus R$ and contain m_i of the m points from the point set P.
- Here, the set L_i of lines from $L \setminus R$ that intersect Δ_i, form an arrangement $A(L_i)$; the convex faces (cells) in $A(L_i)$ are just the faces of arrangements $A(L)$ or $A(R)$.
- In contrast, by the very definition of A^*, all $A^*(R)$, $A^*(L)$ and $A^*(L_i)$ have only trapezoids and triangles for faces (or cells).

- Let the face Δ_i of $A^*(R)$ intersect n_i lines of $L \setminus R$ and contain m_i of the m points from the point set P.
- Here, the set L_i of lines from $L \setminus R$ that intersect Δ_i, form an arrangement $A(L_i)$; the convex faces (cells) in $A(L_i)$ are just the faces of arrangements $A(L)$ or $A(R)$.
- In contrast, by the very definition of A^*, all $A^*(R)$, $A^*(L)$ and $A^*(L_i)$ have only trapezoids and triangles for faces (or cells).
- Now, using recursion we write $K(m,n) \leq \sum_{i=1}^{s} K(m_i, n_i) + O(nr)$ We explain the $O(nr)$ term using the zone theorem and its non-trivial application

- Let the face Δ_i of $A^*(R)$ intersect n_i lines of $L \setminus R$ and contain m_i of the m points from the point set P.
- Here, the set L_i of lines from $L \setminus R$ that intersect Δ_i, form an arrangement $A(L_i)$; the convex faces (cells) in $A(L_i)$ are just the faces of arrangements $A(L)$ or $A(R)$.
- In contrast, by the very definition of A^*, all $A^*(R)$, $A^*(L)$ and $A^*(L_i)$ have only trapezoids and triangles for faces (or cells).
- Now, using recursion we write $K(m, n) \leq \sum_{i=1}^{s} K(m_i, n_i) + O(nr)$ We explain the $O(nr)$ term using the zone theorem and its non-trivial application
- Using the Canham bound, can write $K(m, n) \leq \sum_{i=1}^{s} (m_i \sqrt{n}_i + n_i) + O(nr)$

- We use the existence of random sample R of size r and establish the upper bound $\sum_{i=1}^{s} m_i(n_i)^\alpha = O(m(\frac{n}{r})^\alpha)$ by showing that the expectation of the summation in the LHS above is bounded as $O(m(\frac{n}{r})^\alpha)$.

- We use the existence of random sample R of size r and establish the upper bound $\sum_{i=1}^{s} m_i(n_i)^\alpha = O(m(\frac{n}{r})^\alpha)$ by showing that the expectation of the summation in the LHS above is bounded as $O(m(\frac{n}{r})^\alpha)$.
- This bound is established in part (ii) of Theorem 11.2 in [4]; part (i) of the same theorem claims that $\Sigma_{i=1}^{s} n_i \leq c_1 nr$, which holds for any $R \subset L$, where $|R| = r$.

- We use the existence of random sample R of size r and establish the upper bound $\sum_{i=1}^{s} m_i(n_i)^{\alpha} = O(m(\frac{n}{r})^{\alpha})$ by showing that the expectation of the summation in the LHS above is bounded as $O(m(\frac{n}{r})^{\alpha})$.
- This bound is established in part (ii) of Theorem 11.2 in [4]; part (i) of the same theorem claims that $\Sigma_{i=1}^{s} n_i \leq c_1 nr$, which holds for any $R \subset L$, where $|R| = r$.
- So, we can write $K(m, n) \leq O(m(n/r)^{\frac{1}{2}}) + O(nr)$

- We use the existence of random sample R of size r and establish the upper bound $\sum_{i=1}^{s} m_i(n_i)^\alpha = O(m(\frac{n}{r})^\alpha)$ by showing that the expectation of the summation in the LHS above is bounded as $O(m(\frac{n}{r})^\alpha)$.
- This bound is established in part (ii) of Theorem 11.2 in [4]; part (i) of the same theorem claims that $\Sigma_{i=1}^{s} n_i \leq c_1 nr$, which holds for any $R \subset L$, where $|R| = r$.
- So, we can write $K(m, n) \leq O(m(n/r)^{\frac{1}{2}}) + O(nr)$
- Now, by setting $r = min(n, \frac{m^{\frac{2}{3}}}{n^{\frac{1}{3}}})$ we get $nr = (mn)^{\frac{2}{3}}$ and therefore, $K(m, n) = O(m^{\frac{2}{3}} n^{\frac{2}{3}} + n)$.

Planar embeddings and *crossing numbers*

- An *embedding* of a graph $G = (V, E)$ in the plane is a planar representation of it, where each vertex is represented by a point in the plane, and each edge $\{u, v\}$ is represented by a curve connecting the points corresponding to the vertices u and v.

Planar embeddings and *crossing numbers*

- An *embedding* of a graph $G = (V, E)$ in the plane is a planar representation of it, where each vertex is represented by a point in the plane, and each edge $\{u, v\}$ is represented by a curve connecting the points corresponding to the vertices u and v.
- The *crossing number of such an embedding* is the number of pairs of intersecting curves that correspond to pairs of edges with no common endpoints.

Planar embeddings and *crossing numbers*

- An *embedding* of a graph $G = (V, E)$ in the plane is a planar representation of it, where each vertex is represented by a point in the plane, and each edge $\{u, v\}$ is represented by a curve connecting the points corresponding to the vertices u and v.
- The *crossing number of such an embedding* is the number of pairs of intersecting curves that correspond to pairs of edges with no common endpoints.
- The crossing number $cr(G)$ of G is the minimum possible crossing number in an embedding of it in the plane.

Planar embeddings and *crossing numbers*

- An *embedding* of a graph $G = (V, E)$ in the plane is a planar representation of it, where each vertex is represented by a point in the plane, and each edge $\{u, v\}$ is represented by a curve connecting the points corresponding to the vertices u and v.
- The *crossing number of such an embedding* is the number of pairs of intersecting curves that correspond to pairs of edges with no common endpoints.
- The crossing number $cr(G)$ of G is the minimum possible crossing number in an embedding of it in the plane.
- The only and trivial planar embedding of the graph K_3 has crossing number 0. Hence it is a planar graph.

Planar embeddings and *crossing numbers*

- An *embedding* of a graph $G = (V, E)$ in the plane is a planar representation of it, where each vertex is represented by a point in the plane, and each edge $\{u, v\}$ is represented by a curve connecting the points corresponding to the vertices u and v.
- The *crossing number of such an embedding* is the number of pairs of intersecting curves that correspond to pairs of edges with no common endpoints.
- The crossing number $cr(G)$ of G is the minimum possible crossing number in an embedding of it in the plane.
- The only and trivial planar embedding of the graph K_3 has crossing number 0. Hence it is a planar graph.
- The complete graph K_4 of four vertices has crossing number o as well. In every planar embedding, the graph K_5 has at least one pair of edges crossing. Hence, it is a non-planar graph. $K_{3,3}$ also has crossing number 1.

- Kuratowski showed 1930 that a graph is planar if and only if it has no subgraph *homeomorphic* to K_5 or $K_{3,3}$.

- Kuratowski showed 1930 that a graph is planar if and only if it has no subgraph *homeomorphic* to K_5 or $K_{3,3}$.
- The following *Crossing Number Theorem* was proved by Ajtai, Chvatal, Newborn and Szemeredi in 1982, and independently, by Leighton.

- Kuratowski showed 1930 that a graph is planar if and only if it has no subgraph *homeomorphic* to K_5 or $K_{3,3}$.
- The following *Crossing Number Theorem* was proved by Ajtai, Chvatal, Newborn and Szemeredi in 1982, and independently, by Leighton.
- The crossing number of any simple graph (i.e., a graph with no multi-edges or no self-loops) with $|E| \geq 4|V|$ is at least $|E|^3/64|V|^2$.

- Kuratowski showed 1930 that a graph is planar if and only if it has no subgraph *homeomorphic* to K_5 or $K_{3,3}$.
- The following *Crossing Number Theorem* was proved by Ajtai, Chvatal, Newborn and Szemeredi in 1982, and independently, by Leighton.
- The crossing number of any simple graph (i.e., a graph with no multi-edges or no self-loops) with $|E| \geq 4|V|$ is at least $|E|^3/64|V|^2$.
- We know Eulers formula for any spherical polyhedron, with $|V|$ vertices, $|E|$ edges and $|F|$ faces, $|V| - |E| + |F| = 2$.

- Kuratowski showed 1930 that a graph is planar if and only if it has no subgraph *homeomorphic* to K_5 or $K_{3,3}$.
- The following *Crossing Number Theorem* was proved by Ajtai, Chvatal, Newborn and Szemeredi in 1982, and independently, by Leighton.
- The crossing number of any simple graph (i.e., a graph with no multi-edges or no self-loops) with $|E| \geq 4|V|$ is at least $|E|^3/64|V|^2$.
- We know Eulers formula for any spherical polyhedron, with $|V|$ vertices, $|E|$ edges and $|F|$ faces, $|V| - |E| + |F| = 2$.
- Any maximal planar graph (i.e., one to which no edge can be added without losing planarity) has triangular $|F|$ triangular faces implying $3|F| = 2|E|$.

- Kuratowski showed 1930 that a graph is planar if and only if it has no subgraph *homeomorphic* to K_5 or $K_{3,3}$.
- The following *Crossing Number Theorem* was proved by Ajtai, Chvatal, Newborn and Szemeredi in 1982, and independently, by Leighton.
- The crossing number of any simple graph (i.e., a graph with no multi-edges or no self-loops) with $|E| \geq 4|V|$ is at least $|E|^3/64|V|^2$.
- We know Eulers formula for any spherical polyhedron, with $|V|$ vertices, $|E|$ edges and $|F|$ faces, $|V| - |E| + |F| = 2$.
- Any maximal planar graph (i.e., one to which no edge can be added without losing planarity) has triangular $|F|$ triangular faces implying $3|F| = 2|E|$.
- Hence, for any simple planar graph with $|V| = n \geq 3$ vertices, we have $|E| = |V| + |F| - 2 \leq |V| + (2/3)|E| - 2$ or $|E| \leq 3n - 6$, implying that it has at most $3n$ edges.

- Kuratowski showed 1930 that a graph is planar if and only if it has no subgraph *homeomorphic* to K_5 or $K_{3,3}$.
- The following *Crossing Number Theorem* was proved by Ajtai, Chvatal, Newborn and Szemeredi in 1982, and independently, by Leighton.
- The crossing number of any simple graph (i.e., a graph with no multi-edges or no self-loops) with $|E| \geq 4|V|$ is at least $|E|^3/64|V|^2$.
- We know Eulers formula for any spherical polyhedron, with $|V|$ vertices, $|E|$ edges and $|F|$ faces, $|V| - |E| + |F| = 2$.
- Any maximal planar graph (i.e., one to which no edge can be added without losing planarity) has triangular $|F|$ triangular faces implying $3|F| = 2|E|$.
- Hence, for any simple planar graph with $|V| = n \geq 3$ vertices, we have $|E| = |V| + |F| - 2 \leq |V| + (2/3)|E| - 2$ or $|E| \leq 3n - 6$, implying that it has at most $3n$ edges.
- Therefore, the crossing number of any simple graph with n vertices and m edges is at least $m - 3n$.

- Let $G = (V, E)$ be a graph with $|E| \geq 4|V|$ embedded in the plane with $t = cr(G)$ crossings.

- Let $G = (V, E)$ be a graph with $|E| \geq 4|V|$ embedded in the plane with $t = cr(G)$ crossings.
- Let H be the random induced subgraph of G obtained by picking each vertex of G, randomly and independently, to be a vertex of H with probability p (whose value is to be chosen later).

- Let $G = (V, E)$ be a graph with $|E| \geq 4|V|$ embedded in the plane with $t = cr(G)$ crossings.
- Let H be the random induced subgraph of G obtained by picking each vertex of G, randomly and independently, to be a vertex of H with probability p (whose value is to be chosen later).
- Then, the expected number of vertices in H is $p|V|$, the expected number of edges is $p^2|E|$, and the expected number of crossings (in its given embedding) is $p^4 t$.

- Let $G = (V, E)$ be a graph with $|E| \geq 4|V|$ embedded in the plane with $t = cr(G)$ crossings.
- Let H be the random induced subgraph of G obtained by picking each vertex of G, randomly and independently, to be a vertex of H with probability p (whose value is to be chosen later).
- Then, the expected number of vertices in H is $p|V|$, the expected number of edges is $p^2|E|$, and the expected number of crossings (in its given embedding) is $p^4 t$.
- Therefore, we have $p^4 t \geq p^2|E| - 3p|V|$, implying $t \geq |E|/p^2 - 3|V|/p^3$.

- Let $G = (V, E)$ be a graph with $|E| \geq 4|V|$ embedded in the plane with $t = cr(G)$ crossings.
- Let H be the random induced subgraph of G obtained by picking each vertex of G, randomly and independently, to be a vertex of H with probability p (whose value is to be chosen later).
- Then, the expected number of vertices in H is $p|V|$, the expected number of edges is $p^2|E|$, and the expected number of crossings (in its given embedding) is $p^4 t$.
- Therefore, we have $p^4 t \geq p^2|E| - 3p|V|$, implying $t \geq |E|/p^2 - 3|V|/p^3$.
- Substituting $p = 4|V|/|E|$, which is less than one, we get the result.

- Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars, Computational Geometry: Algorithms and Applications (3rd ed.), TELOS, Santa Clara, CA, USA, 2008.
- Jiri Matousek, *Lectures on Discrete Geometry*, Springer.
- Ketan Mulmuley, Computational Geometry: An Introduction Through Randomized Algorithms, Prentice Hall, 1994.
- Janos Pach and Pankaj Agarwal, *Combinatorial Geometry*, Wiley-Interscience Series in Discrete Mathematics and Optimization, 1995.
- B. Chazelle, The discrepancy method: Randomness and complexity, Cambridge University Press, 2000.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, Introduction to algorithms, Second Edition, Prentice-Hall India, 2003.
- Udi Manber, Introduction to algorithms: A creative approach, Addision-Wesley, 1989.

R. Motwani and P. Raghavan, Randomized algorithms, Cambridge University Press, 1995.