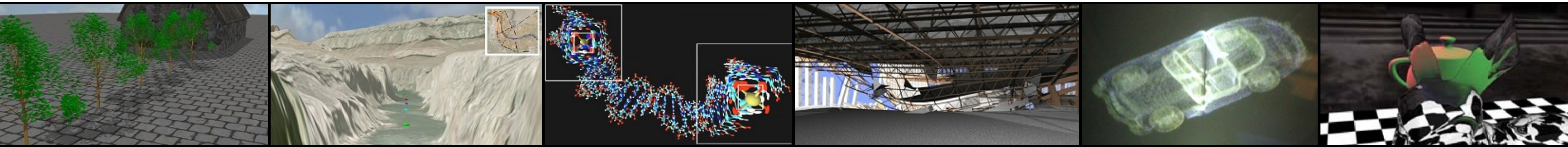
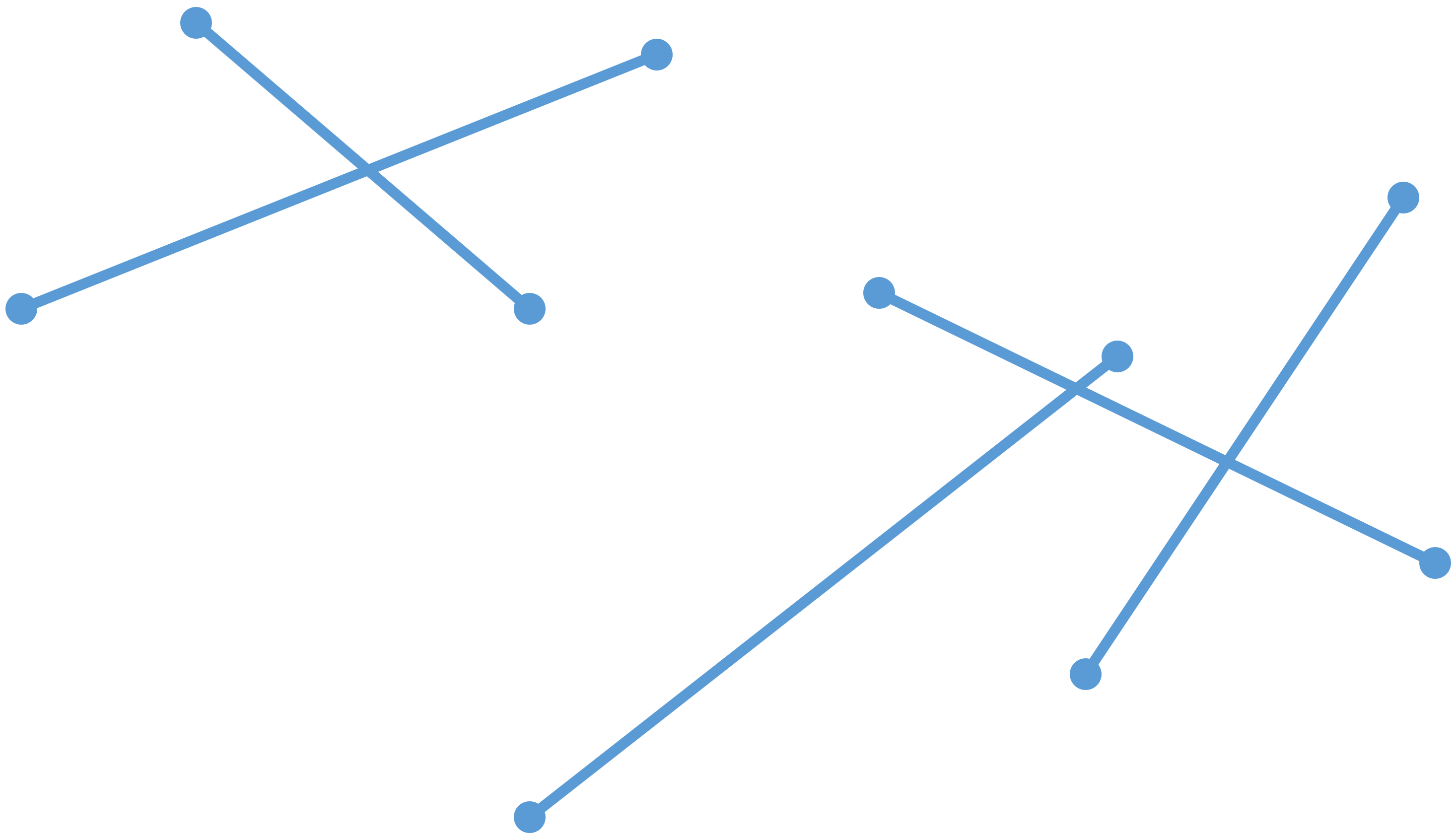
COT 4521: Introduction to Computational Geometry

Segment Intersection AABB Algorithm

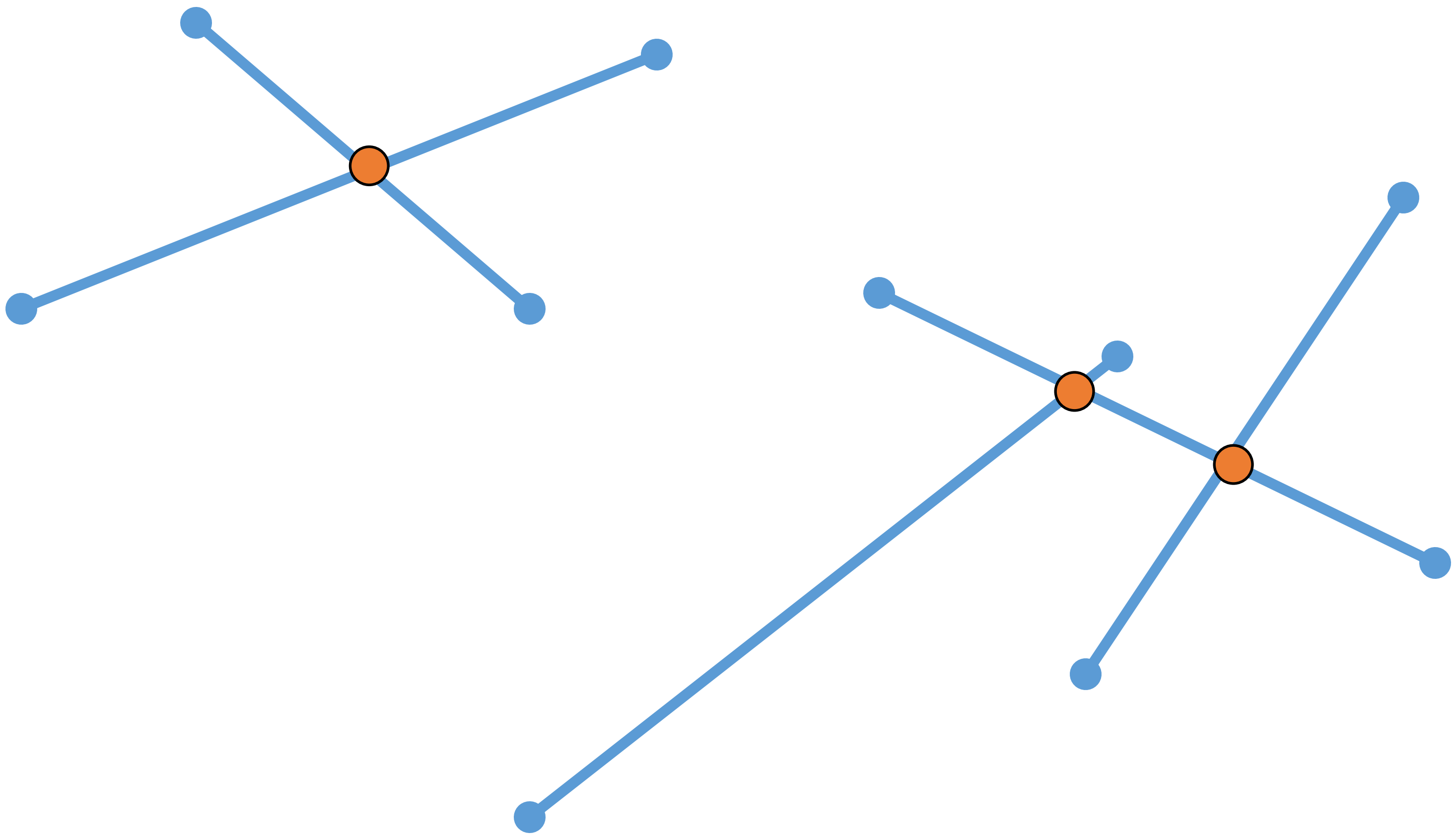
Paul Rosen
Assistant Professor
University of South Florida



Intersection of >2 Line Segments



Intersection of >2 Line Segments



Intersection of line segments

- Problem definition
  - Given N line segments in the plane, report all their points of intersection (pairwise).
- Line Segment Intersection
  - Instance:
    - Set $S = \{s_1, s_2, \dots, s_N\}$ of line segments in the plane;
    - For $1 \leq i \leq N, s_i = (\mathrm{P}_{i1}, \mathrm{P}_{i2})$ (endpoints of the segments); and
    - For $1 \leq j \leq 2, \mathrm{P}_{ij} = \left(x_{ij}, y_{ij}\right)$ (coordinates of the endpoints).
  - Question:
    - Report all points of intersection of segments in S.

INTERSECTION OF LINE SEGMENTS

- ALGORITHM (BRUTE FORCE ALGORITHM)
  - For every pair of segments in S, test the two segments for intersection.
  - (Segment intersection test can be done in constant time using one of the methods we've already discussed.)
- ANALYSIS (PREPROCESSING, QUERY, AND STORAGE COSTS)
  - Preprocessing: None
  - Query: $O(N^2)$; there are $\frac{N(N-1)}{2} = O(N^2)$ pairs, each requiring a constant time test.
  - Storage: $O(N)$; for S.

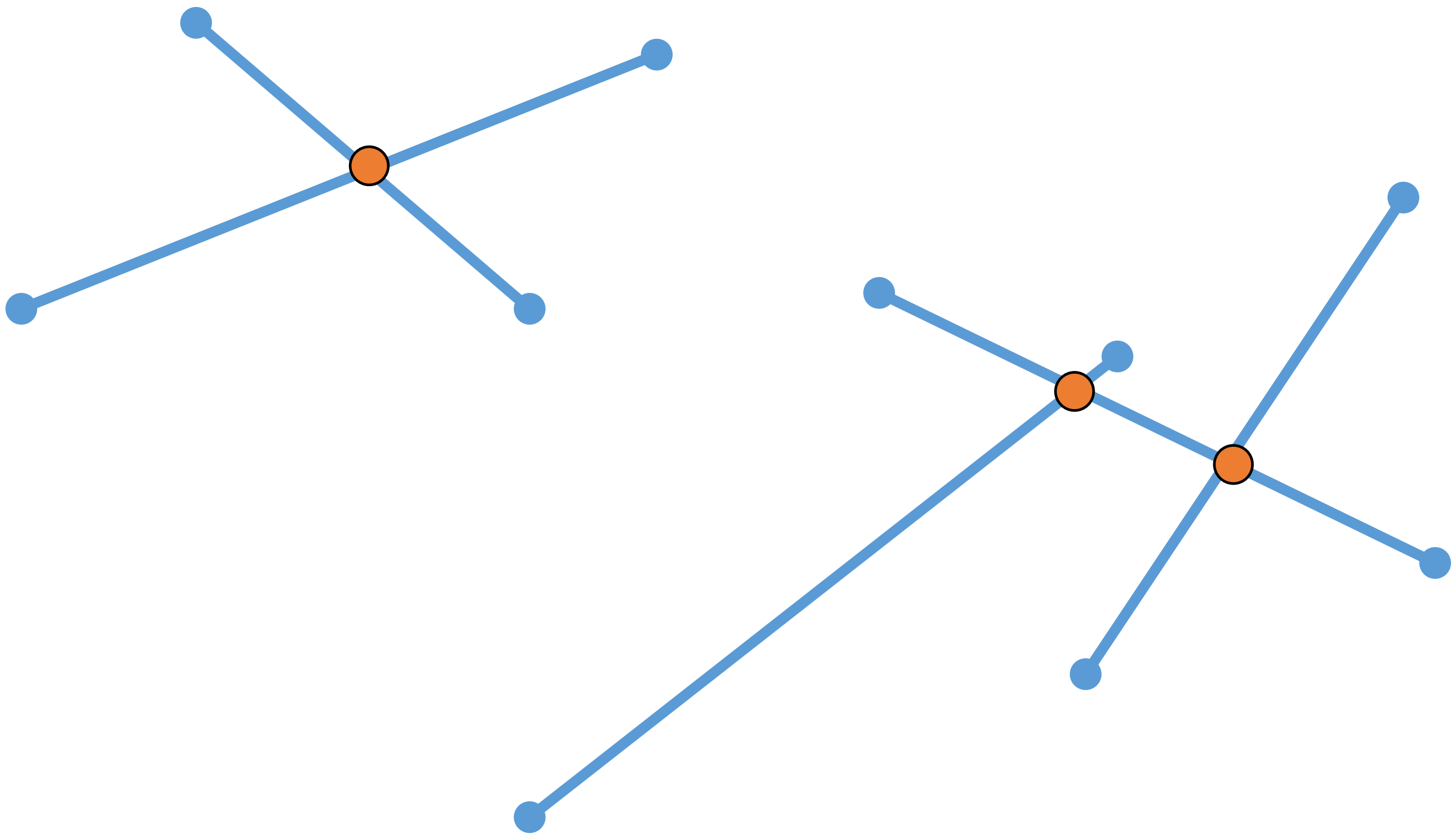
Naïve Intersection of >2 Line Segments

- What is the worst case number of intersections?
  - Choice 1: $O(n)$
  - Choice 2: $O(n \log n)$
  - Choice 3: $O(n^2)$

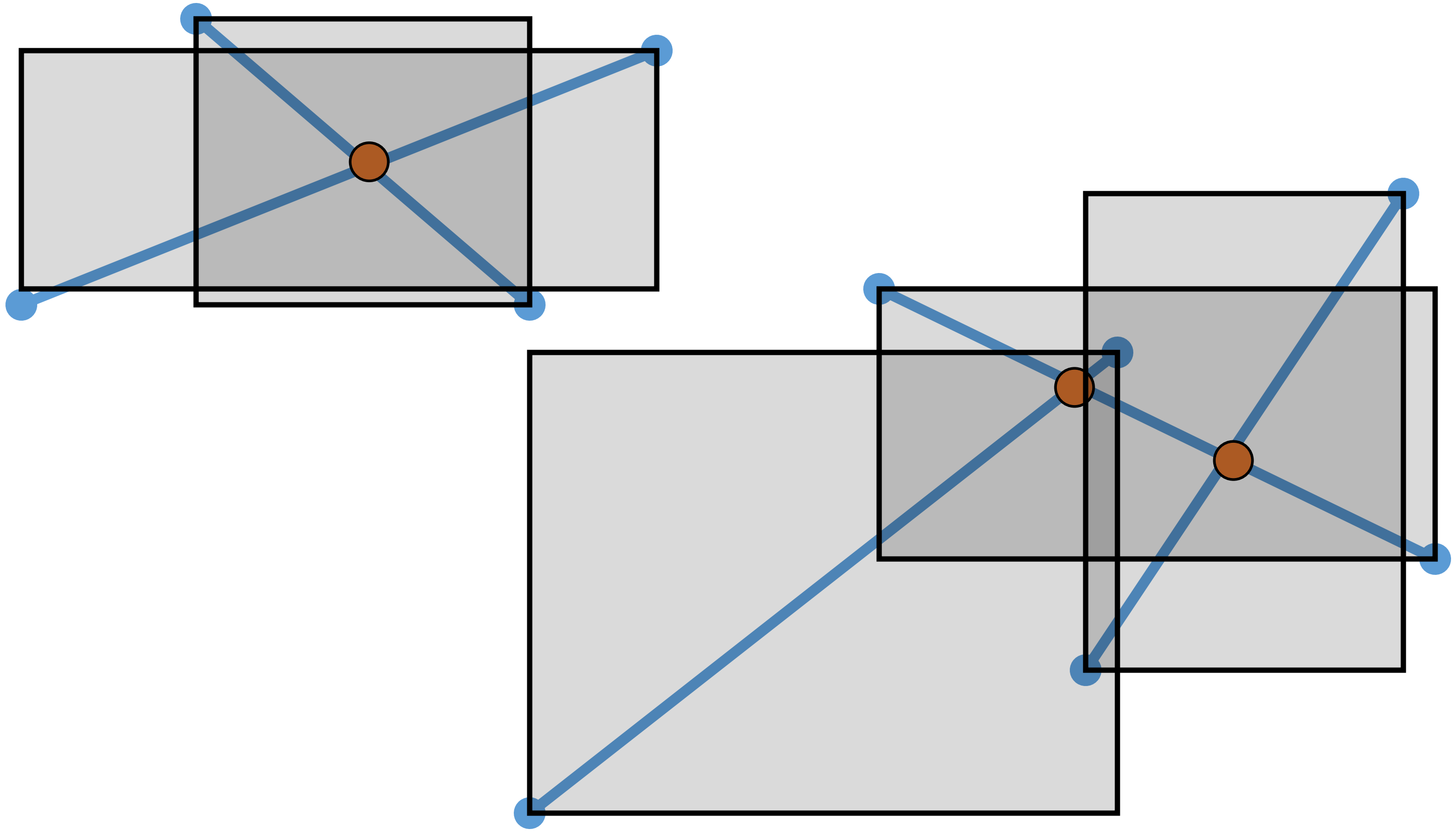
Naïve Intersection of >2 Line Segments

- What is the worst case number of intersections?
  - If all pairs intersect there are (n^2) intersections, then our time bound is optimal as a function of n.
- Can we improve performance?
  - Yes, we will look for output-sensitive algorithms.

Observations?

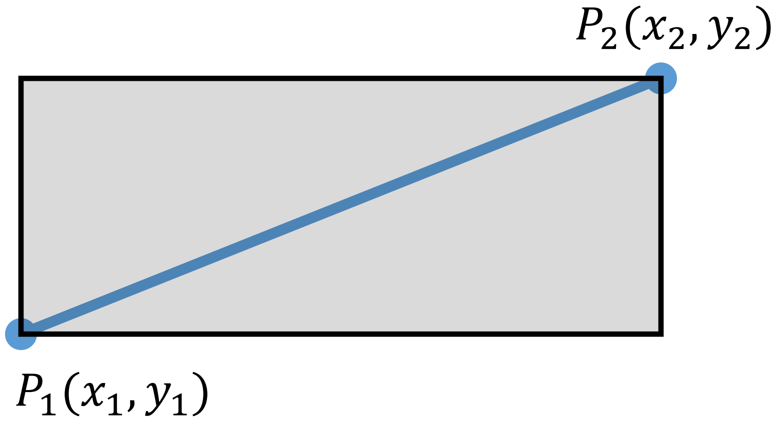


Axis Aligned Bounding Boxes (AABB)



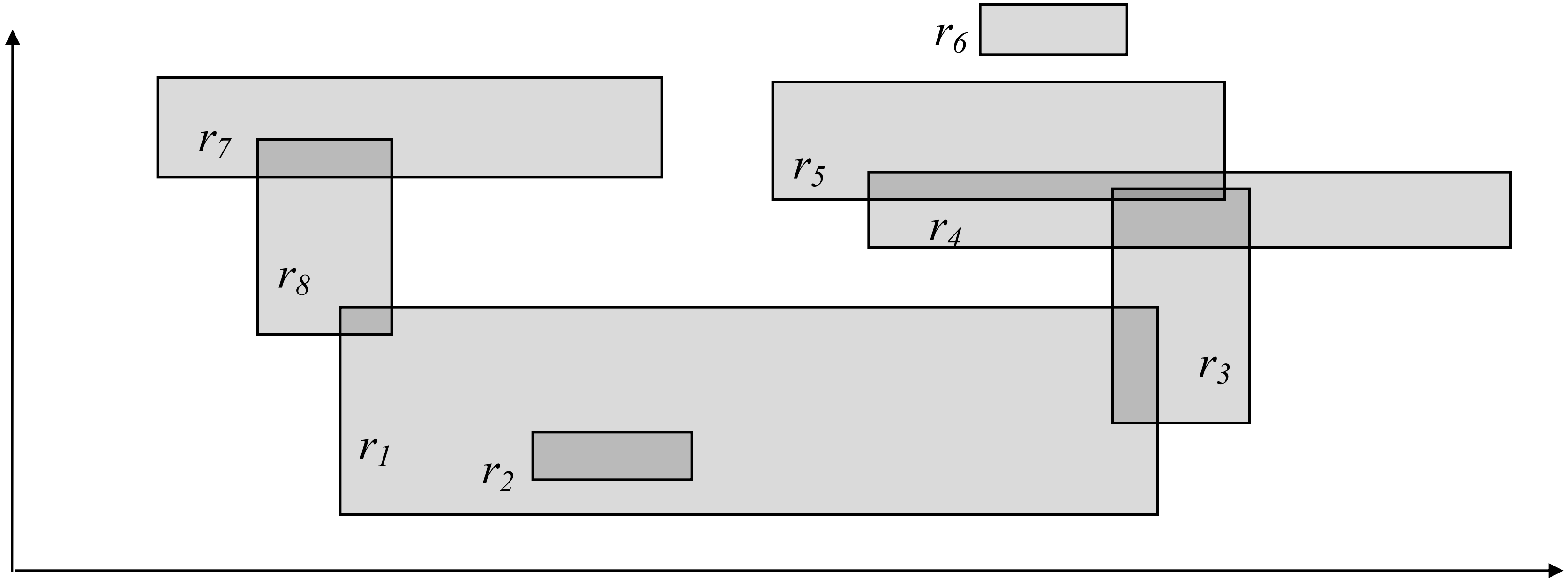
Axis Aligned Bounding Boxes (AABB)

- Many representations
  - 2 Points
  - Point, width, height
  - Intervals
- For our context, we will use intervals
  - $x \in [\min(x_1, x_2), \max(x_1, x_2)]$
  - $y \in [\min(y_1, y_2), \max(y_1, y_2)]$



AABB Intersection

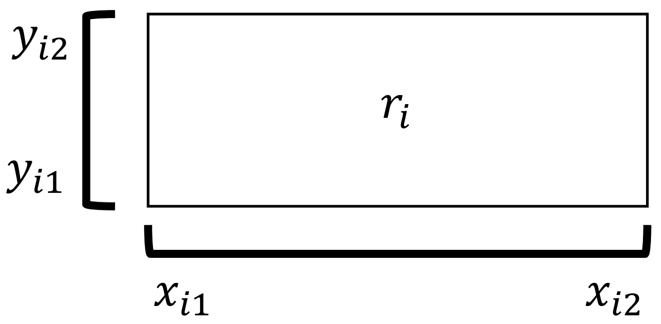
- Given a set of N axis-parallel rectangles in the plane, report all intersecting pairs
  - Intersect ≡ share at least one point

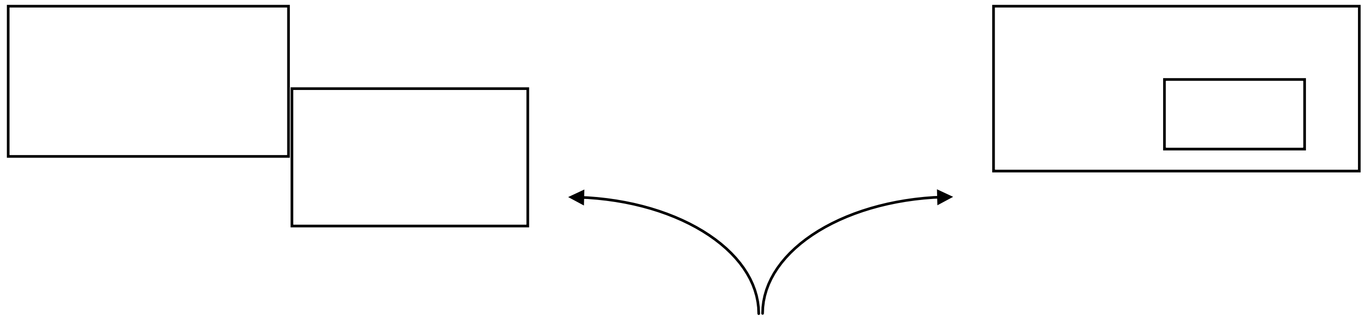


Answer: (r_1, r_2) (r_1, r_3) (r_1, r_8) (r_3, r_4) (r_3, r_5) (r_4, r_5) (r_7, r_8)

INTERSECTION OF RECTANGLES PROBLEM DEFINITION

- RECTANGLE INTERSECTION
- INSTANCE: Set $S = \{r_1, r_2, \ldots, r_N\}$ of rectangles in the plane.
- For $1 \leq i \leq N, r_i = ([x_{i1}, x_{i2}], [y_{i1}, y_{i2}])$
- QUESTION: Report all pairs of rectangles that intersect
  - (Edge and interior intersections should be reported.)

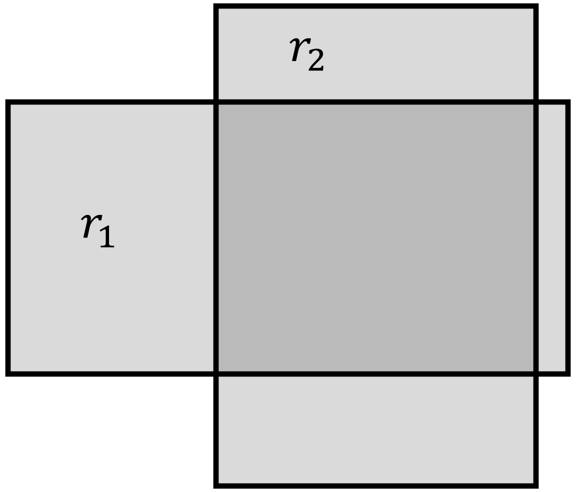




Considered to intersect.

CHECKING IF 2 RECTANGLES INTERSECT

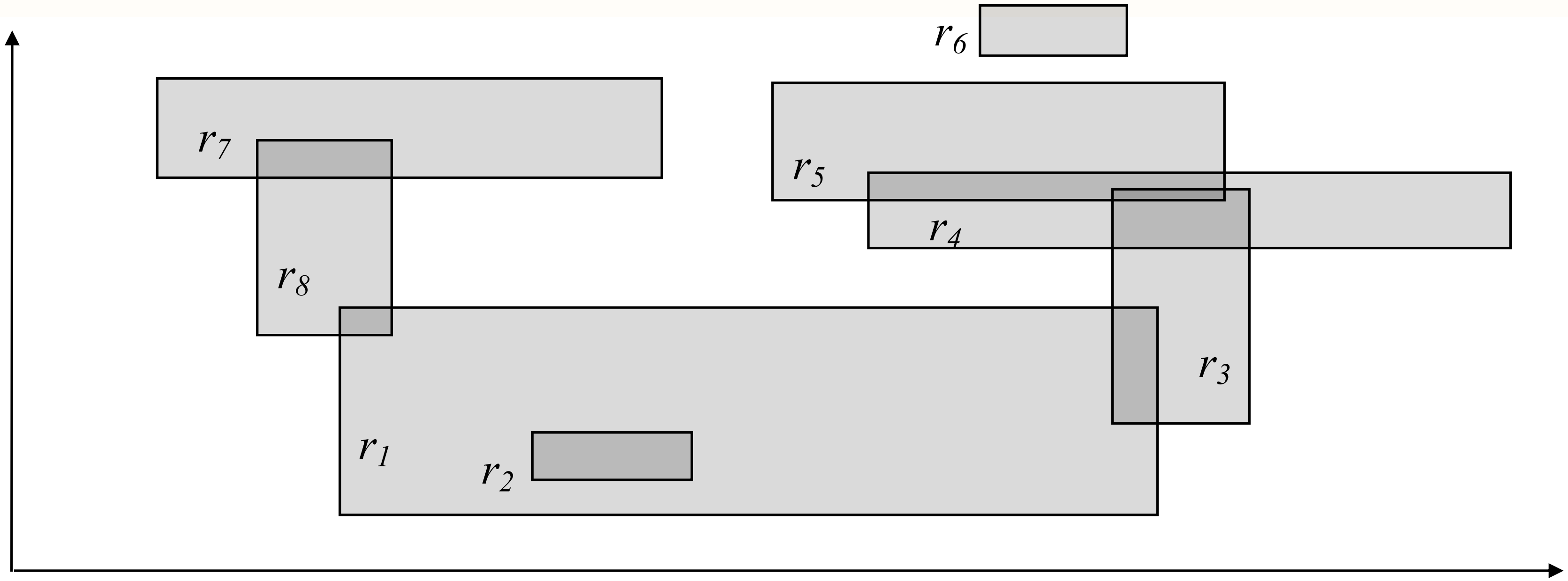
- $r_1 \cap r_2$, IF?
  - $[x_{11}, x_{12}] \cap [x_{21}, x_{22}] \neq \emptyset$ AND $[y_{11}, y_{12}] \cap [y_{21}, y_{22}] \neq \emptyset$
- HOW DO WE CODE THIS INTERSECTION?
  - $[x_{11}, x_{12}] \cap [x_{21}, x_{22}] = [\max(x_{11}, x_{21}), \min(x_{12}, x_{22})]$
  - $[y_{11}, y_{12}] \cap [y_{21}, y_{22}] = [\max(\mathrm{y}_{11}, y_{21}), \min(y_{12}, y_{22})]$
  - Check that the range of both intersections is ≥ 0



Intersection of a set of rectangles

```
Brute force algorithm
```

1. for every pair (r_i, r_j) of rectangles $\in S, i < j$
2. if $(r_i \cap r_j \neq \emptyset)$ then
3. report (r_i, r_j)



Answer: (r1, r2) (r1, r3) (r1, r8) (r3, r4) (r3, r5) (r4, r5) (r7, r8)

Intersection of rectangles

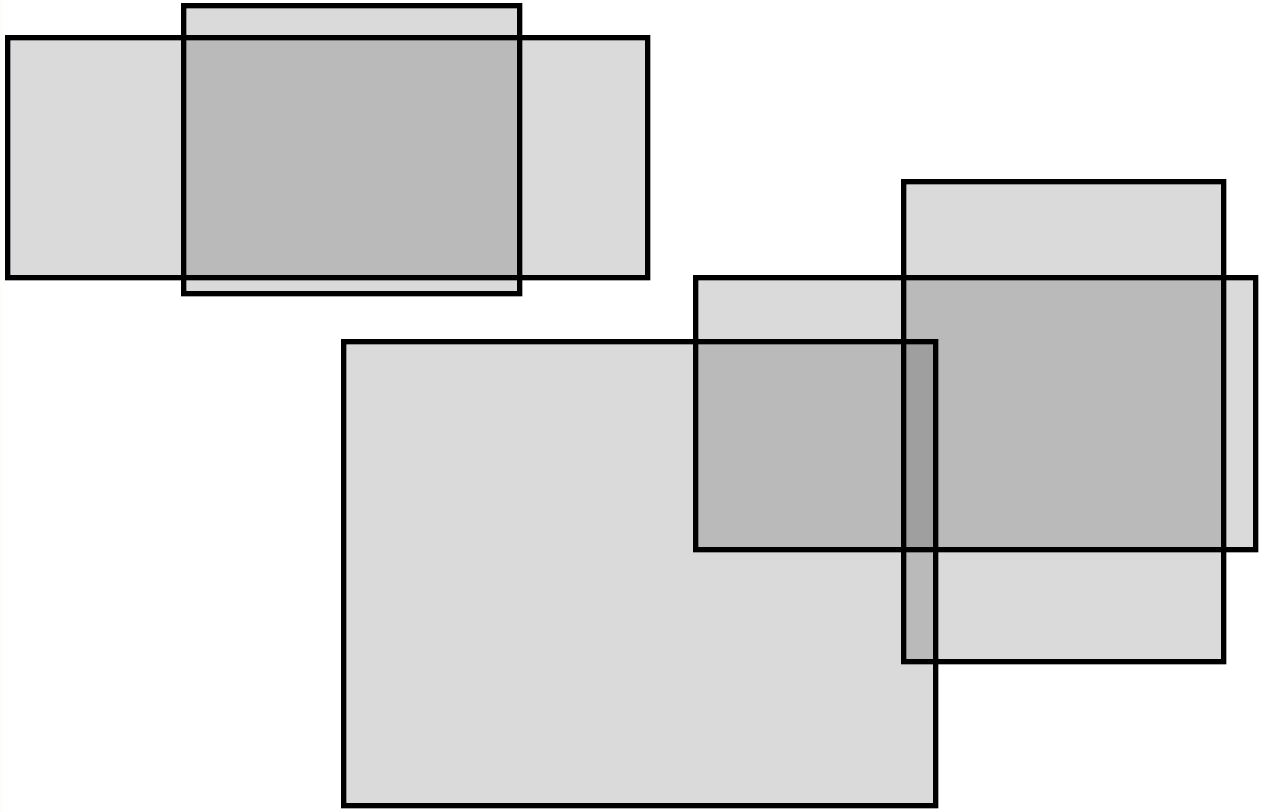
- Analysis
  - Preprocessing?
    - None
  - Query?
    - $\frac{N(N-1)}{2} = O(N^2)$
  - Storage?
    - $O(N)$

- Does this really help us with the Segment intersection problem?

Intersection of rectangles

Algorithm using interval trees

1. Plane sweep algorithm, vertical (top-to-bottom) sweep Event points are Beginning and end of rectangle intervals
2. At the starting interval:
3. Compare rectangle x-interval to active set for overlap.
4. Add rectangle x-interval to active set.
5. At the ending interval remove rectangle X-interval from active set.

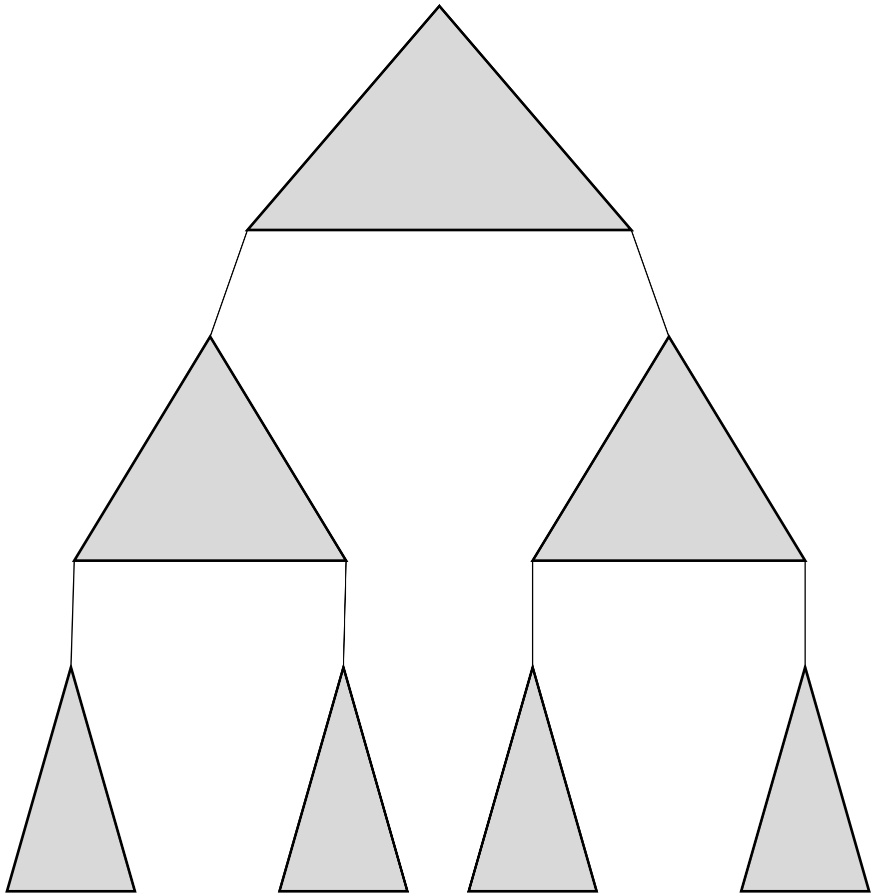


Intersection of rectangles

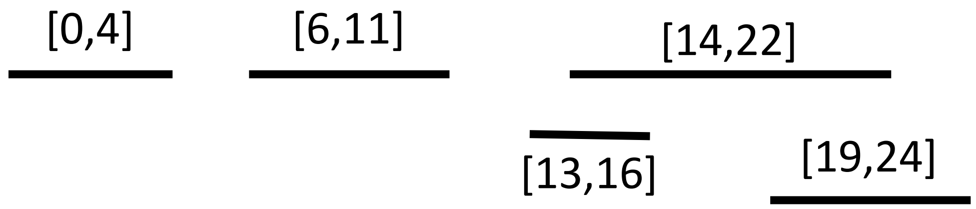
- Analysis
  - Preprocessing: $O(N \log N)$; ordering intervals for sweep
  - Query: Worst case $O(N^2)$; Best case $O(N)$
  - Storage: $O(N)$; active rectangles $O(N)$, event queue $O(N)$.
- Comments
  - We haven't improved worst case, but the best case has gotten significantly better.
  - We are now output sensitive.
  - Can we do any better?

1D Centered Interval Trees

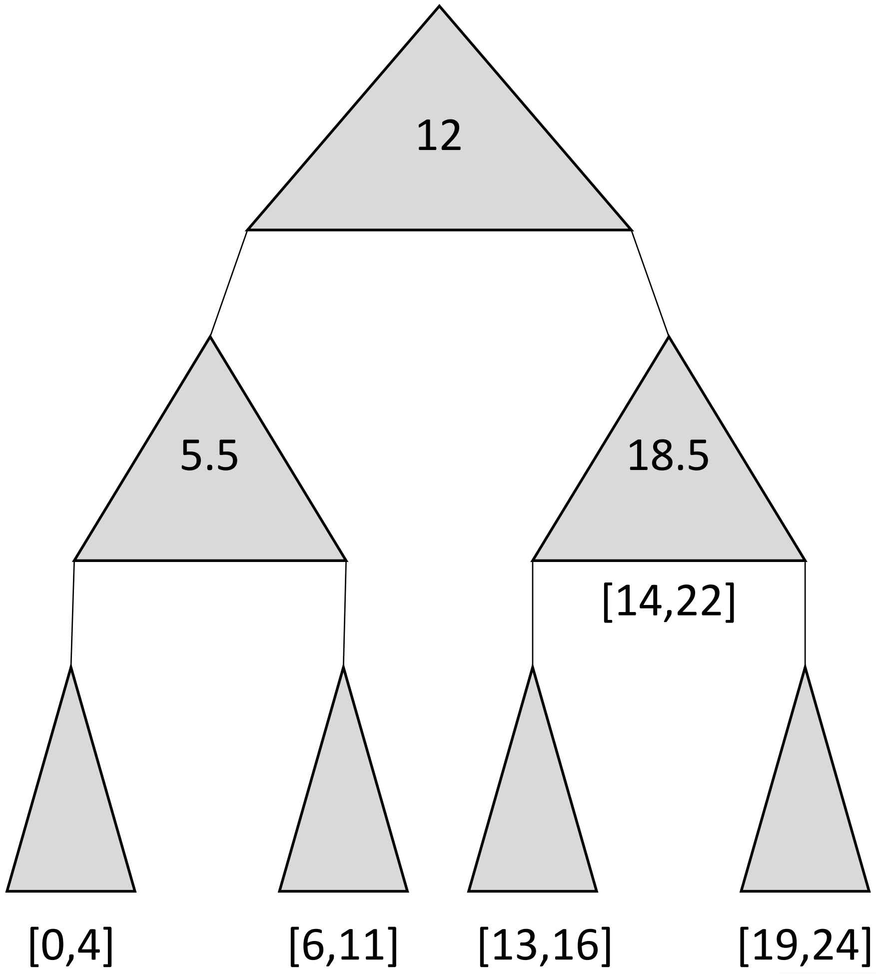
- Tree-based Data structure
- Each node stores a center point
  - Intervals are placed into 3 group,
    - Left of center—placed in left subtree
    - Right of center—placed in right subtree
    - Covering center—placed in a specialized list*



1D Centered Interval Trees



- Performance Analysis
  - Insertion/removal: $O(\log N)$
  - Query: $O(\log N + K)$
  - Storage: $O(N)$



INTERSECTION OF RECTANGLES

- ANALYSIS
  - Preprocessing: $O(N \log N)$; ordering intervals for sweep
  - Query: $O(N \log N + K)$
  - Storage: $O(N)$; interval tree $O(N)$, event queue $O(N)$.
- COMMENTS
  - $O(N \log N + K)$ is lower bound for rectangle intersection problem. Can be shown by lower bounds proof.
  - We've gone to a lot of trouble to improve the time from $O(N^2)$ to $O(N \log N)$ via the interval tree, for good reason.
    - E.g. if $N = 10^6$, $N^2 = 10^{12}$ and $N \log N = 2 \cdot 10^7$

Back to the Segment Intersection Problem

- Can we do better than reducing it to the AABB intersection problem?

- Yes and no, can't do better than $O(N \log N + K)$, but can improve constants