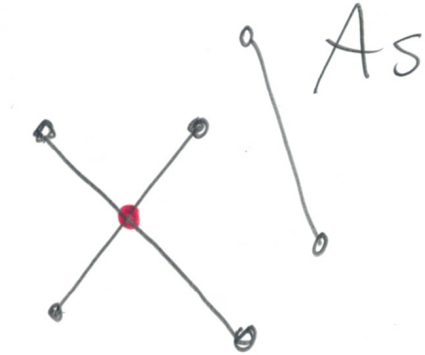
Line Segment Intersection
Balloon Pop Problem
Seg Trees

Input: n line segments (2D)

Goal: find all the intersections.

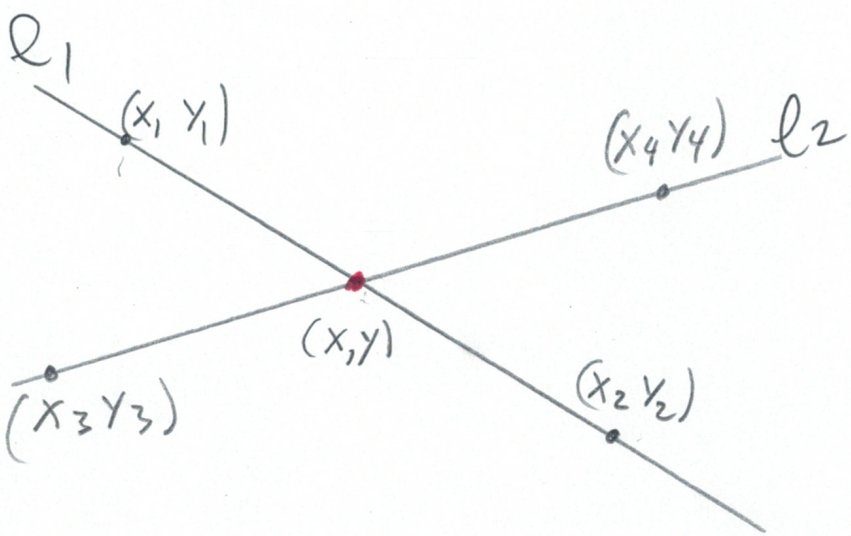
Assumptions:
* no vertical segments
* no 3-way intersections
* no segment ends on another



Trivial $O(n^2)$ Algorithm

Goal: $O((n+k)\log n)$

k ← #intersections.



$$X = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & x_1 - x_2 \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & x_3 - x_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & y_1 - y_2 \\ x_3 - x_4 & y_3 - y_4 \end{vmatrix}}$$

$$Y = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & y_1 - y_2 \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & y_3 - y_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & y_1 - y_2 \\ x_3 - x_4 & y_3 - y_4 \end{vmatrix}}$$

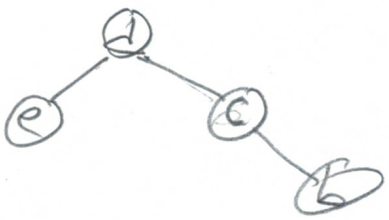
$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

= 2x2 determinant

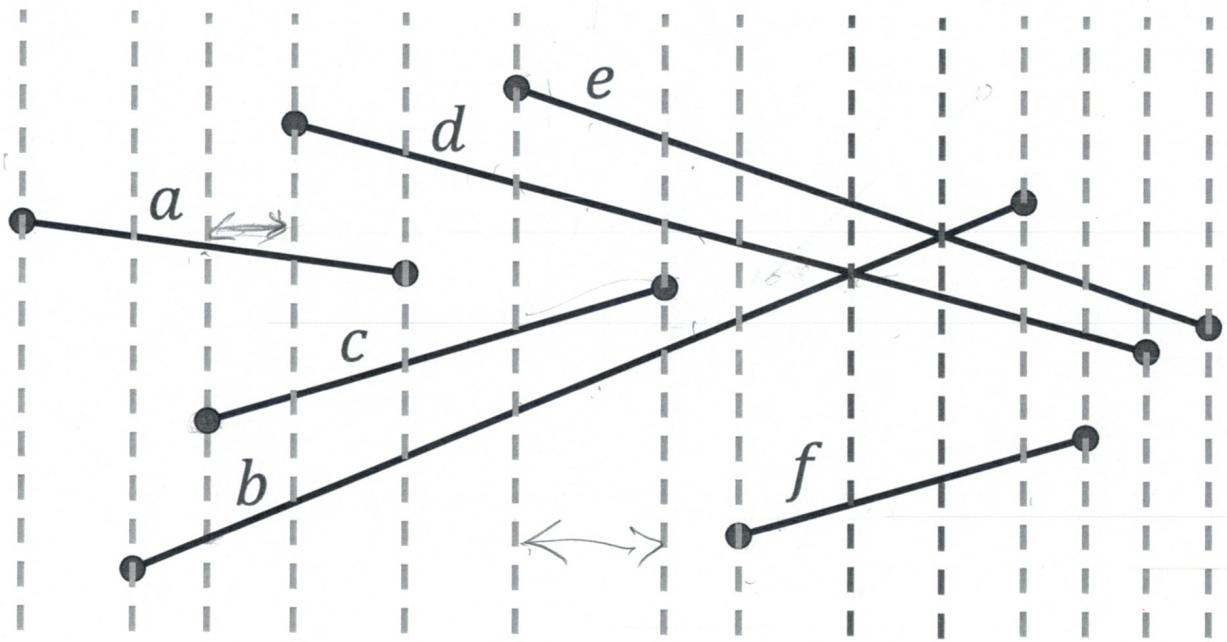
= Cross Product

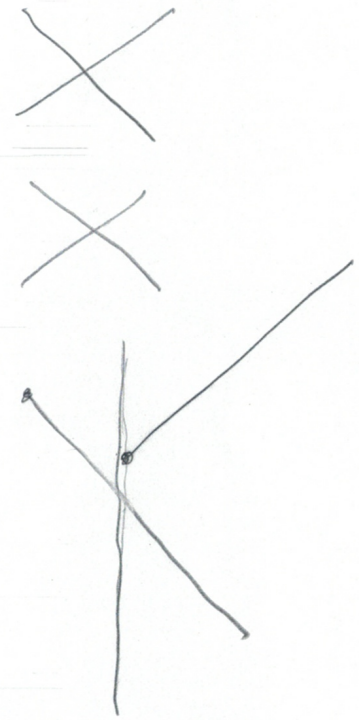
"Interesting" x-coordinates
initially the end points

Two Data Structures: SL (Segment List)
EQ (Event Queue)



					e	e	e	e	b
			d	d	d	e	d	b	e
		a	a	a	c	d	b	d	d
	a	c	c	c	b	b	f	f	f
a	b	b	b	b	b	b			





SL Requirements

Insert new segment
Delete a segment
find successor
find predecessor

$O(\log n)$

Balanced Search tree.

with Key:

(m_i, b_i)

intercept

slope

$y = mx + b$

EQ Needs:

Insert
findmin
deletemin

$O(\log n)$

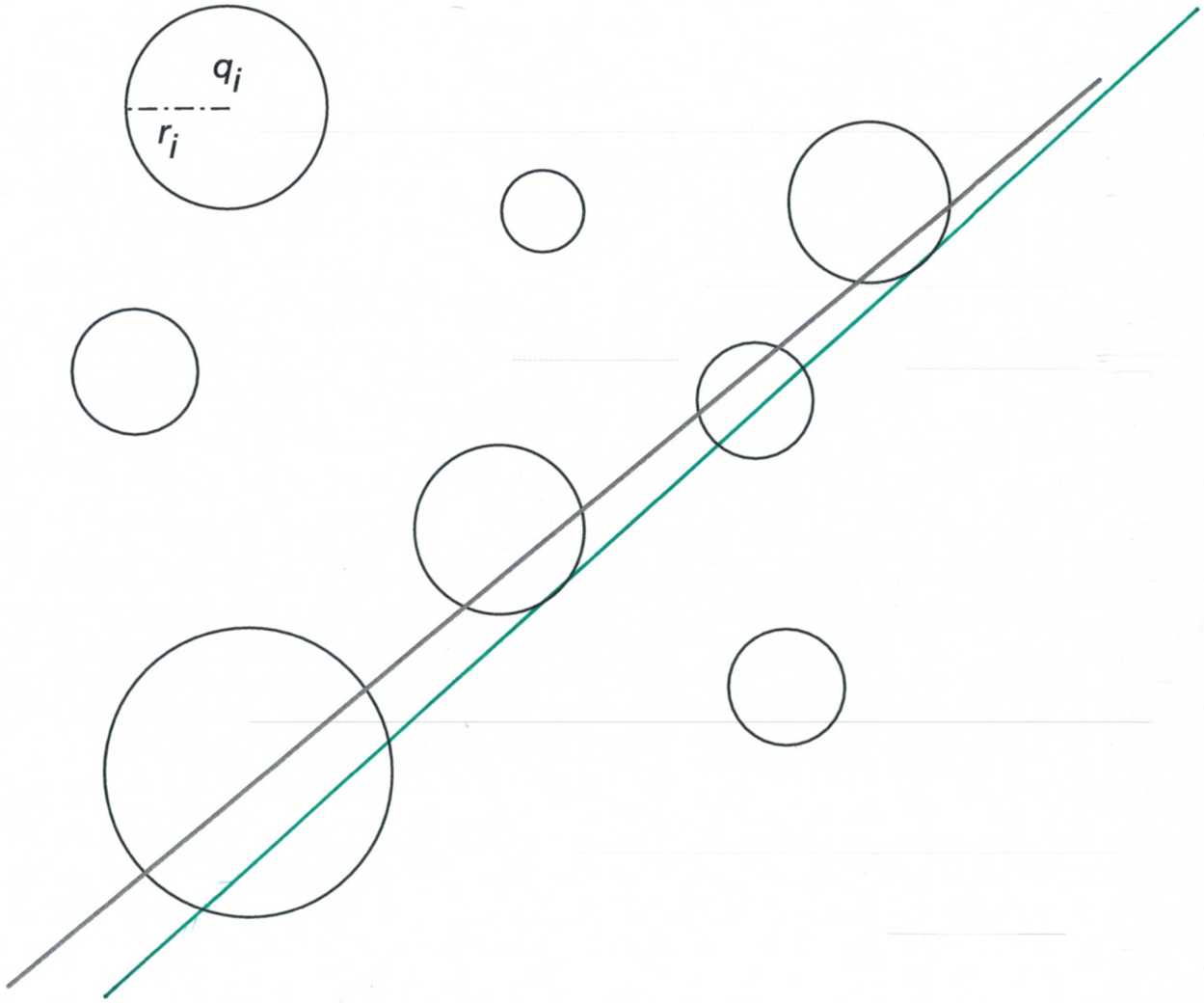
$m_i x + b_i$

```
def FindSegmentIntersections(S):
    create an empty priority queue EQ
    for each segment S, insert its start and end events into EQ
    create an empty balanced tree SL
    while (EQ is not empty)
        E := deletemin(EQ)
        if E is start of segment s then
            insert s into SL
            CheckForIntersect(EQ, s, successor of s)
            CheckForIntersect(EQ, s, predecessor of s)
        else if E is the end of segment s then
            CheckForIntersection(EQ, pred of s, succ of s)
            delete s from SL
        else if E is cross event for segments s1 and s2 then
            println "s1 and s2 intersect"
            remove s1 and s2 from SL
            re-insert s1 and s2 into SL in the opposite order
            CheckForIntersection(EQ, s1, new neighbor of s1)
            CheckForIntersection(EQ, s2, new neighbor of s2)
```

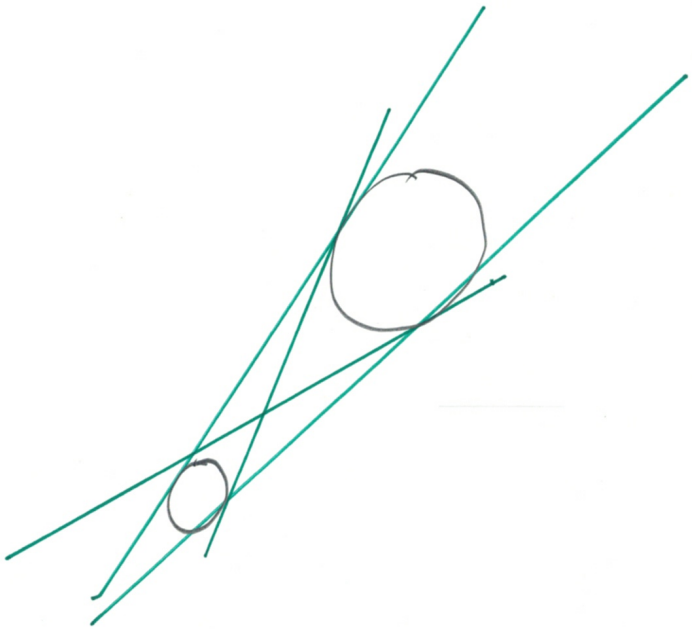
Balloon Pop

Input n circles
non intersecting

Goal: compute the line that intersects the most circles.



WLOG: a solution is tangent to two circles



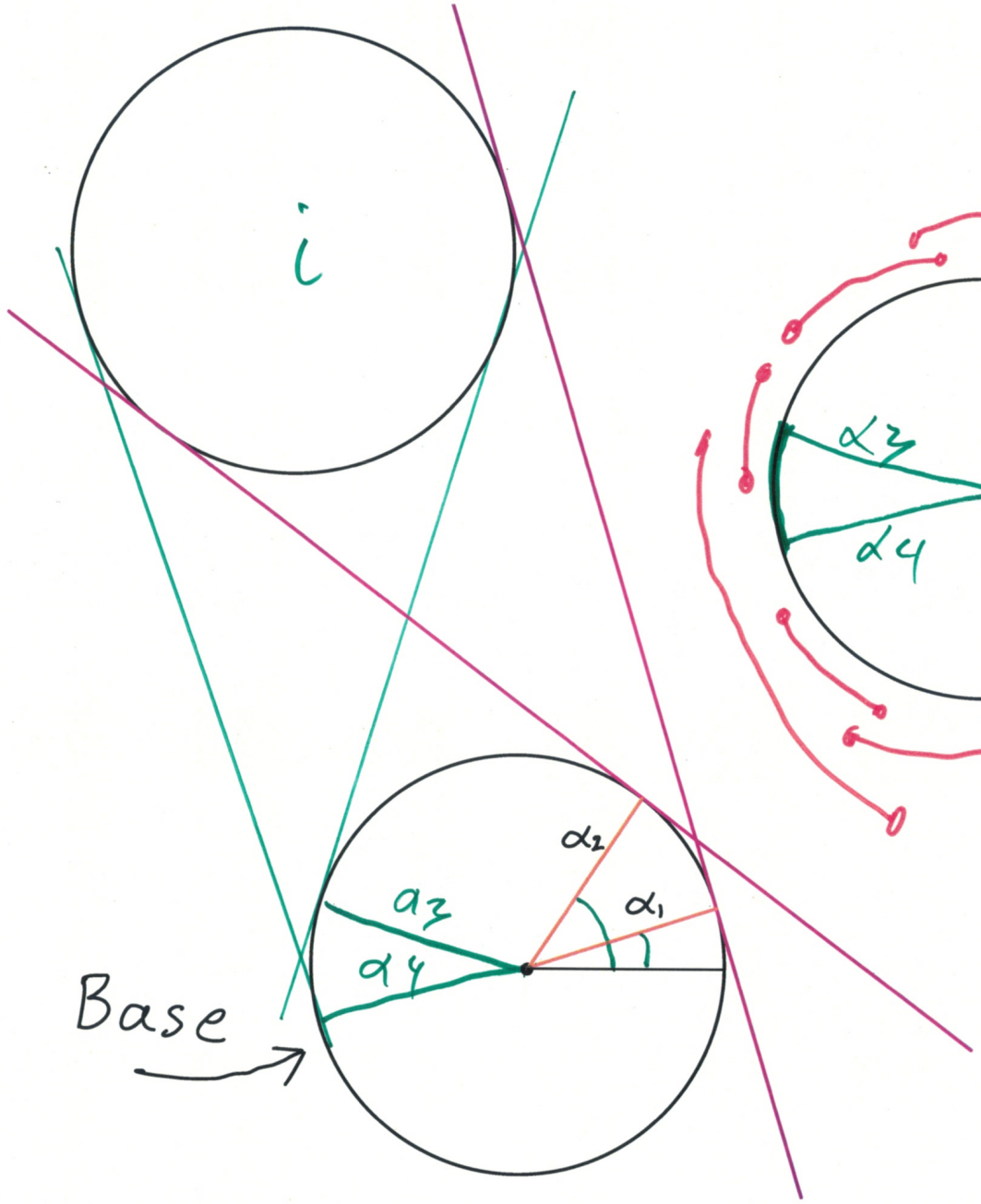
$4n^2$ lines

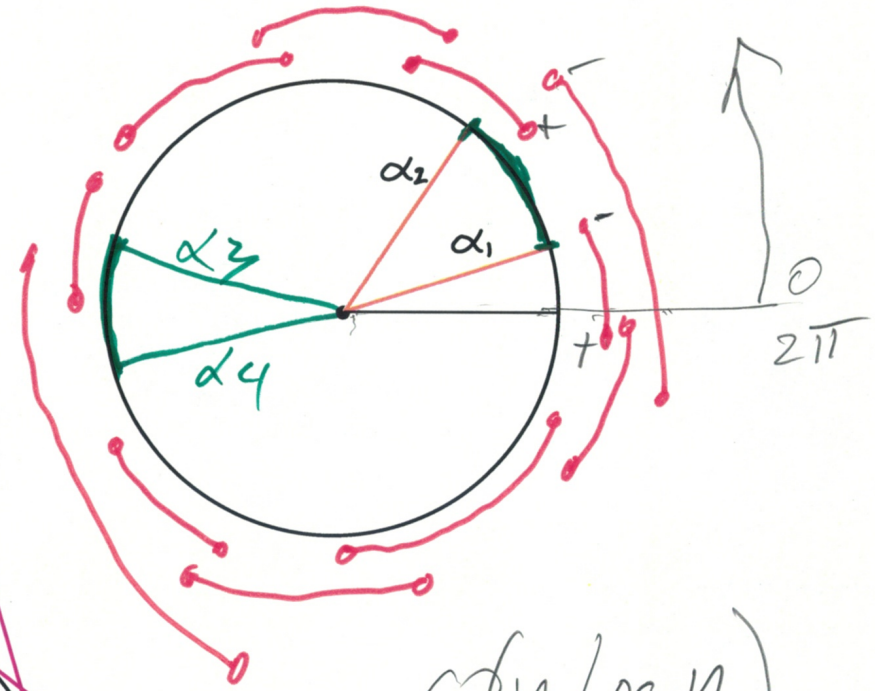
try the $n-2$ others

$O(n^3)$

$O(n^2 \log n)$

sweep angle algorithm


i
α_2
α_1
α_3
α_4
Base


α_2
α_1
α_3
α_4
+
-
+
-
0
2π

$O(n \log n)$

SegTree : Arrays with Superpowers

maintain variables $a[0] \dots a[n-1]$

support:

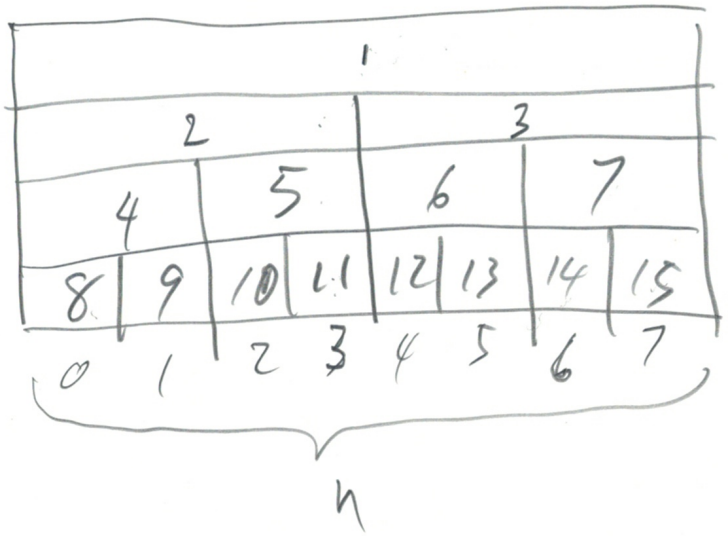
$O(\log n)$

Assign(i, x) : $a[i] \leftarrow x$

Rangesum(i, j) : $\sum_{i \le k \le j} a[k]$

$1 + \log n$ levels

[1, 6]



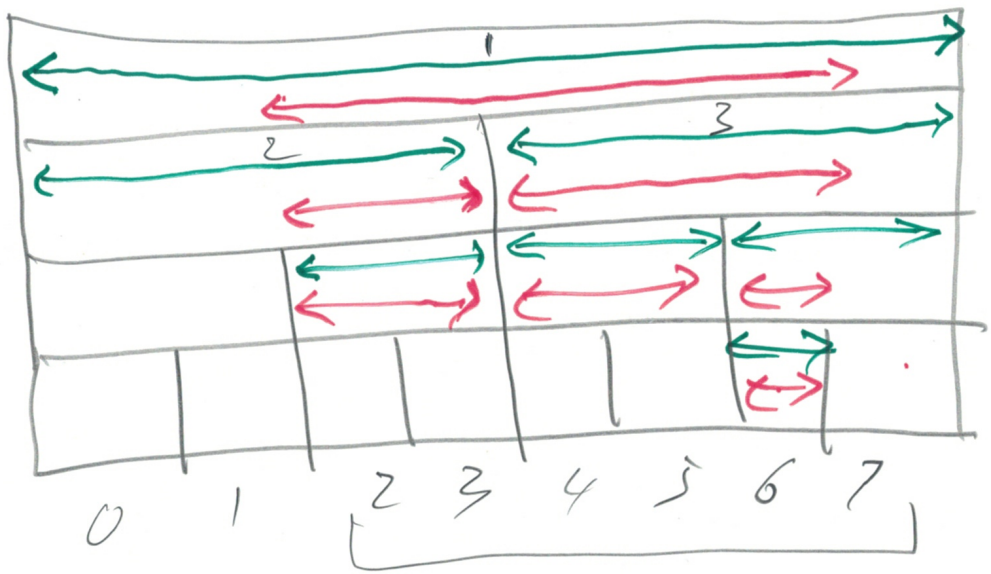
$\text{parent}(i) = \lfloor \frac{i}{2} \rfloor$

$\text{left}(i) = 2i$

$\text{right}(i) = 2i+1$

$N = \lceil \lceil n \rceil \rceil$

Allocate an array of size $2N$ $A[\,]$



The array $A[0] \cdots A[2N-1]$

```
Assign(i, x) {
    i = i + N;
    A[i] ← x;
    for (i = Parent(i); i > 0; i = Parent(i)) {
        A[i] ← A[Left(i)] + A[Right(i)];
    }
}
```