COMP 355
Advanced Algorithms

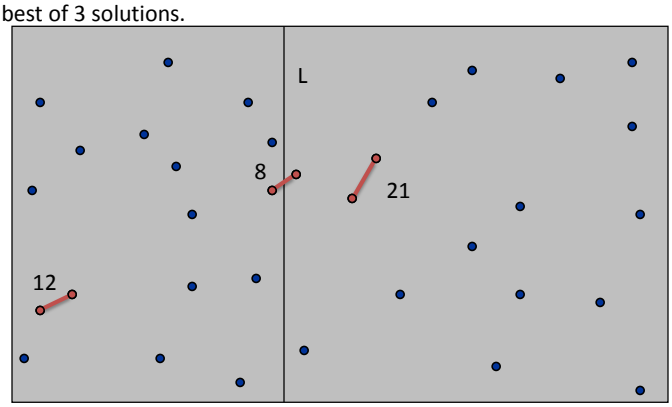
Divide and Conquer: Closest Pair
KT: 5.4

Rhodes College

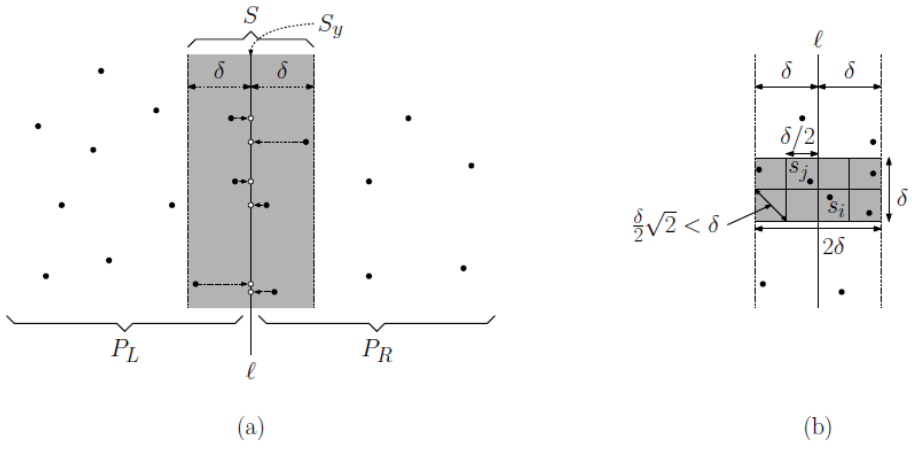
Closest Pair of Points

Algorithm.

- Divide: draw vertical line L so that roughly ½n points on each side.
- Conquer: find closest pair on each side recursively.
- Combine: find closest pair with one point on each side of L.
- Return best of 3 solutions.



Closest Pair of Points



Finding the closest pair in S.

Closest Pair of Points

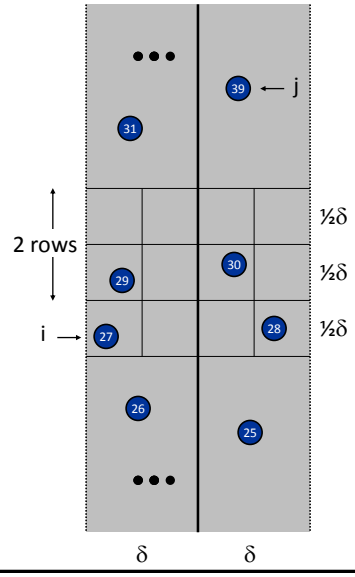
Def. Let s_i be the point in the 2δ-strip, with the i^{th} smallest y-coordinate.

Claim. If $|i - j| \geq 12$, then the distance between s_i and s_j is at least δ.

Pf.

- No two points lie in same ½δ-by-½δ box.
- Two points at least 2 rows apart have distance ≥ 2(½δ). ▪

Fact. Still true if we replace 12 with 7.



Closest Pairs Algorithm

Closest Pair in P

```
closestPair(P = (Px, Py)) {
    n = |P|
    if (n <= 3) solve by brute force                // basis case
    else {
        Find the vertical line L through P's median // divide      O(n log n)
        Split P into PL and PR (split Px and Py as well)
        dL = closestPair(PL)                        // conquer     2T(n / 2)
        dR = closestPair(PR)
        d = min(dL, dR)
        for (i = 1 to n) {                          // create Sy   O(n)
            if (Py[i] is within distance d of L) {
                append Py[i] to Sy
            }
        }
        d' = stripClosest(Sy)                       // closest in strip  O(n)
        return min(d, d')                           // overall closest
    }
}

stripClosest(Sy) {                                  // closest in strip
    m = |Sy|
    d' = infinity
    for (i = 1 to m) {
        for (j = i+1 to min(m, i+7)) {              // search neighbors  O(n)
            if (dist(Sy[i], Sy[j]) <= d') {
                d' = dist(Sy[i], Sy[j])             // new closest found
            }
        }
    }
    return d'
}
```

Total Runtime = 2T(n/2) + n = O(n log n)

5