CSE 417
Algorithms and Complexity

Autumn 2020
Lecture 17
Divide and Conquer Algorithms

Announcements

- Homework 5, Due Friday
  - But will be accepted without penalty through Monday
- Homework 6, Due Wednesday, Nov 18
  - No class Wednesday, Nov 11

Wednesday Results

- Strassen's Algorithm for Matrix Multiplication
  - Recursive, $T(n) = 7\ T(n/2) + cn^2$
  - Runtime: $O(7^{\log n}) = O(n^{\log 7})$ which is about $O(n^{2.807})$
- Counting Inversions
  - Divide and conquer algorithm based on merge sort
  - $O(n \log n)$

Computing the Median

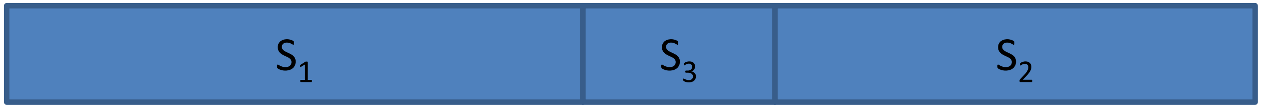
- Given n numbers, find the number of rank n/2
- One approach is sorting
  - Sort the elements, and choose the middle one
  - Can you do better?

Problem generalization

- *Selection*, given n numbers and an integer k, find the k-th largest

Select(A, k)

Select(A, k){
    Choose element x from A
    $S_1 = \{y \text{ in } A \mid y < x\}$
    $S_2 = \{y \text{ in } A \mid y > x\}$
    $S_3 = \{y \text{ in } A \mid y = x\}$
    if ($|S_2| >= k$)
        return Select(S_2, k)
    else if ($|S_2| + |S_3| >= k$)
        return x
    else
        return Select(S_1, $k - |S_2| - |S_3|$)
}



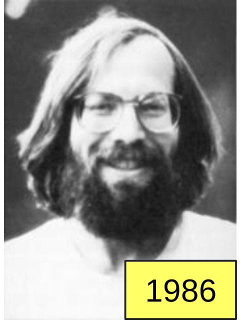
Randomized Selection

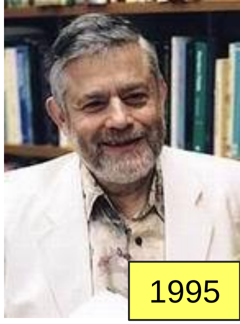
- Choose the element at random
- Analysis can show that the algorithm has expected run time O(n)

Deterministic Selection

- What is the run time of select if we can guarantee that choose finds an x such that $|S_1| < 3n/4$ and $|S_2| < 3n/4$ in $O(n)$ time

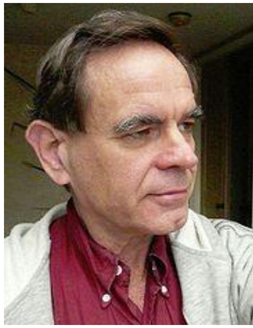
BFPRT Algorithm





- A very clever choose algorithm . . .

Split into n/5 sets of size 5
M be the set of medians of these sets
Let x be the median of M



BFPRT runtime

$|S_1| < 3n/4$, $|S_2| < 3n/4$

Split into n/5 sets of size 5
M be the set of medians of these sets
x be the median of M
Construct S_1 and S_2
Recursive call in S_1 or S_2

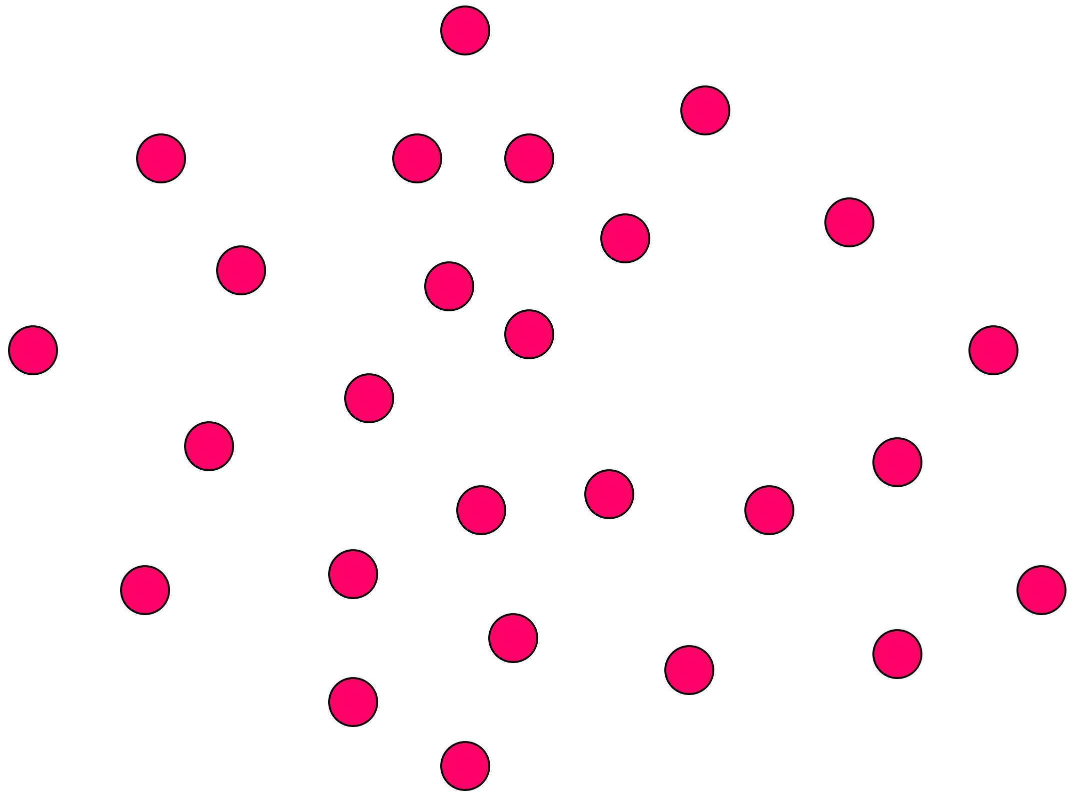
BFPRT Recurrence

- $T(n) <= T(3n/4) + T(n/5) + c\,n$

Prove that $T(n) <= 20\,c\,n$

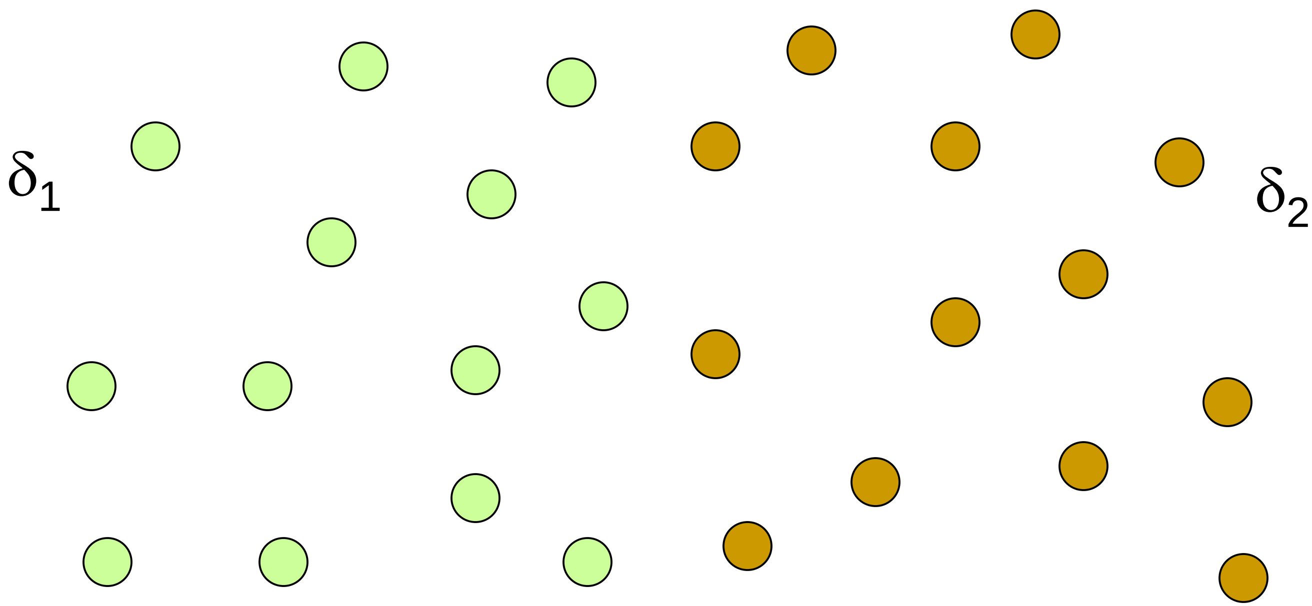
Closest Pair Problem (2D)

- Given a set of points find the pair of points p, q that minimizes dist(p, q)



Divide and conquer

- If we solve the problem on two subsets, does it help? (Separate by median x coordinate)



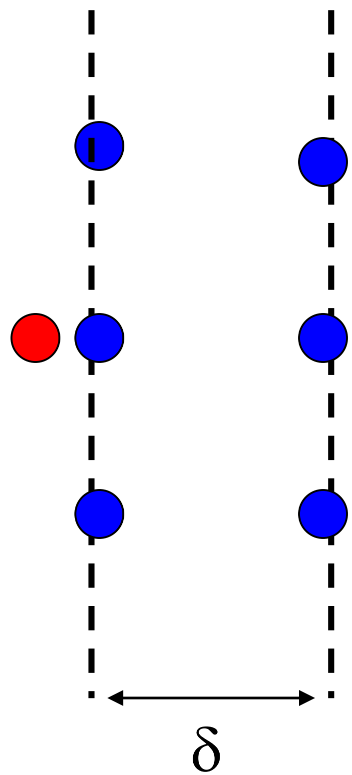
Packing Lemma

Suppose that the minimum distance between points is at least δ, what is the maximum number of points that can be packed in a ball of radius δ?

Combining Solutions

- Suppose the minimum separation from the sub problems is δ
- In looking for cross set closest pairs, we only need to consider points with δ of the boundary
- How many cross border interactions do we need to test?

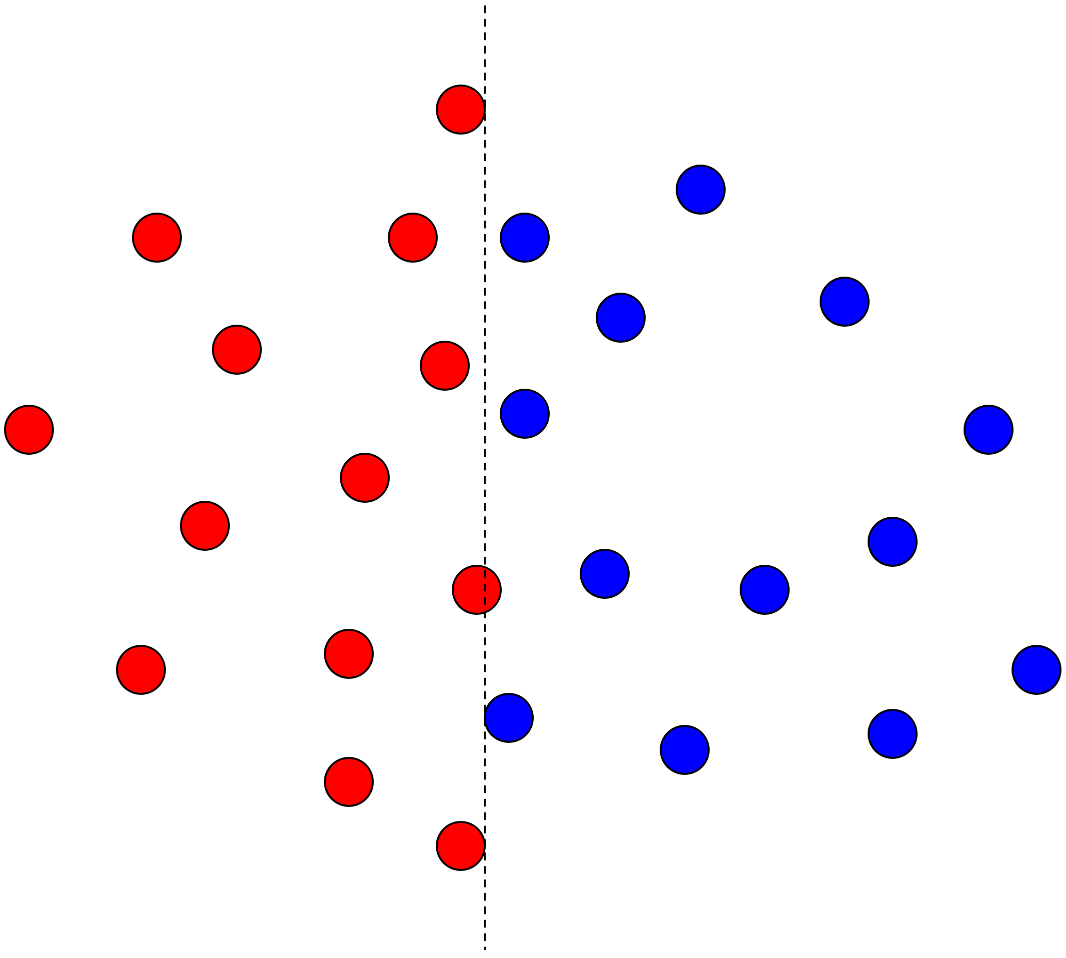
A packing lemma bounds the number of distances to check



Details

- Preprocessing: sort points by y
- Merge step
  - Select points in boundary zone
  - For each point in the boundary
    - Find highest point on the other side that is at most δ above
    - Find lowest point on the other side that is at most δ below
    - Compare with the points in this interval (there are at most 6)

Identify the pairs of points that are compared in the merge step following the recursive calls



Algorithm run time

- After preprocessing:
  - $T(n) = cn + 2\,T(n/2)$

Integer Arithmetic

$$
\begin{array}{r}
9715480283945084383094856701043643845790217965702956767 \\
+\quad 1242431098234099057329075097179898430928779579277597977 \\
\hline
\end{array}
$$

Runtime for standard algorithm to add two n digit numbers:

$$
\begin{array}{r}
2095067093034680994318596846868779409766717133476767930 \\
\times\quad 5920175091777634709677679342929097012308956679993010921 \\
\hline
\end{array}
$$

Runtime for standard algorithm to multiply two n digit numbers:

Recursive Multiplication Algorithm (First attempt)

$x = x_1 2^{n/2} + x_0$

$y = y_1 2^{n/2} + y_0$

$xy = (x_1 2^{n/2} + x_0) (y_1 2^{n/2} + y_0)$

$\quad = x_1 y_1 2^n + (x_1 y_0 + x_0 y_1) 2^{n/2} + x_0 y_0$

Recurrence:

Run time:

Simple algebra

$x = x_1 2^{n/2} + x_0$

$y = y_1 2^{n/2} + y_0$

$xy = x_1y_1 2^n + (x_1y_0 + x_0y_1) 2^{n/2} + x_0y_0$

$p = (x_1 + x_0)(y_1 + y_0) = x_1y_1 + x_1y_0 + x_0y_1 + x_0y_0$

Karatsuba's Algorithm

Multiply n-digit integers x and y

Let $x = x_1 2^{n/2} + x_0$ and $y = y_1 2^{n/2} + y_0$
Recursively compute

$a = x_1 y_1$

$b = x_0 y_0$

$p = (x_1 + x_0)(y_1 + y_0)$

Return $a2^n + (p - a - b)2^{n/2} + b$

Recurrence: $T(n) = 3T(n/2) + cn$

$\log_2 3 = 1.58496250073\ldots$