CSC373
Lecture Notes

Yuchen Wang

September 12, 2019

Contents

1 Divide & Conquer

General framework

1. Break (a large chunk of) a problem into smaller subproblems of the same type
2. Solve each subproblem recursively
3. At the end, quickly combine solutions from the subproblems and/or solve any remaining part of the original problem

1.1 Master Theorem

Useful for analyzing divide-and-conquer running time

Theorem Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence.

$$T(n) = aT(\frac{n}{b}) + f(n)$$

where we interpret $\frac{n}{b}$ to mean either $\lfloor \frac{n}{b} \rfloor$ or $\lceil \frac{n}{b} \rceil$. Then $T(n)$ has the following asymptotic bounds:

1. If $f(n) = \mathcal{O}(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{b}{n}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.

Theorem (from CSC236) Divide-and-conquer algorithms: partition problem into b roughly equal subproblems, solve, and recombine:

$$T(n) \begin{cases} k & \text{if } n \leq B \\ a_1 T(\lceil n/b \rceil) + a_2 T(\lfloor n/b \rfloor) + f(n) & \text{if } n > B \end{cases}$$

where $b, k > 0, a_1, a_2 \geq 0$, and $a = a_1 + a_2 > 0$. $f(n)$ is the cost of splitting and recombining. If f from the previous slide has $f \in \theta(n^d)$, then

$$T(n) \in \begin{cases} \theta(n^d) & \text{if } a < b^d \\ \theta(n^d \log n) & \text{if } a = b^d \\ \theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

1.2 Counting Inversions

Problem Given an array a of length n, count the number of pairs (i, j) such that $i < j$ but $a[i] > a[j]$

Applications

1. Voting theory
2. Collaborative filtering
3. Measuring the "sortedness" of an array
4. Sensitivity analysis of Google's ranking function
5. Rank aggregation for meta-searching on the Web
6. Nonparametric statistics (e.g., Kendall's tau distance)

Brute Force Check all $\theta(n^2)$ pairs

Divide & conquer

1. Divide: break away into two equal halves x and y
2. Conquer: count inversions in each half recursively
3. Combine:
   Solve (remaining): count inversions with one entry in x and one in y
   Merge: add all three counts

SORT-AND-COUNT (L)

IF list L has one element
  RETURN $(0, L)$.

DIVIDE the list into two halves A and B.
$(r_A, A) \leftarrow$ SORT-AND-COUNT(A).
$(r_B, B) \leftarrow$ SORT-AND-COUNT(B).
$(r_{AB}, L') \leftarrow$ MERGE-AND-COUNT(A, B).

RETURN $(r_A + r_B + r_{AB}, L')$.

Count inversions (a, b) with $a \in A$ and $b \in B$, assuming A and B are sorted:

1. Scan A and B from left to right
2. Compare a_i and b_j
3. If $a_i < b_j$, then a_i is not inverted with any element left in B
4. If $a_i > b_j$, then b_j is inverted with every element left in A
5. Append smaller element to sorted list C

How do we formally prove correctness? Induction on n is usually very helpful, allows you to assume correctness of subproblems

Running time analysis

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

Master theorem says this is $T(n) = O(n \log n)$

1.3 Closest Pair in $\mathbb{R}^2$

Problem Given n points of the form (x_i, y_i) in the plane, find the closest pair of points.

Applications

1. Basic primitive in graphics and computer vision
2. Geographic information systems, molecular modeling, air traffic control
3. Special case of nearest neighbor

Brute force running time

$$\Theta(n^2)$$

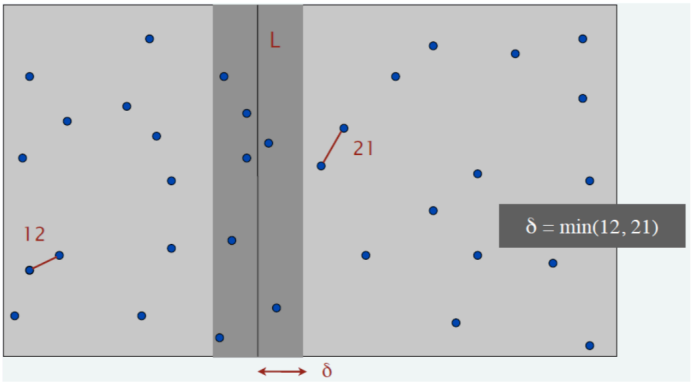
Intuition from 1D? By sorting and checking, the problem would be easily $\mathcal{O}(n \log n)$

Non-degeneracy Assumption No two points have the same x or y coordinate

Closest Pair in $\mathbb{R}^2$

1. Divide: points in equal halves by drawing a vertical line L
2. Conquer: solve each half recursively
3. Combine: find closest pair with one point on each side of L
4. Return the best of 3 solutions

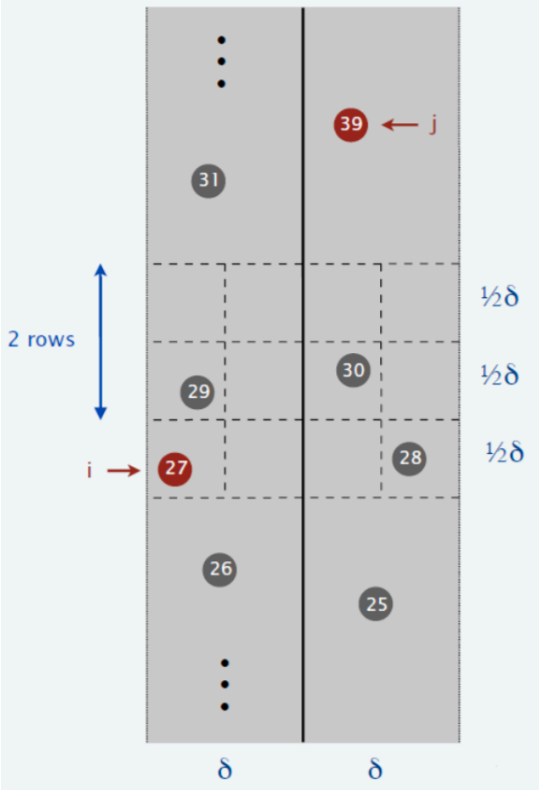
Combine: We can restrict our attention to points within ϵ of L on each side, where ϵ = best of the solutions in two halves



1. Only need to look at points within ϵ of L on each side
2. Sort points on the strip by y coordinate
3. Only need to check each point with next 11 points in sorted list

Why 11? Claim: If two points are at least 12 positions apart in the sorted list, their distance is at least ϵ.
proof:

1. No two points lie in the same $\delta/2 \times \delta/2$ box
2. Two points that are more than two rows apart are at distance at least δ


39
j
31
2 rows
½δ
½δ
½δ
30
29
28
i
27
26
25
δ
δ