7 Divide and Conquer

Tuesday, April 15, 2025 12:47 PM

Algorithm Design Technique.

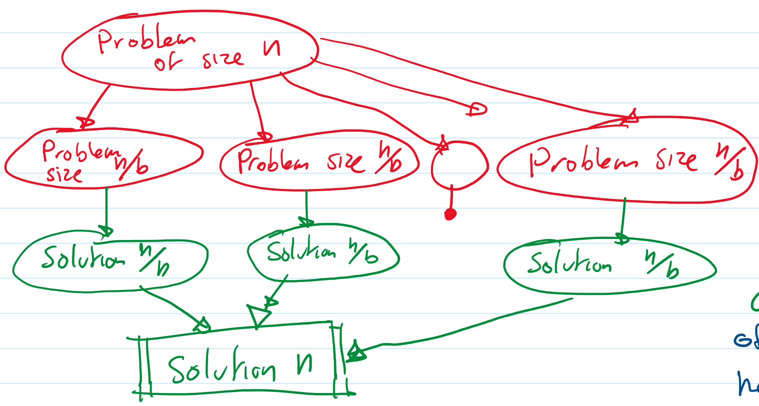
The Basic Strategy:

1.- Divide the problem into several subproblems.
- ideally of the same size
- usually 2 subproblems.

2.- Solve the subproblems
- usually recursively.

3.- The solution to the subproblems are combined to get a solution to the original problem.

- DIAGRAM:



Note;
a subproblems of size n/b need to be solved
$a \leq b$

Note: Some of the most important algorithms are of this kind.

E.g. Sum of an array

```
FUNCTION Sum(A[0..n-1])
  IF n=1 THEN
     RETURN A[0]
  ELSE
     a ← Sum(A[0..⌊n/2⌋-1])
     b ← Sum(A[⌊n/2⌋..n-1])
     RETURN a+b
```

Analysis: basic operation $+$

$$A(n) = A\left(\frac{n}{2}\right) + A\left(\frac{n}{2}\right) + 1$$
$$A(1) = 0$$

} Recurrence.

$A(1) = 0$ } Recurrence.

Solving the recurrance $A(n) \in \Theta(n)$ ....

- ANALYSIS FRAMEWORK FOR DIVIDE AND CONQUER ALGORITHMS:

Divide & Conquer will lead to Recurrences

- input of size n
- divide into b parts of equal size $\frac{n}{b}$
- solve a subparts
- the cost of recombination is $f(n)$

The general formula is:

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

"General Divide and Conquer Recurrence"

Let's solve it:

- Let n be a power of b: $n = b^k$ $\quad k = \log_b n$

$$T(b^k) = a \cdot T(b^{k-1}) + f(b^k)$$

$$T(b^{k-1}) = a \cdot T(b^{k-2}) + f(b^{k-1})$$

$$\begin{aligned}
T(b^k) &= a \cdot [a \cdot T(b^{k-2}) + f(b^{k-1})] + f(b^k) \\
&= a^2 \cdot \underline{T(b^{k-2})} + a \cdot f(b^{k-1}) + f(b^k)
\end{aligned}$$

$$T(b^{k-2}) = a \cdot T(b^{k-3}) + f(b^{k-2})$$

$$\begin{aligned}
&= a^2 \cdot [a \cdot T(b^{k-3}) + f(b^{k-2})] + a \cdot f(b^{k-1}) + f(b^k) \\
&= a^3 \cdot \underline{T(b^{k-3})} + a^2 \cdot f(b^{k-2}) + a \cdot f(b^{k-1}) + f(b^k)
\end{aligned}$$

$$T(b^{k-3}) = a \cdot T(b^{k-4}) + f(b^{k-3})$$

$$\begin{aligned}
&= a^3 \cdot [a \cdot T(b^{k-4}) + f(b^{k-3})] + a^2 \cdot f(b^{k-2}) + a \cdot f(b^{k-1}) + f(b^k) \\
&= a^4 \cdot T(b^{k-4}) + a^3 \cdot f(b^{k-3}) + a^2 \cdot f(b^{k-2}) + a \cdot f(b^{k-1}) + f(b^k)
\end{aligned}$$

after i substitutions

$$T(b^k) = a^i \cdot T(b^{k-i}) + a^{i-1} \cdot f(b^{k-i+1}) + a^{i-2} \cdot f(b^{k-i+2}) + a^{i-3} \cdot f(b^{k-i+3}) + \dots + a^0 \cdot f(b^k)$$

Let $i = k$

$$= a^k \cdot T(1) + a^{k-1} \cdot f(b^1) + a^{k-2} \cdot f(b^2) + a^{k-3} \cdot f(b^3) + \dots a^0 \cdot f(b^k)$$

$$= a^k \cdot T(1) + \frac{a^k}{a} \cdot f(b^1) + \frac{a^k}{a^2} \cdot f(b^2) + \frac{a^k}{a^3} \cdot f(b^3) + \dots + \frac{a^k}{a^k} \cdot f(b^k)$$

$$= a^k \cdot \left[T(1) + \sum_{j=1}^{k} \frac{f(b^j)}{a^j} \right] \; ; \; n = b^k \; ; \; k = \log_b n \; ; \; a^k = a^{\log_b n} = n^{\log_b a}$$

So:

$$T(n) = n^{\log_b a} \cdot \left[T(1) + \sum^{\log_b n} \frac{f(b^j)}{\dots} \right]$$

So:

$$T(n) = n^{\log_b a} \cdot \left[T(1) + \sum_{j=1}^{\log_b n} \frac{f(b^j)}{a^j} \right].$$

What impacts the order of growth of $T(n)$??

- The value of b
- The value of a
- The order of growth of $f(n)$

The Master Theorem:

Given the recurrance:

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

If $f(n) \in \Theta(n^d)$ where $d \geq 0$ then:

$$T(n) \in \begin{cases} \Theta(n^d) & \text{if } a < b^d \quad (1) \\ \Theta(n^d \log n) & \text{if } a = b^d \quad (2) \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \quad (3) \end{cases}$$

Apply #1:

$$A(n) = 2 \cdot A\left(\frac{n}{2}\right) + 1$$

$$A(1) = 0$$

$a = 2$
$b = 2$
$d = 0$

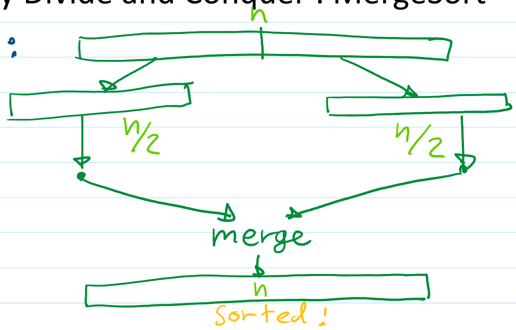
$f(n) = f(n^0) \in \Theta(n^0)$

- $a > b^d = 2 > 2^0$ so (3) applies

$$A(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 2}) = \Theta(n)$$

- Sorting by Divide and Conquer : MergeSort

Given:



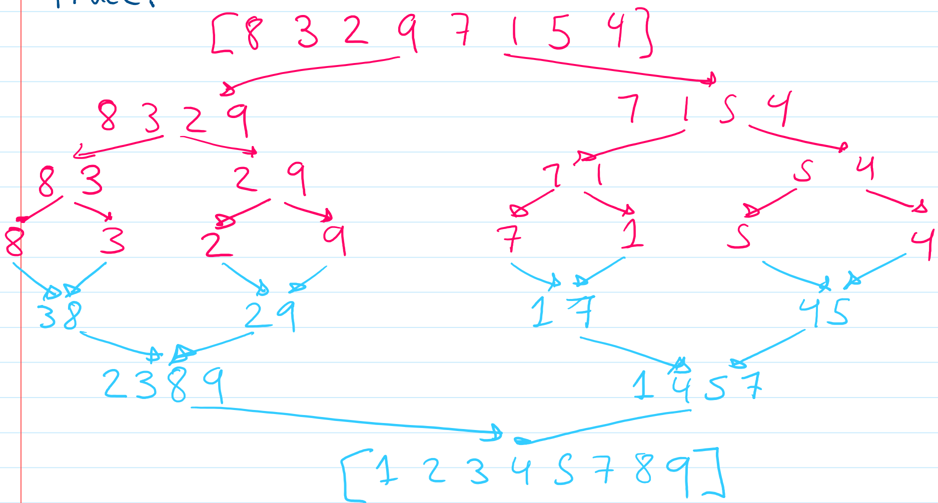
PseudoCode MergeSort $(A[0 \dots n-1])$

IF $n > 1$

copy $A[0 .. \lfloor \frac{n}{2} \rfloor - 1]$ to $B[0 \dots \lfloor \frac{n}{2} \rfloor - 1]$

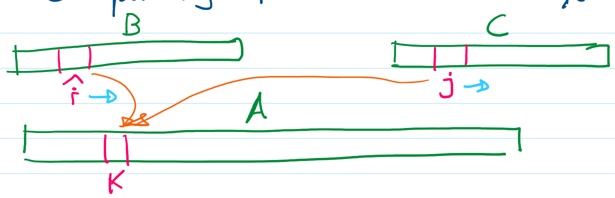
IF $n > 1$
    copy $A[0..\lfloor \frac{n}{2} \rfloor -1]$ to $B[0...\lfloor \frac{n}{2} \rfloor -1]$
    copy $A[\lfloor \frac{n}{2} \rfloor .. n-1]$ to $C[0..\lceil \frac{n}{2} \rceil -1]$
    MergeSort(B)
    Merge Sort (C)
    Merge (B,C,A)

Trace:



• Merge:

Comparing two sorted arrays



FUNCTION Merge ($B[0..p-1]$, $C[0..q-1]$, $A[0...p+q-1]$)
  $i \leftarrow 0$; $j \leftarrow 0$; $k \leftarrow 0$;
  WHILE $i < p$ and $j < q$ DO
    IF $B[i] < C[j]$ THEN
        $A[k] \leftarrow B[i]$
        $i \leftarrow i+1$
    ELSE
        $A[k] \leftarrow C[j]$
        $j \leftarrow j+1$
    $k \leftarrow k+1$
  IF $i = p$ THEN
      copy $C[j..q-1]$ to $A[k..p+q-1]$
  ELSE
      copy $B[i..p-1]$ to $A[k..p+q-1]$

MergeSort is Cool 😎
  ⊕ is "stable" elements of same value retain their position relative to each other.

- ⊕ is "stable": elements of same value retain their position relative to each other.
- ⊖ needs extra storage.

Analysis: basic operation. $<$ Comparison

$$C(n) = 2 \cdot C\left(\frac{n}{2}\right) + \underset{\text{merge.}}{T(n)}$$

$$T_{\text{merge}}(n) = n-1$$

$$C(n) = 2 \cdot C\left(\frac{n}{2}\right) + n-1$$

Apply the Master's Theorem:

$a = 2$
$b = 2$
$d = 1$ $\quad n-1 \in \Theta(n^1)$

So $a = b^d$
case ② $\quad C(n) \in \Theta(n^d \log n) = \Theta(n \cdot \log n)$ linear logarithmic.

MergeSort is better!
How Much better?

n	BubbleSort	MergeSort
100	10,000	665
500	250,000	4,483
10,000	100,000,000	132,880

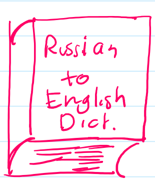
Note: n^2 algorithms are deceptive
useful for small inputs.
but eventually explode.

- Sorting by Divide and Conquer : QuickSort

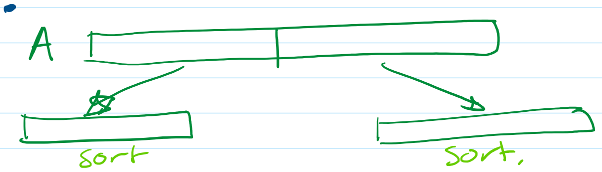
– '60 C.A.R. Hoare
"Tony"

– Russian-to-English
Translator.

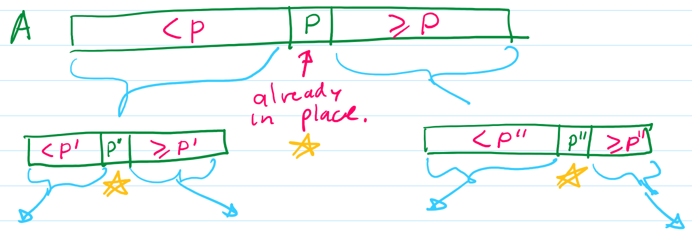
Sort the
text



Intuition:



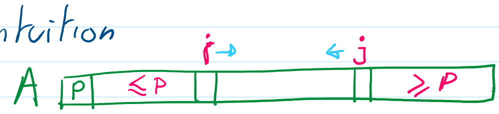
But: ¿How to avoid Merging?
by partitioning:



Algorithm: QuickSort (A[l..r])
IF l<r THEN
  s ← partition(A[l..r])
  QuickSort (A[l..s-1])
  QuickSort (A[s+1..r])

• A new way of partitioning: (Hoare Partition)

Intuition



• if A[i] ≤ p then i ← i+1
• if A[j] ≥ p Then j ← j-1

eventually: A[i] > p and A[j] < p
then Swap(A[i], A[j])

when do we stop?: when j ≤ i

FUNCTION Hoare Partition (A[l..r])
  p ← A[l]
  i ← l;   j ← r+1;
  REPEAT
    REPEAT i ← i+1 UNTIL A[i] ≥ P *
    REPEAT j ← j-1 UNTIL A[j] ≤ P
    Swap (A[i], A[j])
  UNTIL j ≤ i
  Swap (A[i], A[j]) //undo last swap.
  Swap (A[l], A[j])
  RETURN j

Trace

p: 6

6 10 7 9 3 4 8 1

i → i   ← j ← j

6 1 7 9 3 4 8 10

i   j ← j

6 1 4 9 3 7 8 10

i j

6 1 4 3 9 7 8 10

j i

6 1 4 9 3 7 8 10

6 1 4 3 9 7 8 10

3 1 4 6 9 7 8 10

* Note: this could go at the end of the

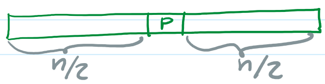
$5 \quad 2 \quad 4 \quad \boxed{6} \quad 1 \quad 7 \quad 8 \quad 9$

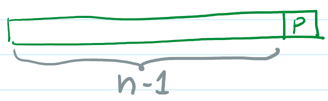
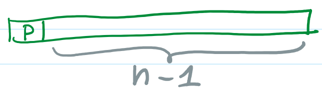
RETURN j

★ Note: this could go of the end of the array.

Analysis:

- Hoare Partition: basic operation $\geq$ $\leq$ Comparison
  Hoare partition takes: n or $n+1$ comparisons.
- Quality of Partitions:

  Best: 

  Worst:  or 

- Quick Sort.

- Best $C_{best}(n) = \underbrace{n}_{\text{partitioning}} + \underbrace{2 \cdot C(n/2)}_{\text{Quicksort each subarray}}$

  Apply Masters Theorem:

  $a = 2$ $\qquad f(n) = n \in \Theta(n^1)$

  $b = 2$

  $d = 1$

  Case: $a = b^d$ $\quad 2$

  So: $C_{best}(n) \in \Theta(n \cdot \log n)$

- Worst:

  $C_{worst} = (n+1) + n + (n-1) + (n-2) + (n-3) + \ldots\ldots + 3$

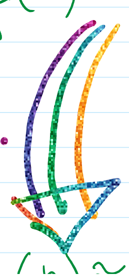
  $= \frac{(n+1)(n+2)}{2} - 3 \in \Theta(n^2)$

  Scenario: when array is already sorted (Reverse Sorted)

- Avg Case:
  - S: position of pivot $0 \leq S \leq n-1$
  - suppose each position of the pivot is equally likely then the pivot can end in a position i with probability $1/n$

$$C_{avg}(n) = \frac{1}{n} \cdot \underbrace{\sum_{S=0}^{n-1}}_{\text{positions for pivot.}} \left[\underbrace{(n+1)}_{\text{partition}} + \underbrace{C_{Avg}(S)}_{\text{Quicksort left subarray}} + \underbrace{C_{Avg}(n-1-S)}_{\text{Quicksort right subarray}} \right]$$

rainbow happens.



$C(n) \approx 2n \ln n \approx 1.39 n \log n \in \Theta(n \cdot \log n)$ linear logarithmic

$C_{avg}(n) \approx 2 \cdot n \ln n \approx 1.39 \cdot n \cdot \log_2 n \in \Theta(n \cdot \log n)$ linear-logarithmic.

On average Quicksort makes 39% more comparisons than the base case.

Quicksort:

⊕ in-place no-need for extra space

⊖ not-stable.

Improvements:

- Improve Selection of Pivot
  - pick at random.
  - look at 3 elements, pick the middle one.
- Swich to insertion sort for small subarrays $n \leq 5..15$
- 3-way partitioning (using 2 pivots)

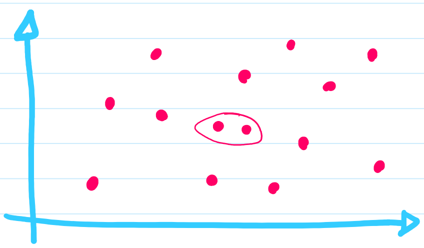
It's possible to achieve 20% to 30% improvement.

- PROBLEM : Closest Pair

Problem in the 2D Plane.

Given a collection of points $\langle x, y \rangle$ in 2D space

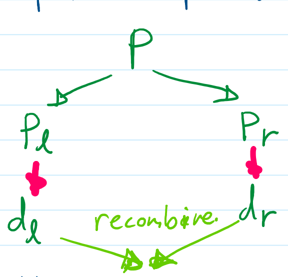
Find the closest 2 points.

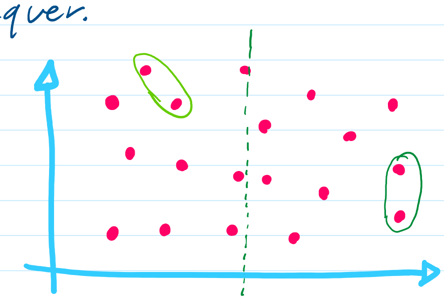


$$d(P_i, P_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

- Apply Divide and Conquer.

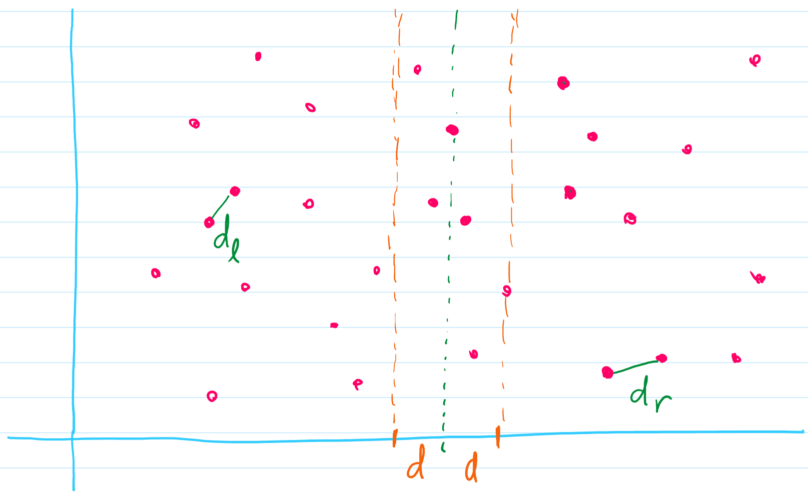
1) Spit the points.





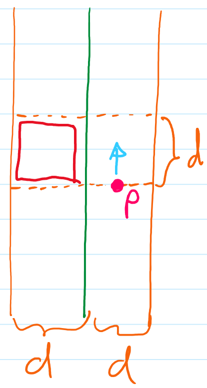
2) How are you going to recombine the solutions of the halves, to create a complete solution.

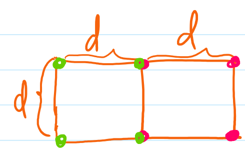
the halves, to create a complete solution.



Let d be the smallest of d_ℓ, d_r

- the only way for a closer pair to exist is if it's extremes are in opposite partitions.
- We can narrow even further





There can only be 8 points.

FUNCTION DQClosestPair (P: collection of points sorted by x value
Q: collection of points sorted by y value)

IF $|P| \le 3$ THEN
  Solve by Brute Force — 3 comparisons.
ELSE
  Copy first $\lceil \frac{n}{2} \rceil$ points of P into P_ℓ
  copy same $\lceil \frac{n}{2} \rceil$ points from Q into Q_ℓ
  copy remaining $\lfloor \frac{n}{2} \rfloor$ points of P into P_r
  copy remaining $\lfloor n/2 \rfloor$ points of Q into Q_r

  $d_\ell \leftarrow$ DQClosestPair (P_ℓ, Q_ℓ)
  $d_r \leftarrow$ DQClosestPair (P_r, Q_r)

  # merge solutions
  $d \leftarrow \min(d_\ell, d_r)$
  $m \leftarrow P[\lceil n/2 \rceil - 1].x$

$d \leftarrow \min(d_L, d_R)$

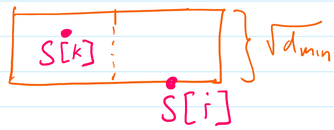
$m \leftarrow P[\lceil n/2 \rceil - 1].x$

Copy from Q all points which $|x-m| < d$ into $S[0 .. n_s - 1]$

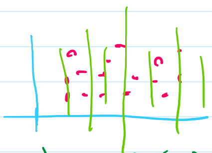
$d_{min} \leftarrow d^2$

$d(p_i, p_j)^2 = (x_i - x_j)^2 + (y_i - y_j)^2$

```
FOR i ← 0 TO n_s-1 DO
    k ← i+1
    WHILE k < n_s and (S[k].y - S[i].y)^2 < dmin DO
        d' = (S[k].x - S[i].x)^2 + (S[k].y - S[i].y)^2
        IF d' < dmin THEN
            dmin ← d'
        k ← k+1
RETURN √dmin
```



Analysis:



$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \underbrace{f(n)}_{\text{cost of searching the middle strip.}}$$

- $f(n)$ basic. operation. $<$ comparison
  $\hookrightarrow C(n) = \sum_{i=0}^{n_s-1} 8 = 8 \cdot n_s \in \Theta(n^1)$
- $f(n) \in \Theta(n^1)$

Apply Master theorem:

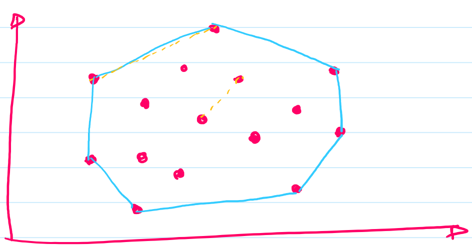
$a = 2$
$b = 2$
$d = 1$

Then: case $a = b^d$ (2)

$T(n) \in \Theta(n^d \cdot \log n) = \Theta(n \cdot \log n)$ linear-logarithmic.

Note: Presorting is not an issue.
A good sort is $\Theta(n \cdot \log n)$
merge Sort
Quick Sort

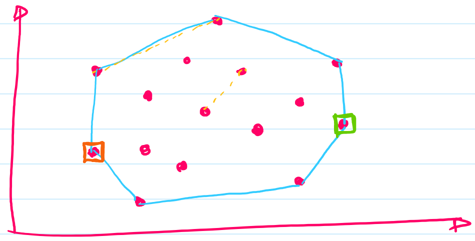
- PROBLEM : Convex Hull



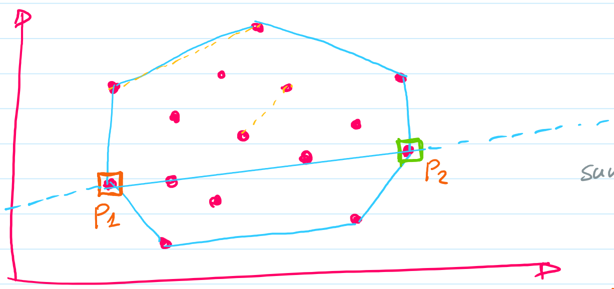
Find the smallest convex polygon that contains all points

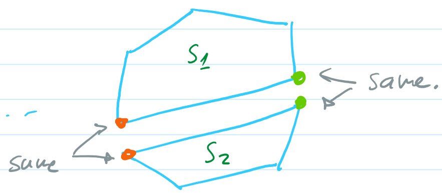
Convex polygon:- for any two points p_1, p_2 in the polygon, the line $p_1 - p_2$ resides completely inside the polygon points p_1, p_2 in the polygon, the line $p_1 - p_2$ resides completely inside the polygon

idea: sort the points by x value



We gain 2 points: the extremes are in the convex hull

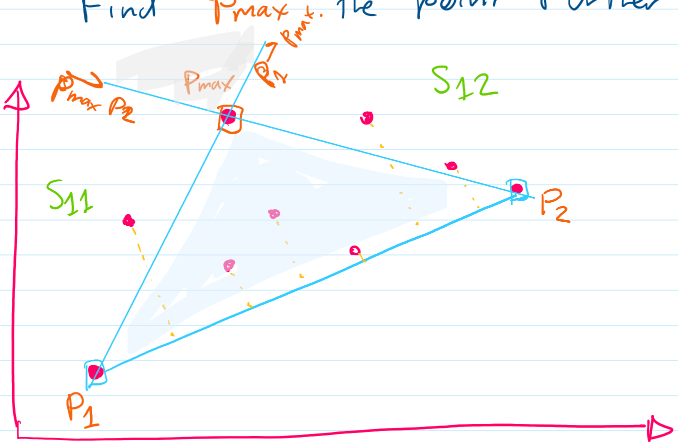




S_1: points to the left of $\overrightarrow{P_1 P_2}$
S_2: points to the right of $\overrightarrow{P_1 P_2}$

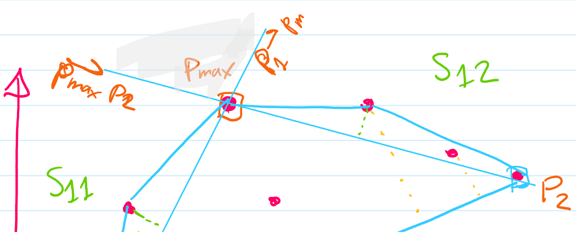
Consider S_1

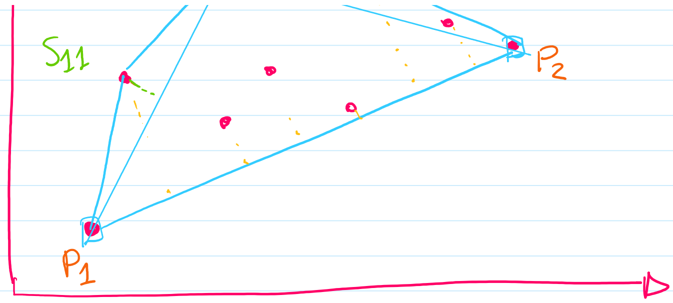
Find P_{max} s.t. the point further from the line.



- P_{max} is in the convex hull
- Points in the triangle $\Delta P_1, P_2, P_{max}$ a not in the convex hull
- there can be no points left of $P_1 \to P_{max}$ or left of $P_{max} \to P_2$

Repeat the same process in S_{11} and S_{12}





Base Case. $|S| = \emptyset$ the base line is in the convex hull
$|S| = 1$ the lines to the point are in the hull

- How to compare 3 points?

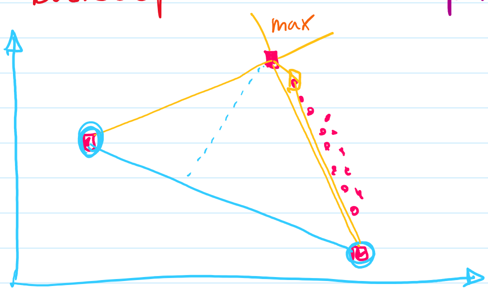
Given 3 points q_1 q_2 q_3 we can get information about the $\Delta q_1, q_2, q_3$ by looking at the determinant of the matrix

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \stackrel{det}{=} x_1 \cdot y_2 + x_3 \cdot y_1 + x_2 \cdot y_3 - x_3 \cdot y_2 - x_2 \cdot y_1 - x_1 \cdot y_3$$

- the determinant is twice the area of the triangle $\Delta q_1 q_2 q_3$
- If q_3 is to the left of $\overrightarrow{q_1 q_2}$, the area will be positive otherwize, it will be negative.

Analysis: basic operation Comparing two points.

Consider the worst



- $T_{worst}(n) = n + (n-1) + (n-2) + (n-3) + \dots + 1 = \frac{n(n+1)}{2}$
  $\in \Theta(n^2)$ cuadratic.
- $T_{avg}(n) \in \Theta(n \log n)$ good news!!
- With a normal distribution of points $T(n) \in \Theta(n)$

- PROBLEM : Multiplication

12875 "Algorisms"

- PROBLEM : Multiplication

e.g.

$$\begin{array}{r} 17875 \\ \times\ 31027 \\ \hline 125125 \\ 35750 \\ 00000 \\ 17875 \end{array}$$

"Algorisms"

Analysis: basic operation. Single digit multiplication.

$$M(n) = n^2$$

Can we do better? "Anatoly Karatsuba"

$$\begin{array}{rr|r|r} & & a_1 & a_2 \\ & a & 1233 & 5727 \\ \times & b & 0092 & 3782 \\ \hline & & b_1 & b_2 \end{array}$$

$$a \cdot b = (a_1 \cdot b_1) \cdot 10^8 + \left((a_2 \cdot b_1) + (a_1 \cdot b_2)\right) \cdot 10^4 + (a_2 \cdot b_2)$$

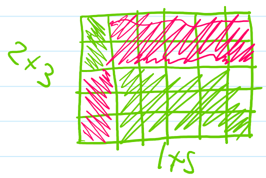
Let us re-formulate to save on multiplications.

e.g

$$\begin{array}{r} 23 \\ \times\ 15 \\ \hline \end{array}$$

$$23 \cdot 15 = (2 \cdot 1) \cdot 10^2 + \left((2 \cdot 5) + (3 \cdot 1)\right) \cdot 10^1 + (3 \cdot 5) \cdot 10^0$$

Lets change Perspective:



5·6

$$(2 \cdot 5) + (3 \cdot 1) = 5 \cdot 6 - (2 \cdot 1) - (3 \cdot 5)$$

$$23 \cdot 15 = (2 \cdot 1) \cdot 10^2 + \left((5 \cdot 6) - (2 \cdot 1) - (3 \cdot 5)\right) \cdot 10^1 + (3 \cdot 5) \cdot 10^0$$

In general, for 2 digit number.

$$a = a_1 a_0 \qquad b = b_1 b_0$$

$$a \cdot b = C_2 \cdot 10^2 + C_1 \cdot 10^1 + C_3 \cdot 10^0$$

$$C_2 = a_1 \cdot b_1$$

$$C_0 = a_0 \cdot b_0$$

$$C_1 = (a_1 + a_0) \cdot (b_1 + b_0) - C_2 - C_0$$

Let's scale to multi-digit numbers of length n digits

$a = a_1 a_0$ — a_1, a_0 are multi-digit

$$a = a_1 \cdot 10^{n/2} + a_0$$

$b = b_1 b_0$ — b_1, b_0 are multi-digit

$$b = b_1 \cdot 10^{\frac{n}{2}} + b_0$$

$b = b_1 b_0$ $\quad b_1, b_0$ are multi-digit

$$b = b_1 \cdot 10^{\frac{n}{2}} + b_0$$

$$a \cdot b = C_2 \cdot 10^n + C_1 \cdot 10^{n/2} + C_0 \cdot 10^0$$

$$C_2 = a_1 \cdot b_1$$

$$C_0 = a_0 \cdot b_0$$

$$C_1 = (a_1 + a_0) \cdot (b_1 + b_0) - (C_2 + C_1)$$

Analysis: basic operation multiplication.

$$M(n) = 3 \cdot M(n/2) \qquad M(1) = 1$$

Let

$$n = 2^k \qquad k = \log_2 n$$

$$M(2^k) = 3 \cdot M(2^{k-1}) \quad \text{by backwards subtitution}$$

$$M(2^{k-1}) = 3 \cdot M(2^{k-2})$$

$$= 3 \cdot (3 \cdot M(2^{k-2}))$$

$$= 3 \cdot (3 \cdot (3 \cdot M(2^{k-3})))$$

$$= 3 \cdot (3 \cdot (3 \cdot (3 \cdot M(2^{k-4}))))$$

$$= 3^4 \cdot M(2^{k-4})$$

after i substitutions

$$M(2^k) = 3^i \cdot M(2^{k-i})$$

Let $i = k$

$$M(2^k) = 3^k \cdot M(2^{k-k})$$

$$= 3^k \cdot M(2^0)$$

$$M(2^k) = 3^k$$

$$M(n) = 3^{\log_2 n} = n^{\log_2 3} \approx n^{1.585} \quad \text{polynomial}$$

What about additions?

$$A(n) = 3 \cdot A\left(\frac{n}{2}\right) + \underline{5n}$$

Apply Masters theorem:

$a = 3$

$b = 2$ $\qquad 5n \in \Theta(n^1)$

$d = 1$

- $a > b^d$ so (3) applies

$$A(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 3}) = \Theta(n^{1.585})$$

- PROBLEM : Matrix Multiplication

Original

$$\begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \cdot \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} = \begin{bmatrix} a_{00}\cdot b_{00} + a_{01}\cdot b_{10} & a_{00}\cdot b_{01} + a_{01}\cdot b_{10} \\ a_{10}\cdot b_{00} + a_{11}\cdot b_{10} & a_{10}\cdot b_{01} + a_{11}\cdot b_{11} \end{bmatrix}$$

$n = 2$ Multiplications = 8

$$M(n) = n^3$$

V. Strassen:

$$\begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \cdot \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} = \begin{bmatrix} m_1 + m_4 - m_5 + m_7 & m_3 + m_5 \\ m_2 + m_4 & m_1 + m_3 - m_2 + m_6 \end{bmatrix}$$

where:

$$m_1 = (a_{00} + a_{11}) \cdot (b_{00} + b_{11})$$
$$m_2 = (a_{10} + a_{11}) \cdot b_{00}$$
$$m_3 = a_{00} \cdot (b_{01} - b_{11})$$
$$m_4 = a_{11} \cdot (b_{10} - b_{00})$$
$$m_5 = (a_{00} + a_{01}) \cdot b_{11}$$
$$m_6 = (a_{10} + a_{00}) \cdot (b_{00} + b_{01})$$
$$m_7 = (a_{01} - a_{11}) \cdot (b_{10} + b_{11})$$

7 multiplications.

Sample:

$$\begin{aligned} m_3 + m_5 &= a_{00}\cdot(b_{01} - b_{11}) + (a_{00} + a_{01})\cdot b_{11} \\ &= a_{00}\cdot b_{01} - \cancel{a_{00}\cdot b_{11}} + \cancel{a_{00}\cdot b_{11}} + a_{01}\cdot b_{11} \\ &= a_{00}\cdot b_{01} + a_{01}\cdot b_{11} \end{aligned}$$

How to apply this to large Matrices

Suppose the matrizes are of size 2^k

$$\begin{array}{ccccc} A & & B & & C \\ \left[\begin{array}{c|c} A_{00} & A_{01} \\ \hline A_{10} & A_{11} \end{array}\right] & \cdot & \left[\begin{array}{c|c} B_{00} & B_{01} \\ \hline B_{10} & B_{11} \end{array}\right] & = & \left[\begin{array}{c|c} C_{00} & C_{01} \\ \hline C_{10} & C_{11} \end{array}\right] \end{array}$$

Obtain C from recursive applications of Strassen's Algorithm.

Analysis: counting Multiplications.

Analysis: counting Multiplications.

$$M(n) = 7 \cdot M\left(\frac{n}{2}\right) \qquad M(1) = 1$$

Let $n = 2^k$ $\quad k = \log_2 n$ $\quad$ by backwards substitution

$$M(2^k) = 7 \cdot M(2^{k-1})$$

$$= 7 \cdot (7 \cdot M(2^{k-2})) \qquad M(2^{k-1}) = 7 \cdot M(2^{k-2})$$

$$= 7^2 \cdot M(2^{k-2})$$

$$= 7^3 \cdot M(2^{k-3})$$

$$= 7^4 \cdot M(2^{k-4})$$

after i substitutions

$$M(2^k) = 7^i \cdot M(2^{k-i})$$

Let $i = k$

$$= 7^k \cdot M(2^{k-k}) = 7^k \cdot M(2^0) = 7^k$$

$$M(n) = 7^{\log_2 n} = n^{\log_2 7} \approx n^{2.807}$$

Let's count Additions and Subtractions.

$$A(n) = 7 \cdot A\left(\frac{n}{2}\right) + \underbrace{18 \cdot \left(\frac{n}{2}\right)^2}$$

adding two matrices of size $n/2$

Apply Master's theorem:

$a = 7$

$b = 2$

$d = 2$

$$18 \cdot \left(\frac{n}{2}\right)^2 \in \Theta(n^2)$$

then $a > b^d$ so 3

$$A(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 7}) = \Theta(n^{2.807})$$

There are better ones!!!

Timeline of matrix multiplication exponent

Year	Bound on omega	Authors
1969	2.8074	Strassen[1]
1978	2.796	Pan[9]
1979	2.780	Bini, Capovani [it], Romani[10]
1981	2.522	Schönhage[11]
1981	2.517	Romani[12]
1981	2.496	Coppersmith, Winograd[13]
1986	2.479	Strassen[14]
1990	2.3755	Coppersmith, Winograd[15]
2010	2.3737	Stothers[16]
2012	2.3729	Williams[17][18]
2014	2.3728639	Le Gall[19]
2020	2.3728596	Alman, Williams[20][21]
2022	2.371866	Duan, Wu, Zhou[22]
2024	2.371552	Williams, Xu, Xu, and Zhou[23]
2024	2.371339	Alman, Duan, Williams, Xu, Xu, and Zhou[2]

EoF