CMPSC 465: LECTURE VII

More Examples of Divide-and-Conquer

Ke Chen

September 12, 2025

Median of medians as pivot for selection

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

Time complexity?

- Worst-case: $T(n) = c \cdot n/5 + T(n/5) + \Theta(n) + T(\,?\,)$
$$= T(n/5) + T(\,?\,) + \Theta(n).$$

How good is MoM?

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

$n = 30$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

$n = 30$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

$n = 30$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

- m $\leq (n/5)/2 = n/10$ medians; also $\geq n/10$ medians.

$n = 30$, $m = 20$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

- m $\leq (n/5)/2 = n/10$ medians; also $\geq n/10$ medians.
- For each median $\leq m$, there are at least 2 more in its group $\leq m$.

$n = 30$, $m = 20$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

- m $\leq (n/5)/2 = n/10$ medians; also $\geq n/10$ medians.
- For each median $\leq m$, there are at least 2 more in its group $\leq m$.
- For each median $\geq m$, there are at least 2 more in its group $\geq m$.

$n = 30$, $m = 20$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

- m $\leq (n/5)/2 = n/10$ medians; also $\geq n/10$ medians.
- For each median $\leq m$, there are at least 2 more in its group $\leq m$.
- For each median $\geq m$, there are at least 2 more in its group $\geq m$.

$n = 30$, $m = 20$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

- m $\leq (n/5)/2 = n/10$ medians; also $\geq n/10$ medians.
- For each median $\leq m$, there are at least 2 more in its group $\leq m$.
- For each median $\geq m$, there are at least 2 more in its group $\geq m$.
- In total, we can guarantee $3(n/5)/2 = 3n/10$ numbers $\leq m$; also $3n/10$ numbers $\geq m$.

$n = 30$, $m = 20$

19	7	15	42	5	88	91	4	29	21
13	67	54	18	20	73	8	36	49	2
9	25	31	44	12	6	80	14	22	3

How good is MoM?

- m $\leq (n/5)/2 = n/10$ medians; also $\geq n/10$ medians.
- For each median $\leq m$, there are at least 2 more in its group $\leq m$.
- For each median $\geq m$, there are at least 2 more in its group $\geq m$.
- In total, we can guarantee $3(n/5)/2 = 3n/10$ numbers $\leq m$; also $3n/10$ numbers $\geq m$.

$n = 30$, $m = 20$, 15 numbers $\leq m$, 15 numbers $\geq m$

19	7	15	5	4	13	18	3	8	2
9	12	6	14	20	73	29	36	49	21
88	25	31	44	67	54	80	91	22	42

Median of medians as pivot

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

Time complexity?

- Worst-case: $T(n) = T(n/5) + T(\text{?}) + \Theta(n)$

Median of medians as pivot

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

Time complexity?

- Worst-case: $T(n) = T(n/5) + T($ n-3(n/5)/2 $) + \Theta(n)$

Median of medians as pivot

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

Time complexity?

- Worst-case: $T(n) = T(n/5) + T(\text{n-3(n/5)/2}) + \Theta(n)$

$$= T(n/5) + T(\text{7n/10}) + \Theta(n).$$

Median of medians as pivot

Select(A, k):

1. Divide the input A into groups of 5.
2. Find the median of each group.
3. Recursively find the median m of all these $n/5$ medians.
4. Partition A with pivot m.
5. Do recursive call as in RandomizedSelect.

Time complexity?

- Worst-case: $T(n) = T(n/5) + T(\text{n-3(n/5)/2}) + \Theta(n)$

$$= T(n/5) + T(\text{7n/10}) + \Theta(n).$$

- (Exercise) Prove that $T(n) = O(n)$.
- (Exercise) What happens if we group by 3, or 7?

Matrix multiplication

Input: Two $n \times n$ matrices A and B

Output: $C = A \cdot B$

Matrix multiplication

Input: Two $n \times n$ matrices A and B

Output: $C = A \cdot B$

Recall:

$$\begin{array}{ccccc} A & \cdot & B & = & C \\ \begin{bmatrix} & & \\ & i & \\ & & \\ & & \end{bmatrix} & \cdot & \begin{bmatrix} & & \\ & & \\ & & j \\ & & \end{bmatrix} & = & \begin{bmatrix} & & \\ & & i,j \\ & & \\ & & \end{bmatrix} \end{array}$$

Matrix multiplication

Input: Two $n \times n$ matrices A and B

Output: $C = A \cdot B$

Recall:

$$\begin{array}{ccccc} A & \cdot & B & = & C \\ \begin{bmatrix} & & \\ & i & \\ & & \\ & & \end{bmatrix} & \cdot & \begin{bmatrix} & & j & \\ & & j & \\ & & j & \\ & & j & \end{bmatrix} & = & \begin{bmatrix} & & i,j & \\ & & & \\ & & & \\ & & & \end{bmatrix} \end{array}$$

$$C(i,j) = \sum_{k=1}^{n} A(i,k) \cdot B(k,j).$$

Matrix multiplication

Input: Two $n \times n$ matrices A and B

Output: $C = A \cdot B$

Recall:

$$\begin{array}{ccccc} A & \cdot & B & = & C \\ \begin{bmatrix} & & \\ & i & \\ & & \\ & & \\ & & \end{bmatrix} & \cdot & \begin{bmatrix} & & & \\ & & & \\ & & j & \\ & & & \\ & & & \end{bmatrix} & = & \begin{bmatrix} & & \\ & & i,j \\ & & \\ & & \\ & & \end{bmatrix} \end{array}$$

$$C(i,j) = \sum_{k=1}^{n} A(i,k) \cdot B(k,j).$$

Direct computation takes $O(n^3)$ time. Obvious lower bound is $\Omega(n^2)$.

Matrix multiplication

Input: Two $n \times n$ matrices A and B

Output: $C = A \cdot B$

Recall:

$$\begin{array}{ccccc} A & \cdot & B & = & C \\ \begin{bmatrix} \\ \qquad i \qquad \\ \\ \\ \\ \end{bmatrix} & \cdot & \begin{bmatrix} & & \\ & & \\ & & j \\ & & \\ & & \end{bmatrix} & = & \begin{bmatrix} & & \\ & & i,j \\ & & \\ & & \\ & & \end{bmatrix} \end{array}$$

$$C(i,j) = \sum_{k=1}^{n} A(i,k) \cdot B(k,j).$$

Direct computation takes $O(n^3)$ time. Obvious lower bound is $\Omega(n^2)$. **Can we do better?**

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Fact: $C = A \cdot B = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Fact: $C = A \cdot B = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$

- We obtain a divide-and-conquer algorithm with 8 subproblems of size $n/2$ each.

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Fact: $C = A \cdot B = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$

- We obtain a divide-and-conquer algorithm with 8 subproblems of size $n/2$ each.
- To combine results of subproblems, we need 4 additions of $n/2 \times n/2$ matrices, which take $O(n^2)$ time.

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Fact: $C = A \cdot B = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$

- We obtain a divide-and-conquer algorithm with 8 subproblems of size $n/2$ each.
- To combine results of subproblems, we need 4 additions of $n/2 \times n/2$ matrices, which take $O(n^2)$ time.
- So the running time satisfies $T(n) = 8T(n/2) + O(n^2) \rightsquigarrow$

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Fact: $C = A \cdot B = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$

- We obtain a divide-and-conquer algorithm with 8 subproblems of size $n/2$ each.
- To combine results of subproblems, we need 4 additions of $n/2 \times n/2$ matrices, which take $O(n^2)$ time.
- So the running time satisfies $T(n) = 8T(n/2) + O(n^2) \rightsquigarrow \boxed{T(n) = O(n^3)}$.

Matrix multiplication by divide-and-conquer

Idea: Divide matrices into four $n/2 \times n/2$ blocks and perform multiplication block-wise.

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Fact: $C = A \cdot B = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$

- We obtain a divide-and-conquer algorithm with 8 subproblems of size $n/2$ each.
- To combine results of subproblems, we need 4 additions of $n/2 \times n/2$ matrices, which take $O(n^2)$ time.
- So the running time satisfies $T(n) = 8T(n/2) + O(n^2) \rightsquigarrow \boxed{T(n) = O(n^3)}$.

Same as direct computation!

Strassen's idea (1969)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Let

$$\begin{array}{ll}
P_1 = A_{11}(B_{12} - B_{22}) & P_2 = (A_{11} + A_{12})B_{22} \\
P_3 = (A_{21} + A_{22})B_{11} & P_4 = A_{22}(B_{21} - B_{11}) \\
P_5 = (A_{11} + A_{22})(B_{11} + B_{22}) & P_6 = (A_{12} - A_{22})(B_{21} + B_{22}) \\
P_7 = (A_{11} - A_{21})(B_{11} + B_{12}) &
\end{array}$$

Strassen's idea (1969)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Let

$$\begin{aligned} P_1 &= A_{11}(B_{12} - B_{22}) & P_2 &= (A_{11} + A_{12})B_{22} \\ P_3 &= (A_{21} + A_{22})B_{11} & P_4 &= A_{22}(B_{21} - B_{11}) \\ P_5 &= (A_{11} + A_{22})(B_{11} + B_{22}) & P_6 &= (A_{12} - A_{22})(B_{21} + B_{22}) \\ P_7 &= (A_{11} - A_{21})(B_{11} + B_{12}) \end{aligned}$$

Then $A \cdot B = C = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$

Strassen's idea (1969)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Let

$$\begin{aligned}
P_1 &= A_{11}(B_{12} - B_{22}) & P_2 &= (A_{11} + A_{12})B_{22} \\
P_3 &= (A_{21} + A_{22})B_{11} & P_4 &= A_{22}(B_{21} - B_{11}) \\
P_5 &= (A_{11} + A_{22})(B_{11} + B_{22}) & P_6 &= (A_{12} - A_{22})(B_{21} + B_{22}) \\
P_7 &= (A_{11} - A_{21})(B_{11} + B_{12})
\end{aligned}$$

Then $A \cdot B = C = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$

Verify: $$\begin{aligned} P_1 + P_2 &= A_{11}(B_{12} - B_{22}) + (A_{11} + A_{12})B_{22} \\ &= A_{11}B_{12} - A_{11}B_{22} + A_{11}B_{22} + A_{12}B_{22} \\ &= A_{11}B_{12} + A_{12}B_{22} \end{aligned}$$

Strassen's idea (1969)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Let

$$\begin{array}{ll}
P_1 = A_{11}(B_{12} - B_{22}) & P_2 = (A_{11} + A_{12})B_{22} \\
P_3 = (A_{21} + A_{22})B_{11} & P_4 = A_{22}(B_{21} - B_{11}) \\
P_5 = (A_{11} + A_{22})(B_{11} + B_{22}) & P_6 = (A_{12} - A_{22})(B_{21} + B_{22}) \\
P_7 = (A_{11} - A_{21})(B_{11} + B_{12}) &
\end{array}$$

Then $A \cdot B = C = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$

$P_1, \ldots, P_7$ requires 7 $n/2 \times n/2$ matrix multiplications, therefore
$T(n) = 7T(n/2) + O(n^2) \rightsquigarrow$.

Strassen's idea (1969)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Let

$$\begin{array}{ll} P_1 = A_{11}(B_{12} - B_{22}) & P_2 = (A_{11} + A_{12})B_{22} \\ P_3 = (A_{21} + A_{22})B_{11} & P_4 = A_{22}(B_{21} - B_{11}) \\ P_5 = (A_{11} + A_{22})(B_{11} + B_{22}) & P_6 = (A_{12} - A_{22})(B_{21} + B_{22}) \\ P_7 = (A_{11} - A_{21})(B_{11} + B_{12}) & \end{array}$$

Then $A \cdot B = C = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$

$P_1, \ldots, P_7$ requires 7 $n/2 \times n/2$ matrix multiplications, therefore
$T(n) = 7T(n/2) + O(n^2) \rightsquigarrow O(n^{\log 7}) = O(n^{2.81})$.

Strassen's idea (1969)

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

Let

$$\begin{aligned} P_1 &= A_{11}(B_{12} - B_{22}) & P_2 &= (A_{11} + A_{12})B_{22} \\ P_3 &= (A_{21} + A_{22})B_{11} & P_4 &= A_{22}(B_{21} - B_{11}) \\ P_5 &= (A_{11} + A_{22})(B_{11} + B_{22}) & P_6 &= (A_{12} - A_{22})(B_{21} + B_{22}) \\ P_7 &= (A_{11} - A_{21})(B_{11} + B_{12}) \end{aligned}$$

Then $A \cdot B = C = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$

$P_1, \ldots, P_7$ requires 7 $n/2 \times n/2$ matrix multiplications, therefore $T(n) = 7T(n/2) + O(n^2) \rightsquigarrow O(n^{\log 7}) = O(n^{2.81})$.

Can we do better?

Matrix multiplication

Can we do better?

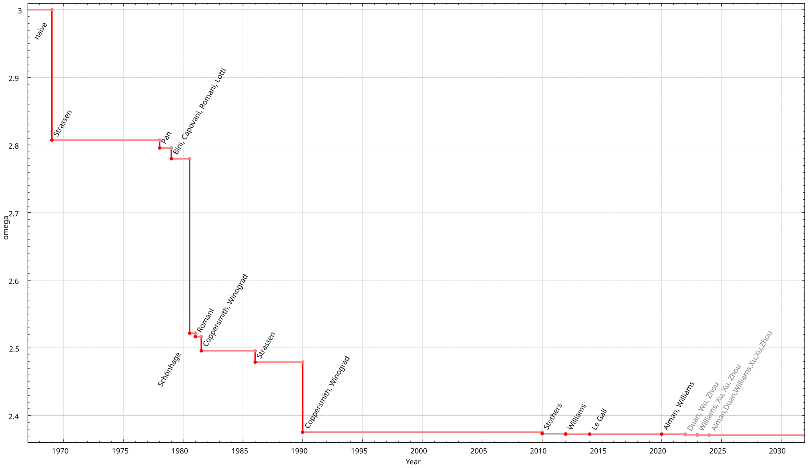
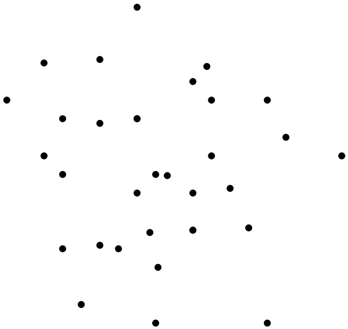


Figure by Jochen Burghardt

Closest pair of points

Input: A set of n points in the plane.

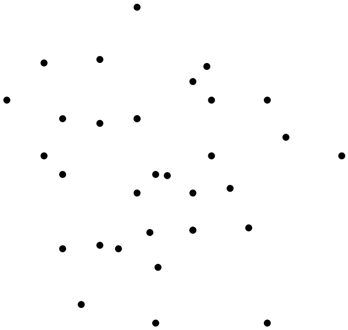
Output: A pair of points with the smallest Euclidean distance.



Closest pair of points

Input: A set of n points in the plane.

Output: A pair of points with the smallest Euclidean distance.

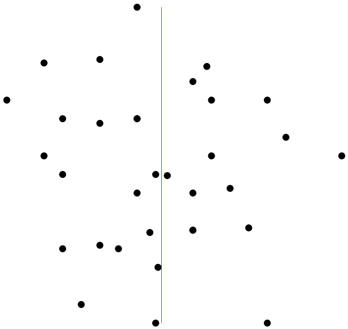


Brute-force: $O(n^2)$.

Closest pair of points

Input: A set of n points in the plane.

Output: A pair of points with the smallest Euclidean distance.



Brute-force: $O(n^2)$.

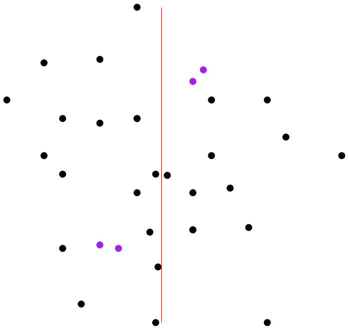
Divide-and-conquer:

1. Divide the points into two parts.

Closest pair of points

Input: A set of n points in the plane.

Output: A pair of points with the smallest Euclidean distance.



Brute-force: $O(n^2)$.

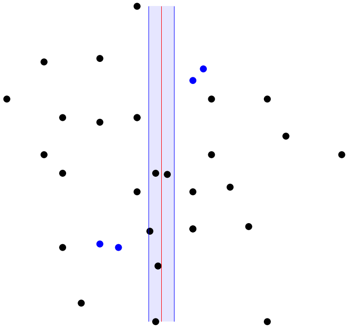
Divide-and-conquer:

1. Divide the points into two parts.
2. Find a closest pair in each.

Closest pair of points

Input: A set of n points in the plane.

Output: A pair of points with the smallest Euclidean distance.



Brute-force: $O(n^2)$.

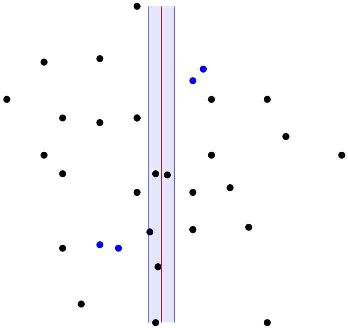
Divide-and-conquer:

1. Divide the points into two parts.
2. Find a closest pair in each.
3. Find if any cross-boundary pair is closer.

Closest pair of points

Input: A set of n points in the plane.

Output: A pair of points with the smallest Euclidean distance.



Brute-force: $O(n^2)$.

Divide-and-conquer:

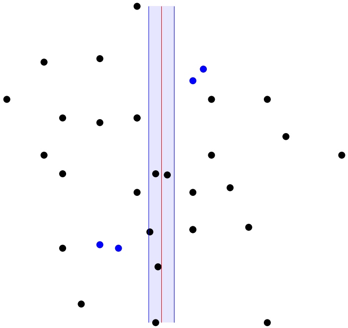
1. Divide the points into two parts.
2. Find a closest pair in each.
3. Find if any cross-boundary pair is closer.

Can be done in $O(n)$ time, therefore $T(n) = 2T(n/2) + O(n) \rightsquigarrow$

Closest pair of points

Input: A set of n points in the plane.

Output: A pair of points with the smallest Euclidean distance.



Brute-force: $O(n^2)$.

Divide-and-conquer:

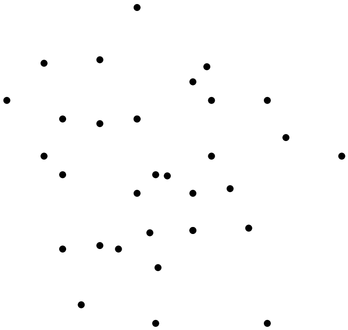
1. Divide the points into two parts.
2. Find a closest pair in each.
3. Find if any cross-boundary pair is closer.

Can be done in $O(n)$ time, therefore $T(n) = 2T(n/2) + O(n) \rightsquigarrow$ $\boxed{O(n \log n)}$.

Convex hull

Input: A set of n points in the plane.

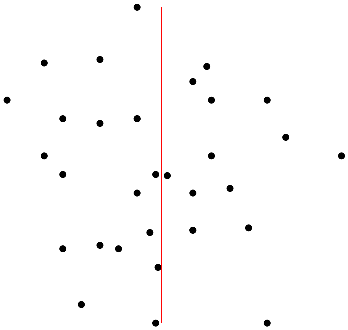
Output: The convex hull, i.e., smallest convex shape that contains all the points.



Convex hull

Input: A set of n points in the plane.

Output: The convex hull, i.e., smallest convex shape that contains all the points.



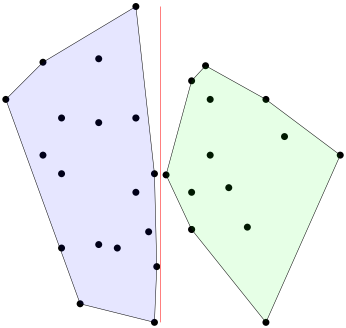
Divide-and-conquer:

1 Divide the points into two parts.

Convex hull

Input: A set of n points in the plane.

Output: The convex hull, i.e., smallest convex shape that contains all the points.



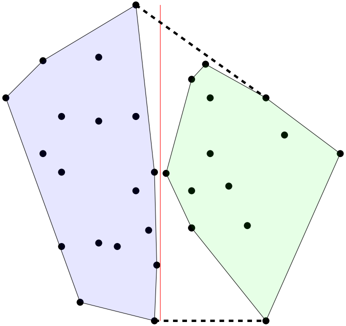
Divide-and-conquer:

1. Divide the points into two parts.
2. Compute the convex hull of each.

Convex hull

Input: A set of n points in the plane.

Output: The convex hull, i.e., smallest convex shape that contains all the points.



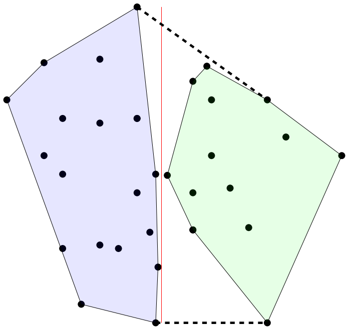
Divide-and-conquer:

1. Divide the points into two parts.
2. Compute the convex hull of each.
3. Merge the two convex polygons.

Convex hull

Input: A set of n points in the plane.

Output: The convex hull, i.e., smallest convex shape that contains all the points.



Divide-and-conquer:

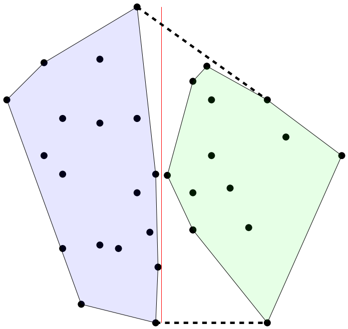
1. Divide the points into two parts.
2. Compute the convex hull of each.
3. Merge the two convex polygons.

Can be done in $O(n)$ time, therefore $T(n) = 2T(n/2) + O(n) \rightsquigarrow$

Convex hull

Input: A set of n points in the plane.

Output: The convex hull, i.e., smallest convex shape that contains all the points.



Divide-and-conquer:

1. Divide the points into two parts.
2. Compute the convex hull of each.
3. Merge the two convex polygons.

Can be done in $O(n)$ time, therefore $T(n) = 2T(n/2) + O(n) \rightsquigarrow O(n \log n)$.