Faculty Development Program

Design and Analysis of Algorithms

Brute Force and Exhaustive Search

R S Milton

miltonrs@ssn.edu.in

SSN College of Engineering

Brute Force, or “Just Do It!”

- Systematically try all possibilities.

Brute Force, or "Just Do It!"

- Systematically try all possibilities.
- A lot of careful and accurate work, fit for computer.

Brute Force, or "Just Do It!"

- Systematically try all possibilities.
- A lot of careful and accurate work, fit for computer.
- As the problem size increases, impractical even for a computer!

Brute Force, or “Just Do It!”

- Systematically try all possibilities.
- A lot of careful and accurate work, fit for computer.
- As the problem size increases, impractical even for a computer!
- Helps to understand the nature of the problem.

Brute Force, or “Just Do It!”

- Systematically try all possibilities.
- A lot of careful and accurate work, fit for computer.
- As the problem size increases, impractical even for a computer!
- Helps to understand the nature of the problem.
- Decompose to small problems,

Brute Force, or “Just Do It!”

- Systematically try all possibilities.
- A lot of careful and accurate work, fit for computer.
- As the problem size increases, impractical even for a computer!
- Helps to understand the nature of the problem.
- Decompose to small problems, and then use brute force!

Brute Force, or "Just Do It!"

- Systematically try all possibilities.
- A lot of careful and accurate work, fit for computer.
- As the problem size increases, impractical even for a computer!
- Helps to understand the nature of the problem.
- Decompose to small problems, and then use brute force!
- Ultimately, brute force is the only method!

Brute Force, or "Just Do It!"

- Systematically try all possibilities.
- A lot of careful and accurate work, fit for computer.
- As the problem size increases, impractical even for a computer!
- Helps to understand the nature of the problem.
- Decompose to small problems, and then use brute force!
- Ultimately, brute force is the only method!

Postpone it as long as possible!

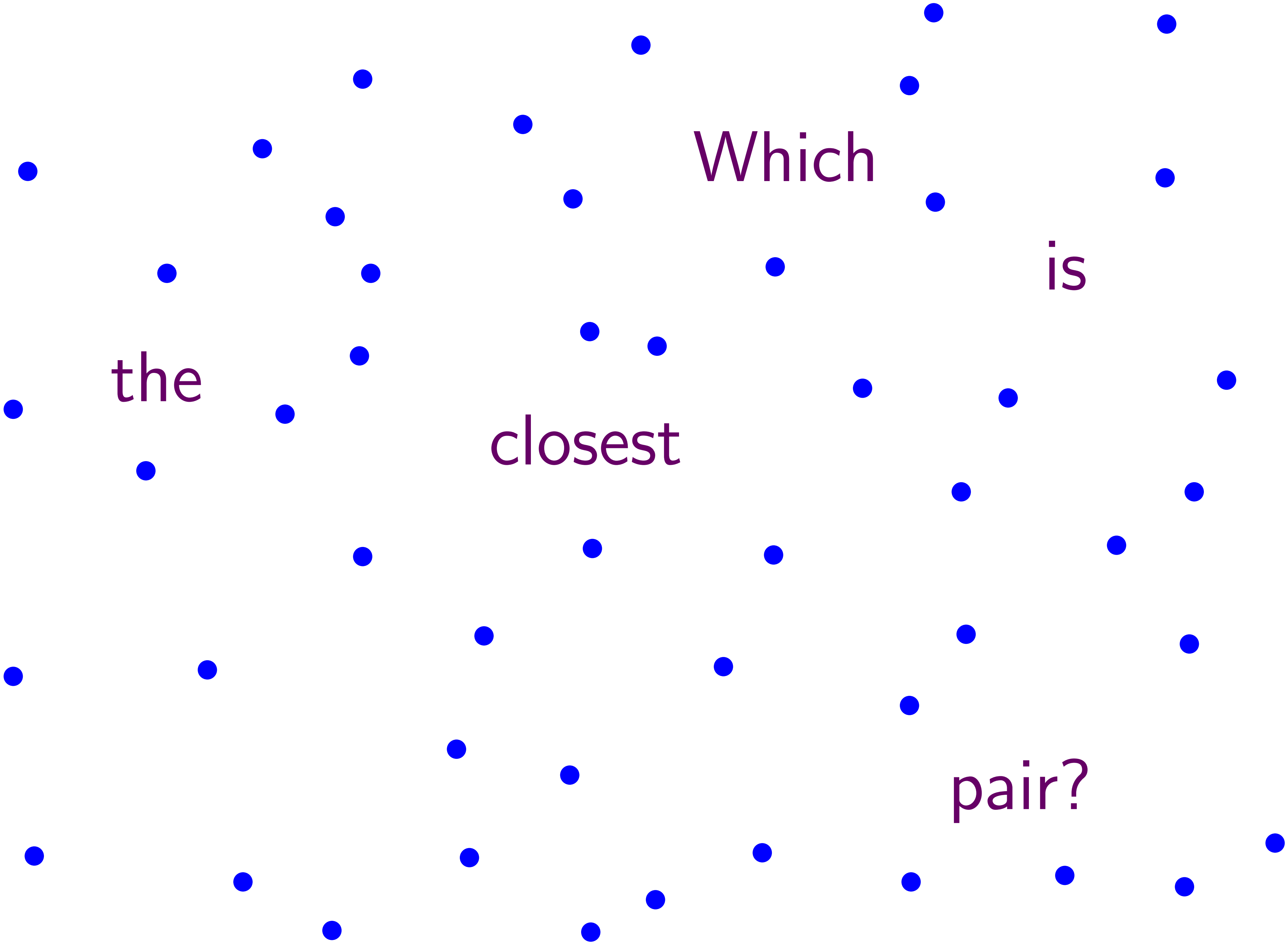
Problems

- Selection sort
- Insertion sort
- Closest pair of points
- Convex Hull
- Traveling Salesman Problem
- Knapsack
- Assignment

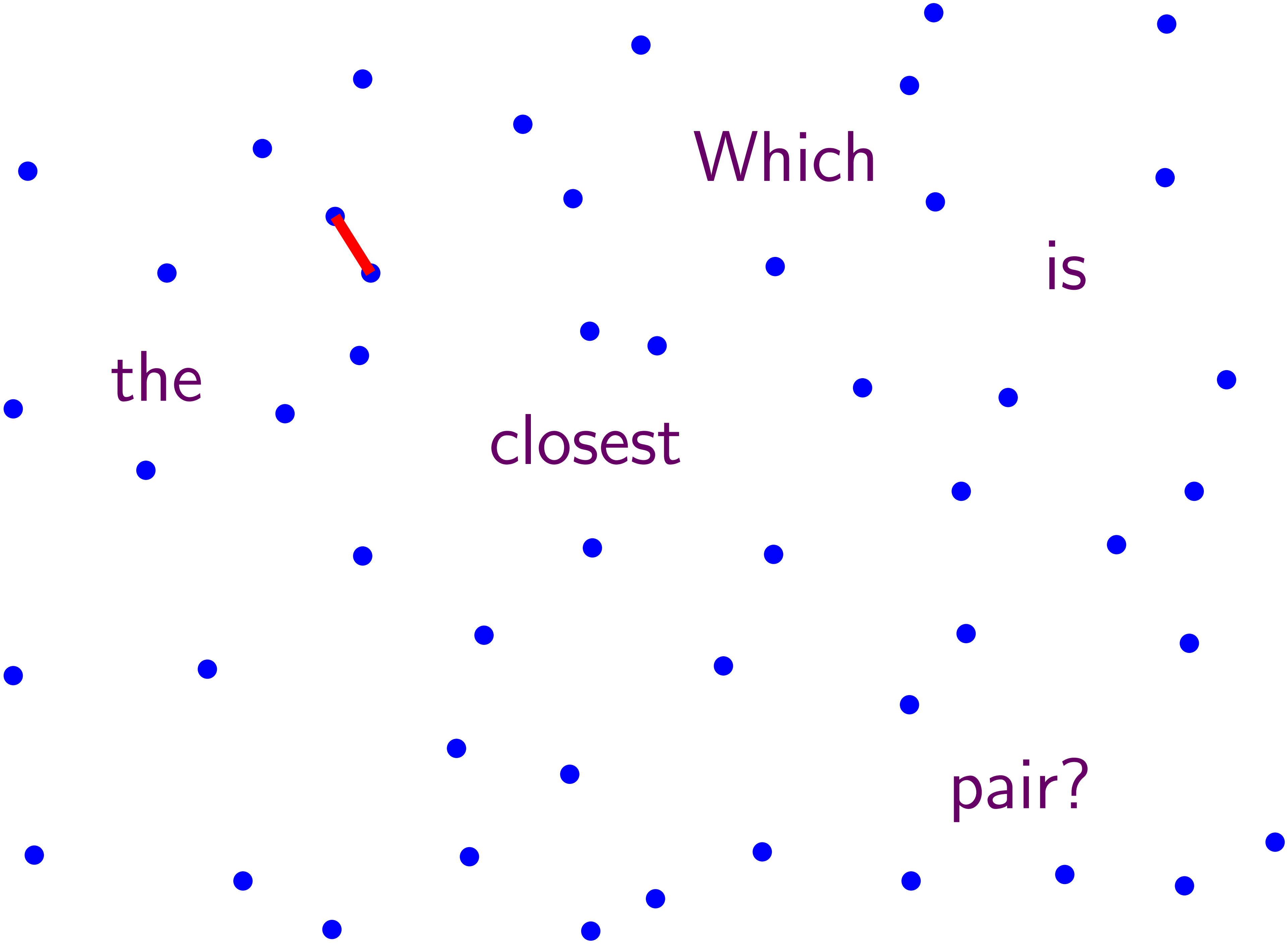
Problems

- Selection sort
- Insertion sort
- Closest pair of points
- Convex Hull
- Traveling Salesman Problem
- Knapsack
- Assignment

Closest Pair of Points



Closest Pair of Points



Closest Pair of Points

Problem:

Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$

Output: Distance between the closest pair of points $\{p, q\} \subseteq P$

Closest Pair of Points

Problem:

Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$

Output: Distance between the closest pair of points $\{p, q\} \subseteq P$

Definition: Distance between p_i and p_j

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

Algorithm

Algorithm

Algorithm ClosestPair (P)

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n - 1$ do

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n - 1$ do
3.   for $j \leftarrow i + 1$ to n do

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n - 1$ do
3. for $j \leftarrow i + 1$ to n do
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n - 1$ do
3. for $j \leftarrow i + 1$ to n do
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
5. return d

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n - 1$ do $\qquad \sum_{i=1}^{n-1}$
3. for $j \leftarrow i + 1$ to n do
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
5. return d

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n-1$ do $\qquad \sum_{i=1}^{n-1}$
3.   for $j \leftarrow i+1$ to n do $\qquad \sum_{j=i+1}^{n}$
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
5. return d

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n-1$ do $\quad \sum_{i=1}^{n-1}$
3. for $j \leftarrow i+1$ to n do $\quad \sum_{j=i+1}^{n}$
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ 2
5. return d

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n - 1$ do $\sum_{i=1}^{n-1}$
3. for $j \leftarrow i + 1$ to n do $\sum_{j=i+1}^{n}$
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ 2
5. return d

$(n-(i+1)+1)2$

$(n - i)2$

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n - 1$ do $\quad \sum_{i=1}^{n-1}$
3. for $j \leftarrow i + 1$ to n do $\quad 2(n - i)$
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
5. return d

Algorithm

Algorithm ClosestPair (P)

- Input: A list P of n points $p_1(x_1, y_1), \ldots p_n(x_n, y_n)$
- Output: The distance between the closest pair of points

1. $d \leftarrow \infty$
2. for $i \leftarrow 1$ to $n-1$ do $\quad \sum_{i=1}^{n-1}$
3. for $j \leftarrow i+1$ to n do $\quad 2(n-i)$
4. $d \leftarrow \min(d, \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
5. return d

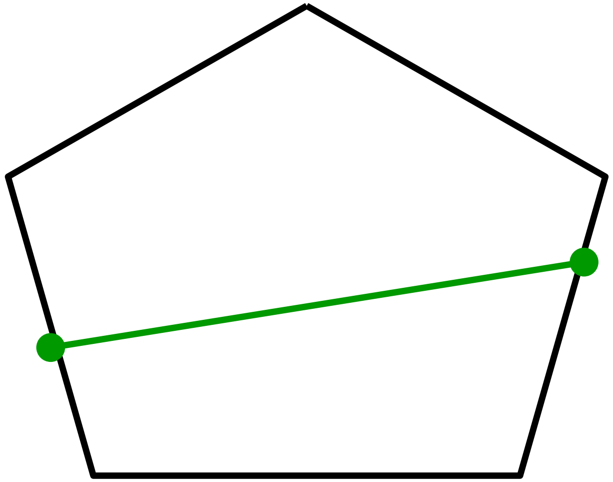
$$= 2((n-1) + \ldots + 1) = 2\frac{(n-1)n}{2} = (n-1)n = n^2 - n = O(n^2)$$

Convex Set

Convex set: A set of points in the plane is called convex if for any two points p and q in the set, the entire line segment with the endpoints at p and q belongs to the set.

Convex Set

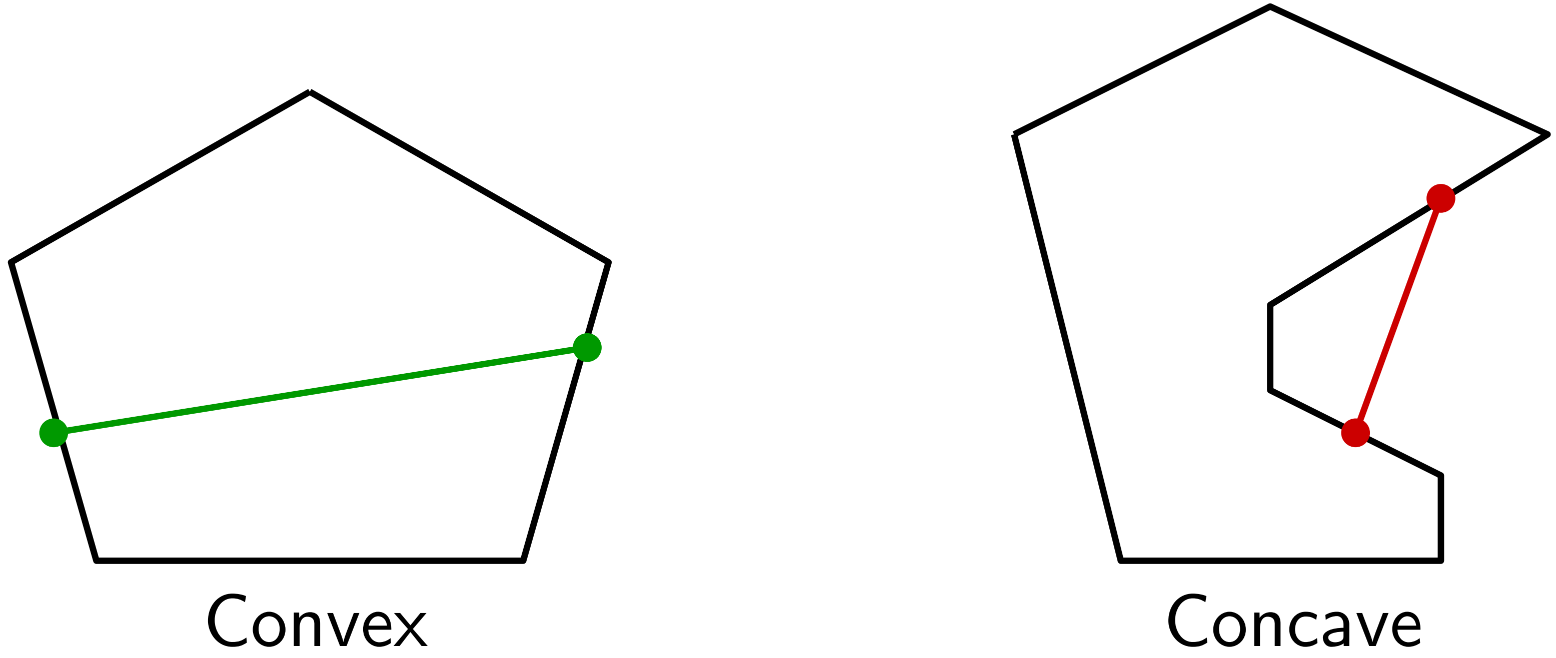
Convex set: A set of points in the plane is called convex if for any two points p and q in the set, the entire line segment with the endpoints at p and q belongs to the set.



Convex

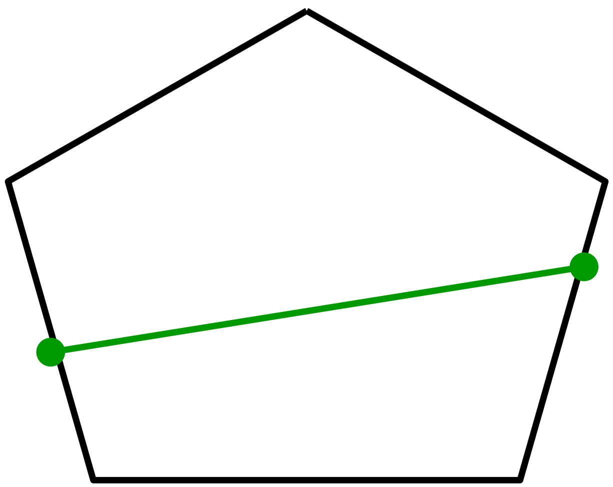
Convex Set

Convex set: A set of points in the plane is called convex if for any two points p and q in the set, the entire line segment with the endpoints at p and q belongs to the set.

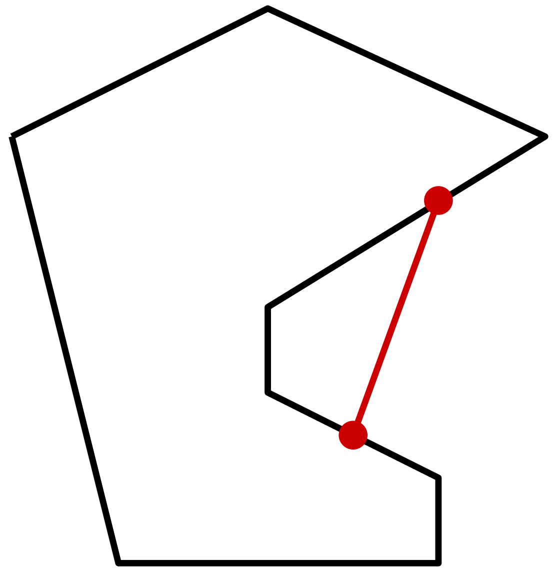


Convex Set

Convex set: A set of points in the plane is called convex if for any two points p and q in the set, the entire line segment with the endpoints at p and q belongs to the set.



Convex



Concave

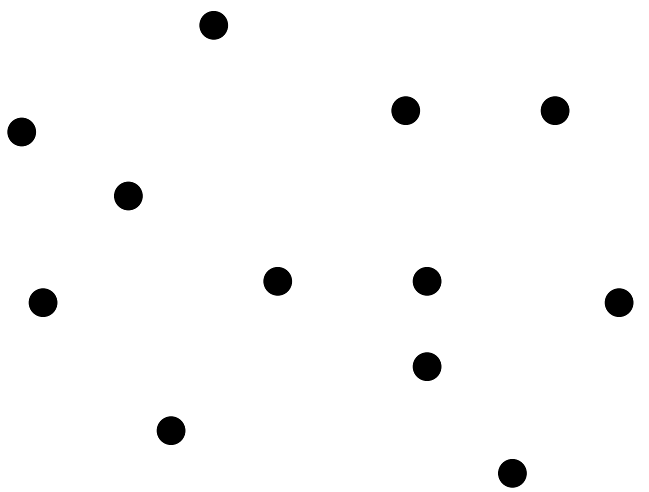
- Straight line
- Triangle
- Rectangle
- Any convex polygon

Convex Hull

Convex hull: The convex hull of a set of n points in the plane is the smallest convex polygon that contains all of them either inside or on its boundary.

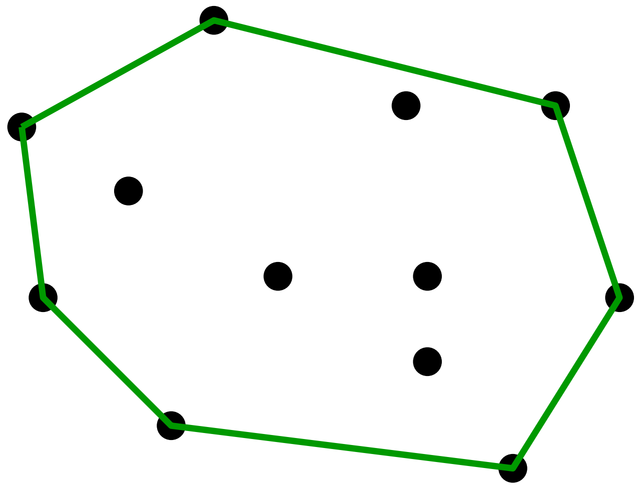
Convex Hull

Convex hull: The convex hull of a set of n points in the plane is the smallest convex polygon that contains all of them either inside or on its boundary.



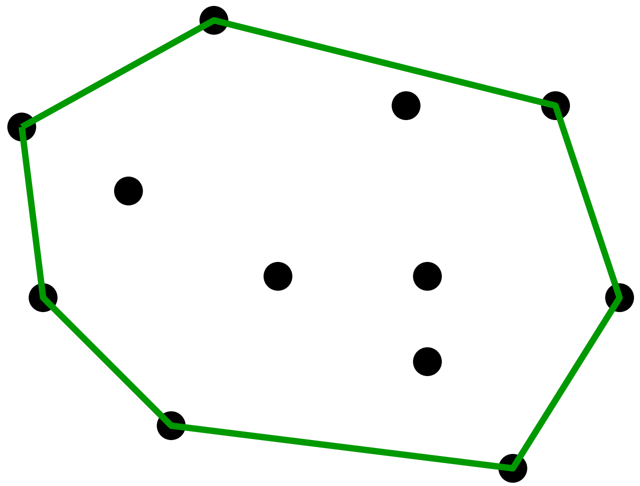
Convex Hull

Convex hull: The convex hull of a set of n points in the plane is the smallest convex polygon that contains all of them either inside or on its boundary.



Convex Hull

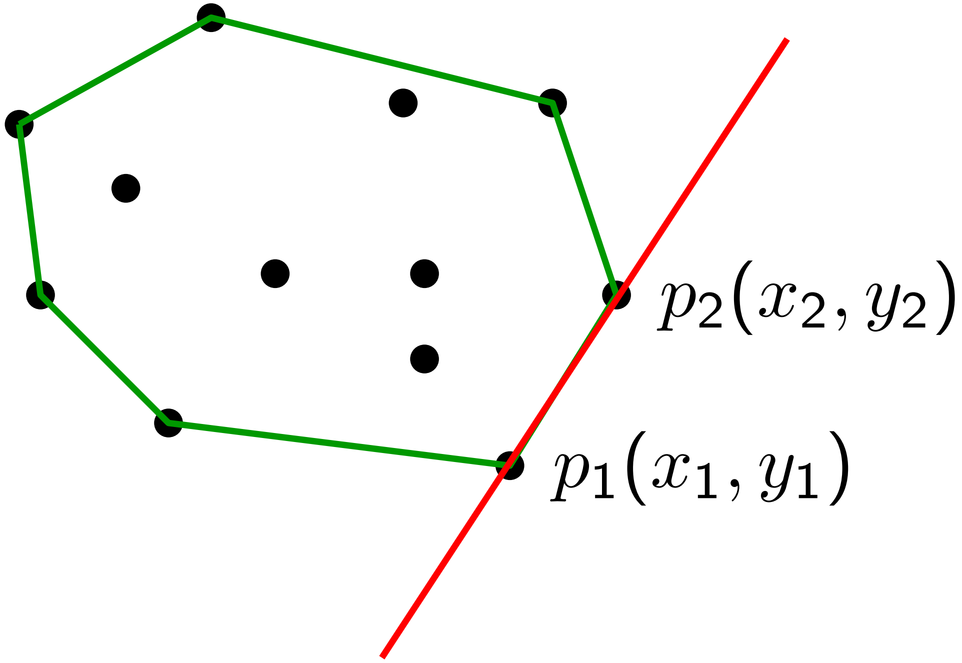
Convex hull: The convex hull of a set of n points in the plane is the smallest convex polygon that contains all of them either inside or on its boundary.



Convex hull(formal): The convex hull of a set S of points is the smallest convex set containing S.

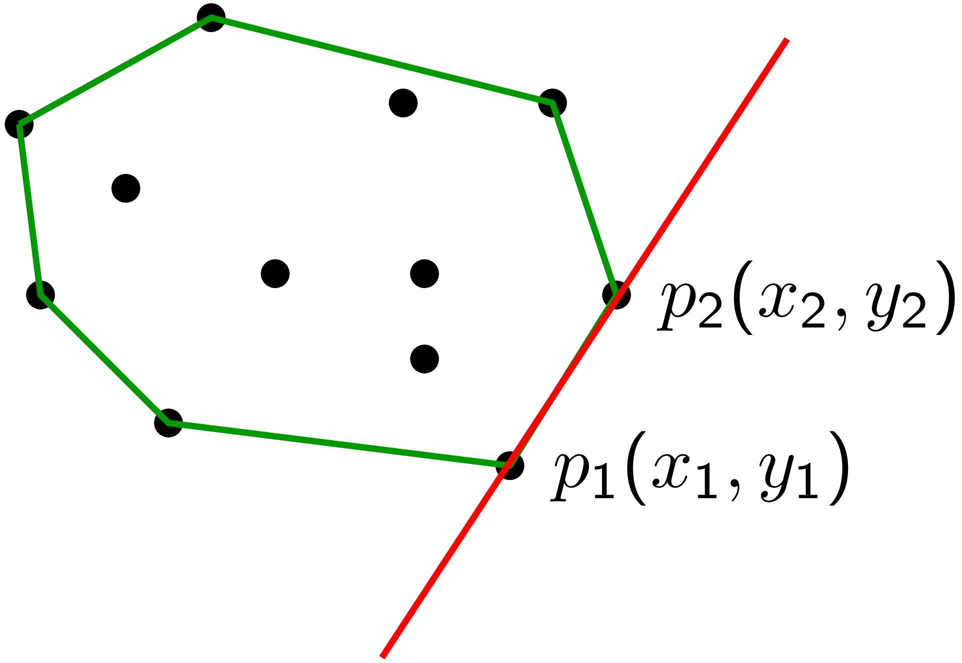
Algorithm

A line segment connecting two points p_1 and p_2 of a set of n points is a part of the convex hull's boundary if and only if all the other points of the set lie on the same side of the straight line through these two points.



Algorithm

A line segment connecting two points p_1 and p_2 of a set of n points is a part of the convex hull's boundary if and only if all the other points of the set lie on the same side of the straight line through these two points.



Striaght line with two points $(x_1, y_1), (x_2, y_2)$

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

$$ax + by = c$$

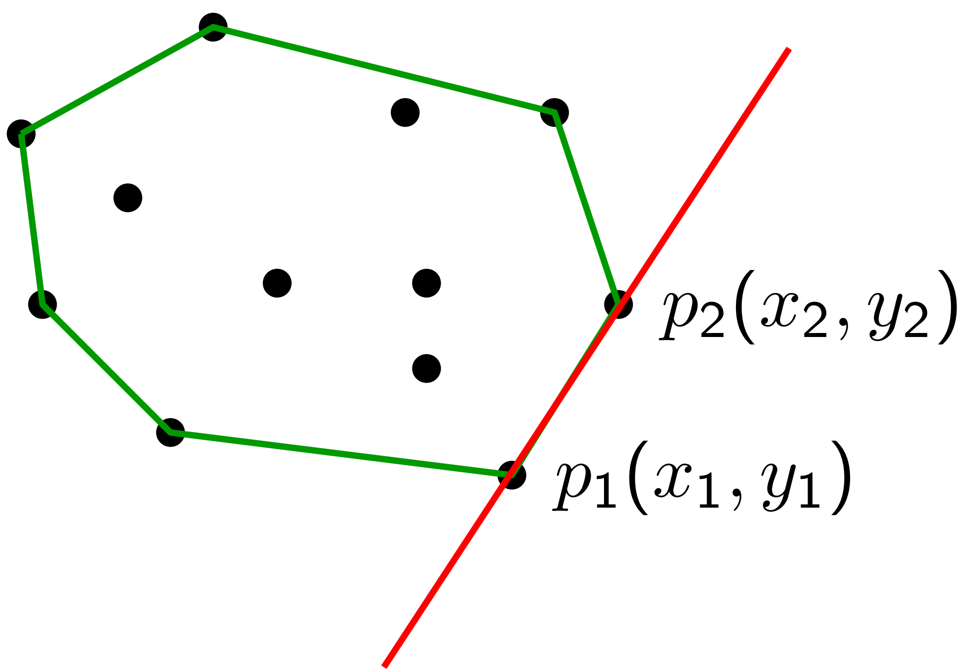
$$a = y_2 - y_1$$

$$b = x_1 - x_2$$

$$c = x_1 y_2 - y_1 x_2$$

Algorithm

A line segment connecting two points p_1 and p_2 of a set of n points is a part of the convex hull's boundary if and only if all the other points of the set lie on the same side of the straight line through these two points.



Striaght line with two points $(x_1, y_1), (x_2, y_2)$

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

$$ax + by = c$$

$$a = y_2 - y_1$$

$$b = x_1 - x_2$$

$$c = x_1 y_2 - y_1 x_2$$

Do certain points lie on the same side of the line?
$\equiv$ Does $ax + by - c$ have the same sign for each of these points?

Analysis

- There are $n\frac{(n-1)}{2}$ pairs of distinct points.

Analysis

- There are $n\frac{(n-1)}{2}$ pairs of distinct points.
- For each of these pairs, find the sign of $ax + by - c$ for each of the other $n - 2$ points.

Analysis

- There are $n\frac{(n-1)}{2}$ pairs of distinct points.
- For each of these pairs, find the sign of $ax + by - c$ for each of the other $n - 2$ points.
- No of checks $= n\frac{(n-1)}{2}(n-2) = \frac{n^3}{2} - \frac{3n^2}{2} + n$

Analysis

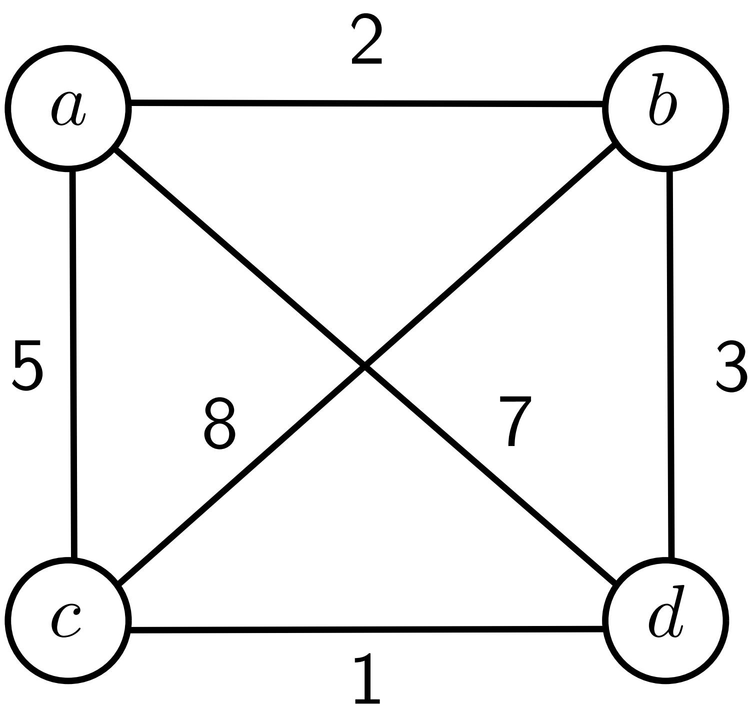
- There are $n\frac{(n-1)}{2}$ pairs of distinct points.
- For each of these pairs, find the sign of $ax + by - c$ for each of the other $n - 2$ points.
- No of checks $= n\frac{(n-1)}{2}(n-2) = \frac{n^3}{2} - \frac{3n^2}{2} + n$
- Running time $= O(n^3)$

Traveling Salesman Problem

TSP: Find the shortest tour through a given set of n cities that visits each city exactly once before returning to the city where it started.

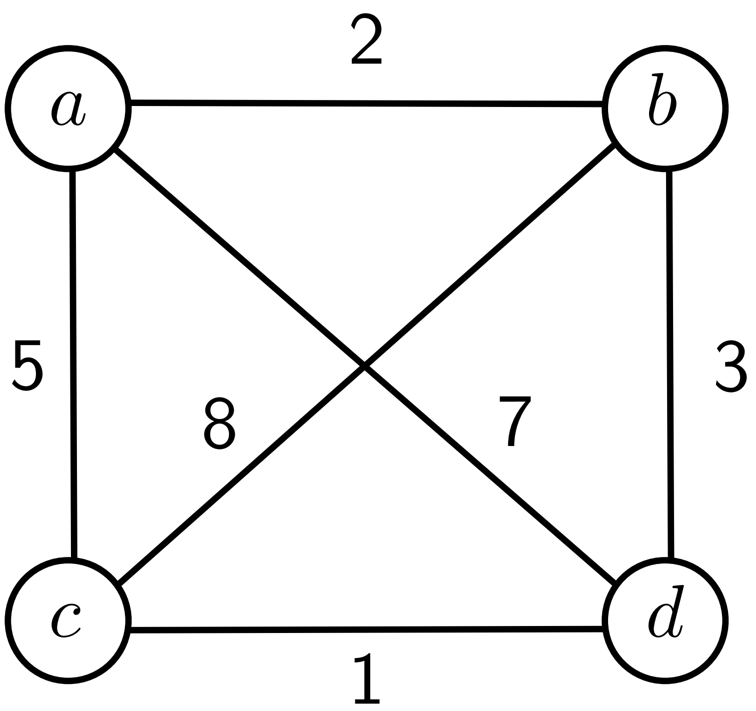
Traveling Salesman Problem

TSP: Find the shortest tour through a given set of n cities that visits each city exactly once before returning to the city where it started.



Traveling Salesman Problem

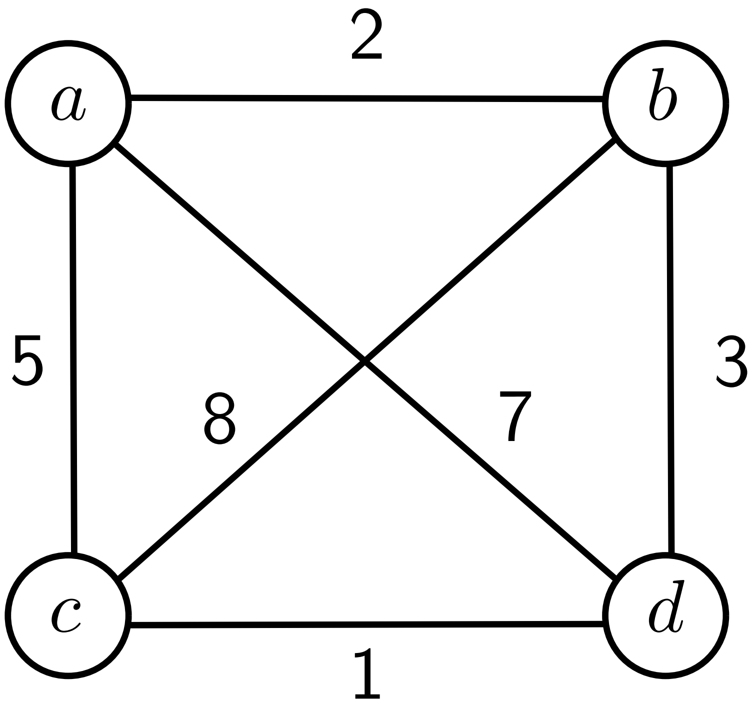
TSP: Find the shortest tour through a given set of n cities that visits each city exactly once before returning to the city where it started.



- Weighted graph
- Vertices are cities
- Edge weights are distances

Traveling Salesman Problem

TSP: Find the shortest tour through a given set of n cities that visits each city exactly once before returning to the city where it started.

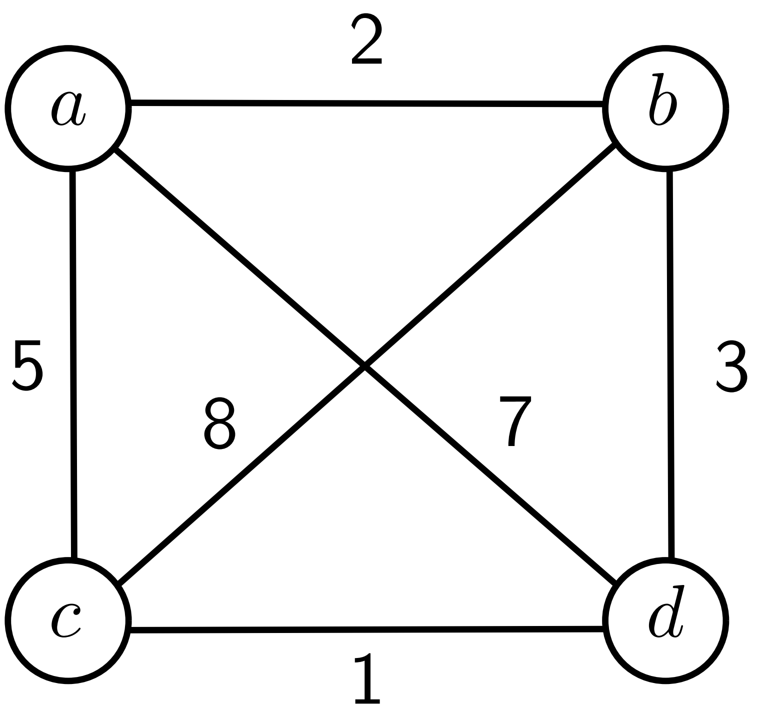


- Weighted graph
- Vertices are cities
- Edge weights are distances

A Hamiltonian circuit is a cycle that passes through all the vertices of the graph exactly once.

Traveling Salesman Problem

TSP: Find the shortest tour through a given set of n cities that visits each city exactly once before returning to the city where it started.



- Weighted graph
- Vertices are cities
- Edge weights are distances

A Hamiltonian circuit is a cycle that passes through all the vertices of the graph exactly once.

TSP: Find the shortest Hamiltonian circuit of the graph.

Hamiltonian Circuit

- A sequence of $n+1$ adjacent vertices $v_{i_0}, v_{i_1}, \ldots, v_{i_n-1}, v_{i_0}$.
- First vertex and the last vertex are the same.
- The other $n-1$ vertices are distinct.

Hamiltonian Circuit

- A sequence of $n + 1$ adjacent vertices $v_{i_0}, v_{i_1}, \ldots, v_{i_n - 1}, v_{i_0}$.
- First vertex and the last vertex are the same.
- The other $n - 1$ vertices are distinct.

Choose one particular vertex as the start and end for all circuits.

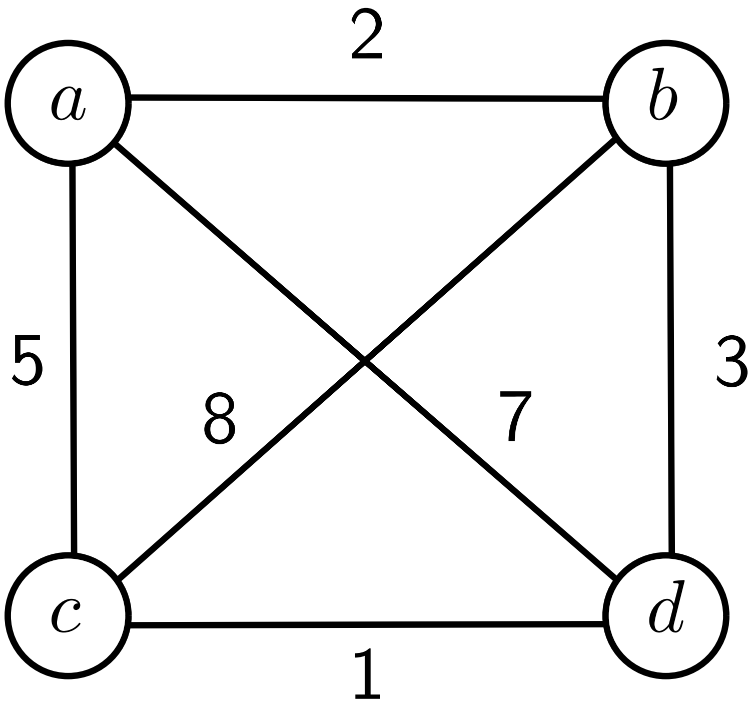
Hamiltonian Circuit

- A sequence of $n+1$ adjacent vertices $v_{i_0}, v_{i_1}, \ldots, v_{i_{n-1}}, v_{i_0}$.
- First vertex and the last vertex are the same.
- The other $n-1$ vertices are distinct.

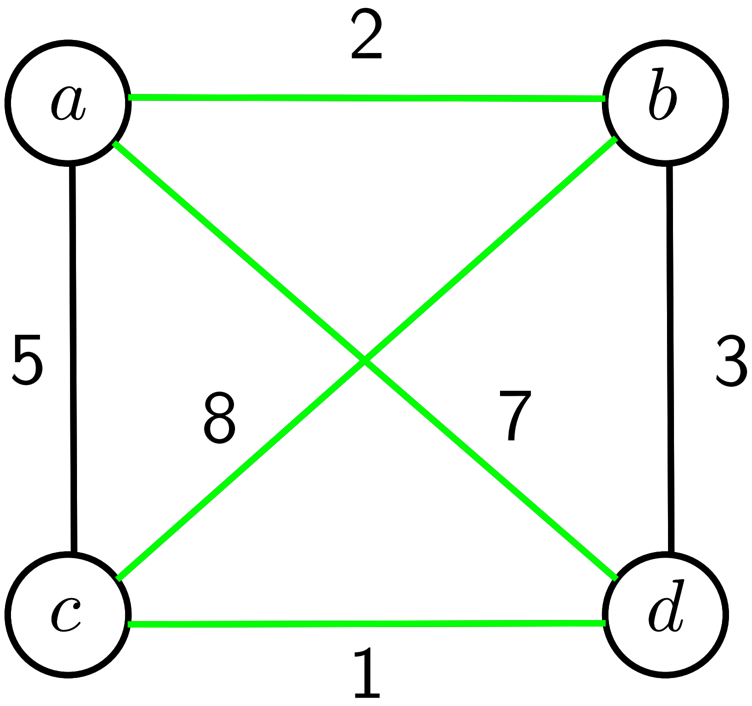
Choose one particular vertex as the start and end for all circuits.

1. Generate all the permutations of $n-1$ intermediate cities,
2. Compute the tour lengths,
3. Find the shortest among them.

Example

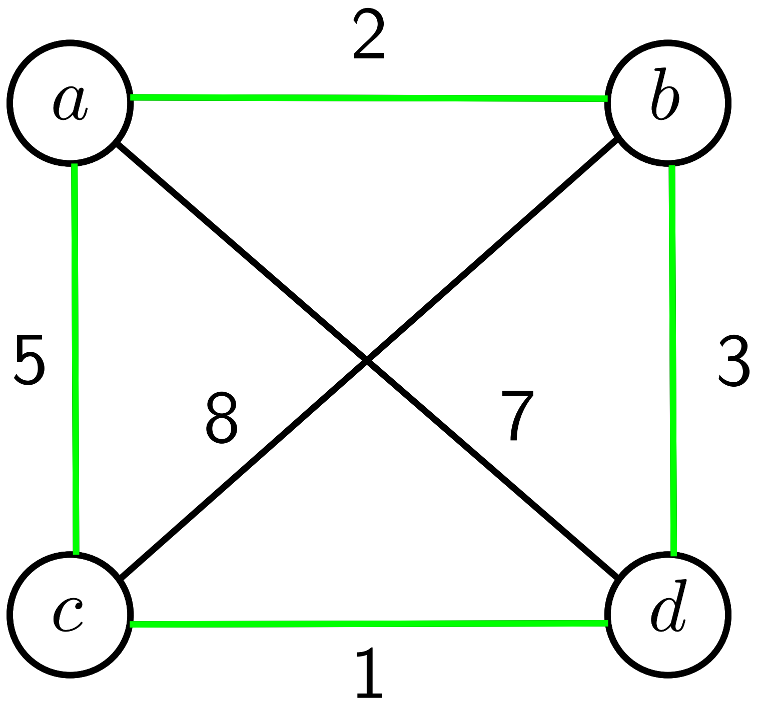


Example



$a \rightarrow b \rightarrow c \rightarrow d \rightarrow a \qquad 2 + 8 + 1 + 7 = 18$

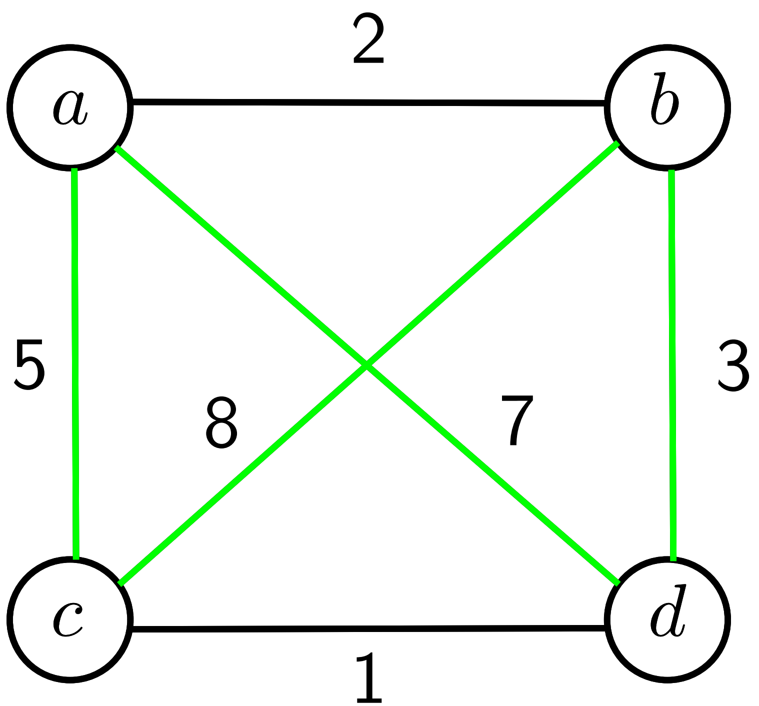
Example



$$a \to b \to c \to d \to a \qquad 2 + 8 + 1 + 7 = 18$$

$$a \to b \to d \to c \to a \qquad 2 + 3 + 1 + 5 = 11$$

Example

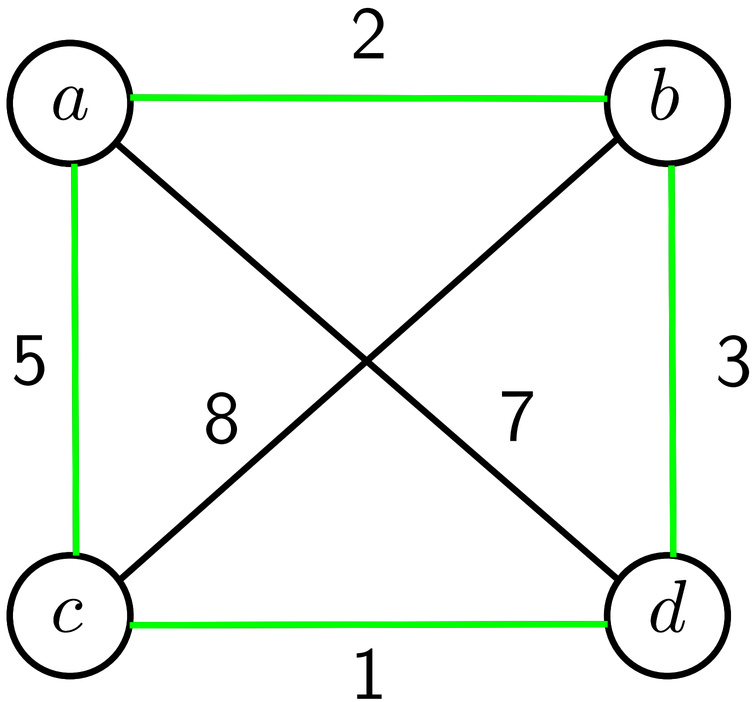


$$a \to b \to c \to d \to a \qquad 2 + 8 + 1 + 7 = 18$$

$$a \to b \to d \to c \to a \qquad 2 + 3 + 1 + 5 = 11$$

$$a \to c \to b \to d \to a \qquad 5 + 8 + 3 + 7 = 23$$

Example



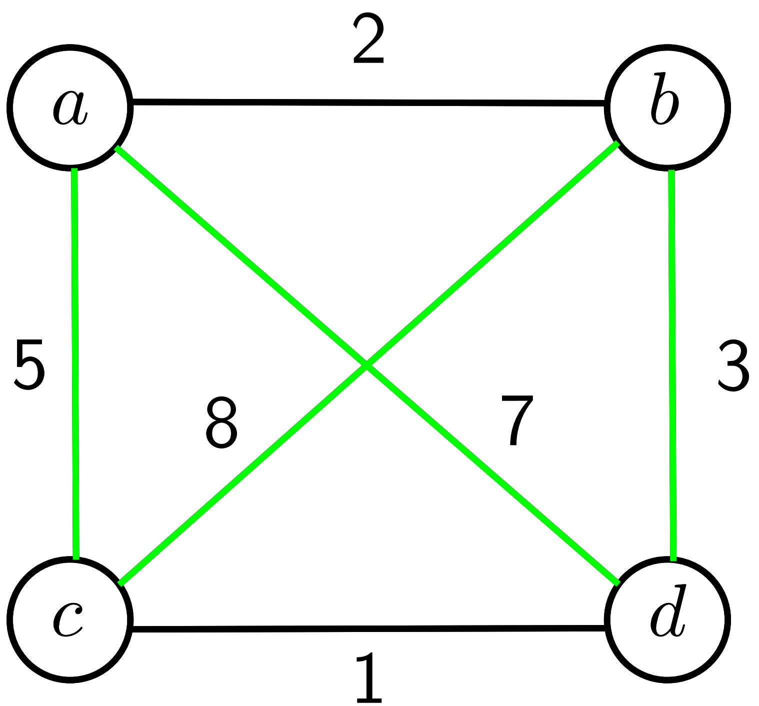
$$a \to b \to c \to d \to a \qquad 2 + 8 + 1 + 7 = 18$$

$$a \to b \to d \to c \to a \qquad 2 + 3 + 1 + 5 = 11$$

$$a \to c \to b \to d \to a \qquad 5 + 8 + 3 + 7 = 23$$

$$a \to c \to d \to b \to a \qquad 5 + 1 + 3 + 2 = 11$$

Example



$$a \to b \to c \to d \to a \qquad 2+8+1+7=18$$

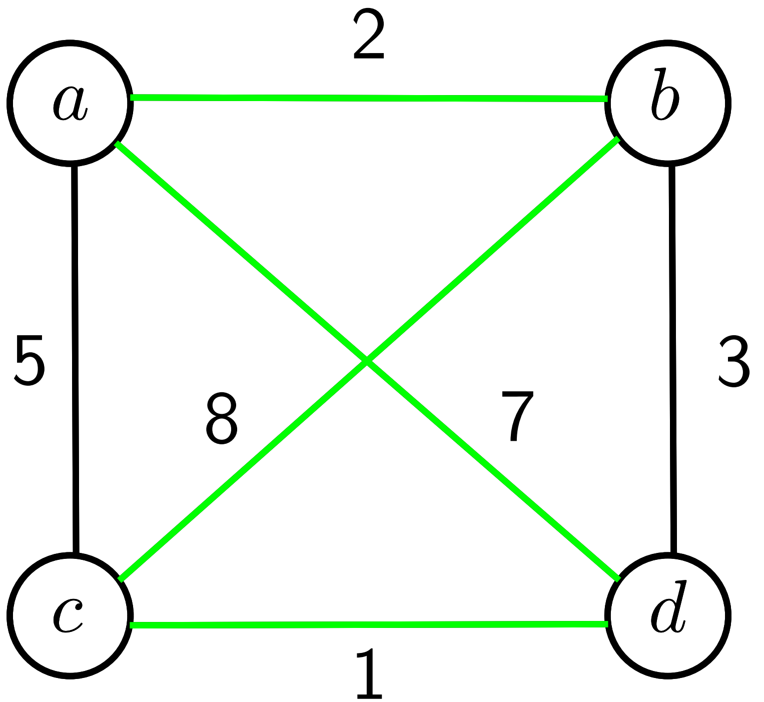
$$a \to b \to d \to c \to a \qquad 2+3+1+5=11$$

$$a \to c \to b \to d \to a \qquad 5+8+3+7=23$$

$$a \to c \to d \to b \to a \qquad 5+1+3+2=11$$

$$a \to d \to b \to c \to a \qquad 7+3+8+5=23$$

Example



$a \to b \to c \to d \to a$ $\quad 2 + 8 + 1 + 7 = 18$

$a \to b \to d \to c \to a$ $\quad 2 + 3 + 1 + 5 = 11$

$a \to c \to b \to d \to a$ $\quad 5 + 8 + 3 + 7 = 23$

$a \to c \to d \to b \to a$ $\quad 5 + 1 + 3 + 2 = 11$

$a \to d \to b \to c \to a$ $\quad 7 + 3 + 8 + 5 = 23$

$a \to d \to c \to b \to a$ $\quad 7 + 1 + 8 + 2 = 18$

Analysis

1. Generate all the permutations of $n-1$ intermediate cities,
2. Compute the tour lengths,
3. Find the shortest among them.

Analysis

1. Generate all the permutations of $n-1$ intermediate cities,
2. Compute the tour lengths,
3. Find the shortest among them.

- $(n-1)!$ permutations of $n-1$ cities.
  Running time $= O(n!)$

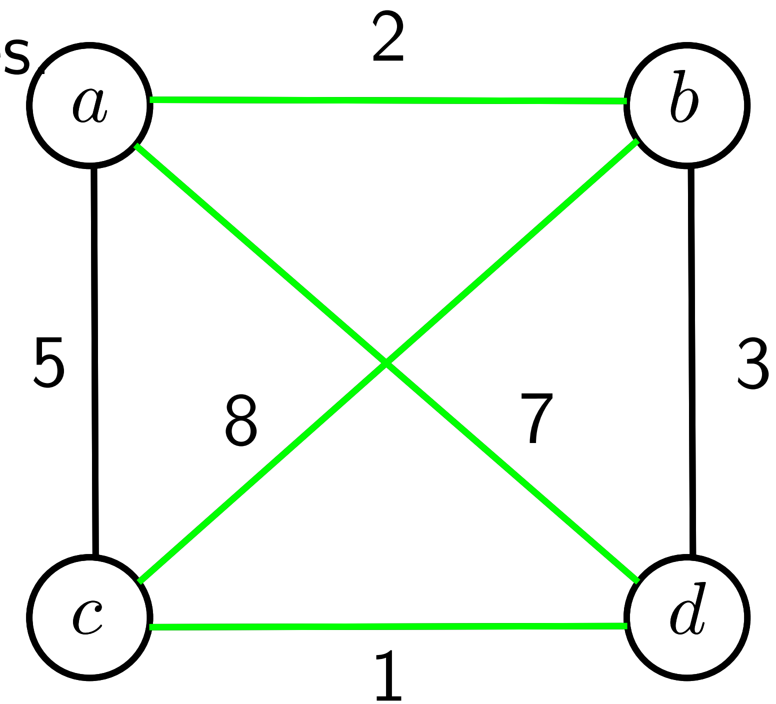
Analysis

1. Generate all the permutations of $n-1$ intermediate cities,
2. Compute the tour lengths,
3. Find the shortest among them.

- $(n-1)!$ permutations of $n-1$ cities.
  Running time $= O(n!)$

$a \to b \to c \to d \to a$
$2+8+1+7=18$

$a \to d \to c \to b \to a$
$7+1+8+2=18$



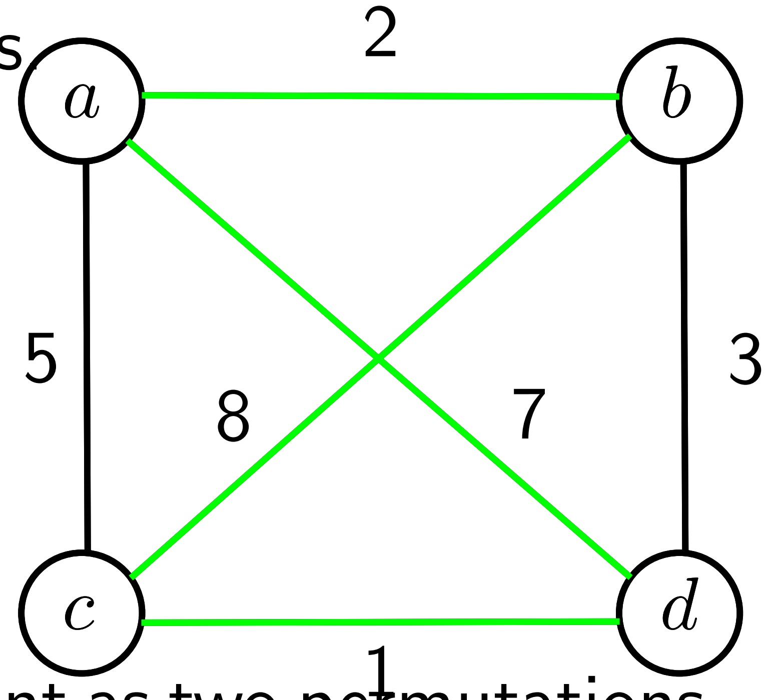
Analysis

1. Generate all the permutations of $n-1$ intermediate cities,
2. Compute the tour lengths,
3. Find the shortest among them.

- $(n-1)!$ permutations of $n-1$ cities.
  Running time $= O(n!)$

  $a \to b \to c \to d \to a$
  $2 + 8 + 1 + 7 = 18$

  $a \to d \to c \to b \to a$
  $7 + 1 + 8 + 2 = 18$



- Each permutation and its reverse count as two permutations.
- $\frac{1}{2}(n-1)!$ permutations of $n-1$ cities.
  But running time $= O(n!)$

Questions!

- State the characteristics of brute force strategy.

Questions!

- State the characteristics of brute force strategy.
- If the problem size of for finding the closest pair is n, what is the size of the state space?

Questions!

- State the characteristics of brute force strategy.
- If the problem size of for finding the closest pair is n, what is the size of the state space?
- What is the running time of brute force algorithm for finding the convex hull?

Questions!

- State the characteristics of brute force strategy.
- If the problem size of for finding the closest pair is n, what is the size of the state space?
- What is the running time of brute force algorithm for finding the convex hull?
- What is Hamiltonian circuit?

Questions!

- State the characteristics of brute force strategy.
- If the problem size of for finding the closest pair is n, what is the size of the state space?
- What is the running time of brute force algorithm for finding the convex hull?
- What is Hamiltonian circuit?
- Generally brute force algorithms are not practical. Why?

Thank you.