Exhaustive Search:

Definition:

A brute force solution to a problem involving search for an element with a special property, usually among combinatorial objects such as a permutation, combination or subsets of a set is turned as exhaustive search.

Ex:

1. Travelling Salesman problem
2. knapsack problem
3. Assignment problem.

Travelling Salesman problem

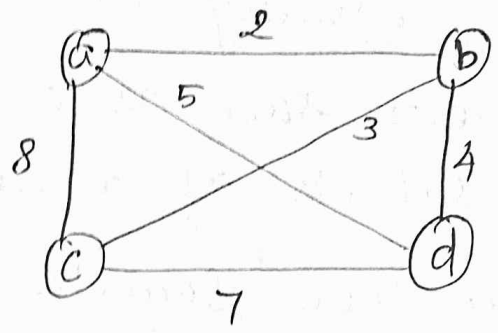
Problem Statement:

→ Given 'n' cities with known distance between each pair

→ Find the shortest tour that passes through all the cities exactly once before returning to the starting city

→ Find shortest Hamiltonian circuit in a weighted connected graph.

Ex:



Sol:-

Tour:	cost
a – b → c – d – a	2+3+7+5 = 17 optimal
a – b – d – c – a	2+4+7+8 = 21
a – c – b – d – a	8+3+4+5 = 20
a – c – d – b – a	8+7+4+2 = 21
a – d – b – c – a	5+4+3+8 = 20
a – d – c – b – a.	5+7+3+2 = 17 optimal

∴ The Time complexity of this alg is $O((n-1)!) \Rightarrow ((4-1)!) = 3! = 6$ path computati

Knapsack problems:

Problem Statement:

→ Given a knapsack with maximum capacity 'W', and a set 'S' consisting of 'n' items.

→ Each item 'i' has some weight w_i and benefit value v_i.

→ The knapsack has to be packed to achieve maximum total value.

Ex:

Item	weight	Value
1	2	$20
2	5	$30
3	10	$50
4	5	$10

Knapsack capacity W=16.

Sol:

→ solve this problem with a straightforward algorithm.

→ we go through all combinations (subsets) & find the one with maximum value and with weight less or equal to W=16.

Subset	Total weight	Total Value.	
{1}	2	$20	
{2}	5	$30	
{3}	10	$50	
{4}	5	$10	
{1,2}	7	$50	
{1,3}	12	$70	
{1,4}	9	$30.	
{1,2,3}	17	$100	Not feasible
{1,2,4}	12	$60	
{1,3,4}	17	$80.	Not feasible
{2,3}	15	$80	
{2,4}	10	$40	
{3,4}	15	$60	
{1,2,3,4}.	22	$110	Not feasible.

Efficiency:

→ Since there are 'n' items, there are 2^n possible combinations of items.

→ The running time will be $O(2^n)$.

Assignment problem.

Problem Statement

→ There are n people who need to be assigned to n jobs, one persons per job. The cost of assigning "person i to job j" is C[i, j].

→ Find an assignment that minimizes the total cost.

Ex1:

C[i,j]	Job A(1)	Job B(2)	Job C(3)	Job D(4)
person A(1)	9	2	7	8
" B(2)	6	4	3	7
" C(3)	5	8	1	8
" D(4)	7	6	9	4

Assignment (col, I)	Total cost.
1, 2, 3, 4	9+4+1+4 = 18
1, 2, 4, 3	9+4+8+9 = 30
1, 3, 2, 4	9+3+8+4 = 24
1, 3, 4, 2	9+3+8+6 = 26
1, 4, 3, 2	9+7+1+6 = 23
1, 4, 2, 3.	9+7+8+9 = 26

Assignment (col, II)	Total cost.
2, 1, 3, 4	2+6+1+4 = 13. optimal.
2, 1, 4, 3	2+6+8+9 = 25
2, 3, 1, 4	2+3+5+4 = 14
2, 3, 4, 1	2+3+8+7 = 20.
2, 4, 1, 3	2+7+5+9 = 23
2, 4, 3, 1	2+7+1+7 = 17

Assignment (col, III)	Total cost.
3, 1, 2, 4	7+6+8+4 = 25
3, 1, 4, 2	7+6+8+6 = 27
3, 2, 1, 4	7+4+5+4 = 20
3, 2, 4, 1	7+4+8+9 = 28
3, 4, 1, 2	7+7+5+6 = 25
3, 4, 2, 1	7+7+8+7 = 27.

Assignment (col, IV)	
4, 1, 2, 3	8+6+8+9 = 31
4, 1, 3, 2	8+6+17+6 = 21
4, 2, 1, 3	8+4+5+9 = 26
4, 2, 3, 1	8+4+11+7 = 20
4, 3, 1, 2	8+3+5+6 = 22
4, 3, 2, 1	8+3+8+7 = 26.