# ARTIFICIAL INTELLIGENCE

**Unit – 3 Heuristic Search Techniques:** Generate & Test, Hill Climbing, Best First Search, Problem Reduction, Constraint Satisfaction, Means-Ends Analysis

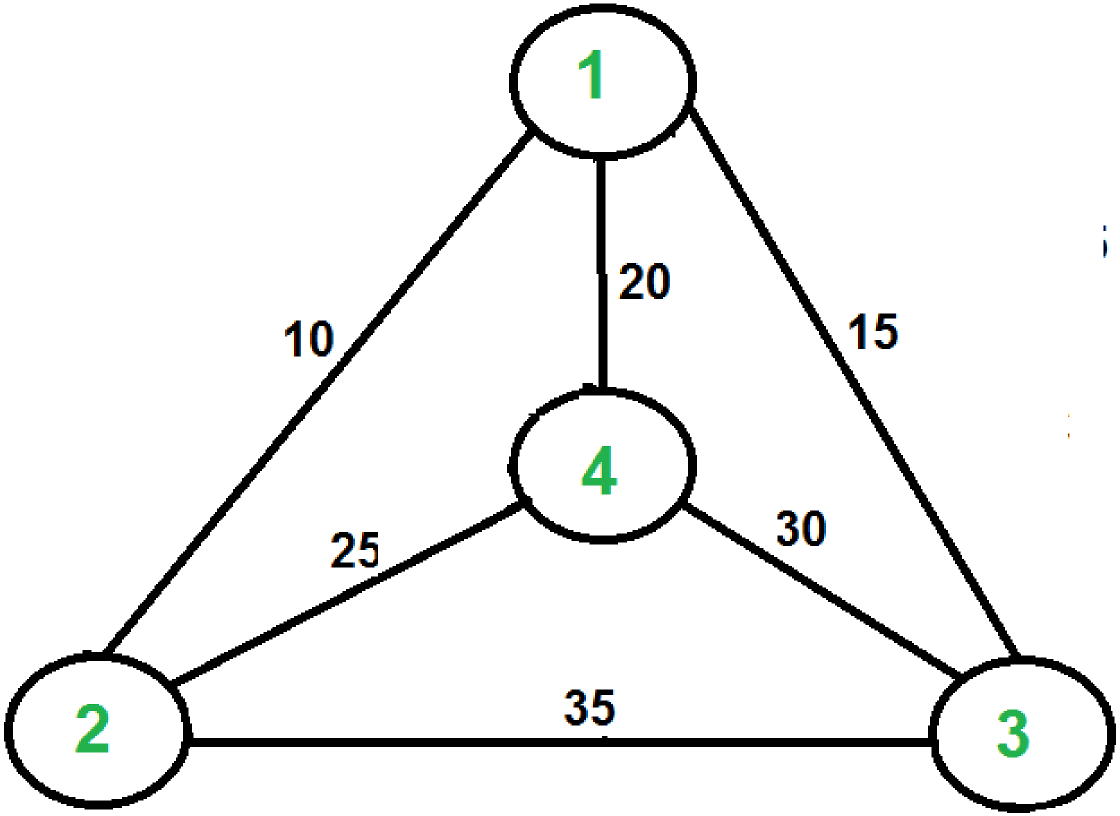
# OUTLINE

- Introduction
- Generate-and-test
- Hill climbing
- Best-first search
- Problem reduction
- Constraint satisfaction
- Means-ends analysis

# TRAVELING SALESMAN PROBLEM

- A salesman has a list of cities, each of which he must visit exactly once.
- There are direct roads between each pair of cities on the list. Find the route the salesman should follow for the shortest possible round trip that both starts and finishes at any one of the cities.
- A simple, motion-causing and systematic control structure could, in principle, solve this problem.
- It would simply explore all possible paths in the tree and return the one with shortest length.
- But this approach breaks down quickly as the number of cities grows.

# TRAVELING SALESMAN PROBLEM



# TRAVELING SALESMAN PROBLEM

- If there are N cities, then the number of different paths among them is 1.2…..(N-1) or (N-1)!
- The time to examine a single path is proportional to N. So the total time required to perform this search is proportional to N.
- Assuming there are only 10 cities, 10! = 36,28,800, which is a very large number.
- The salesman have easily have 25 cities to visit. To solve this problem would take more time than he would be willing to spend.
- This phenomenon is called combinatorial explosion.
- It can be solved using *branch-and-bound* technique, but it still requires exponential time.
- It can be solved by Heuristic technique.

# SEARCH STRATEGIES: HEURISTIC SEARCH

- Heuristic: involving or serving as an aid to learning, discovery, or problem-solving by experimental and especially trial-and-error methods.
  (Merriam-Webster's dictionary)
- Heuristic technique improves the efficiency of a search process, possibly by sacrificing claims of completeness or optimality.
- Heuristic is for combinatorial explosion.
- Optimal solutions are rarely needed.

# NEAREST NEIGHBOR HEURISTIC FOR TSP

1. Select a starting city.
2. Select the one closest to the current city.
3. Repeat step 2 until all cities have been visited.

- $O(n^2)$ vs. $O(n!)$

# TRAVELING SALESMAN PROBLEM

- A heuristic is a function that outlines from problem state description to measures desirability, usually represented as number weights. The value of a heuristic function at a given node in the search process gives a good estimate of that node being on the desired path to the solution. Well-designed heuristic functions can provide a fairly good estimate of whether a path is good or not.

- Let us say that the total of all the distances covered so far is a simple heuristic function in the traveling salesman problem. The main objective of a heuristic function is to guide the search process in the most profitable directions, by suggesting which path to follow first when more than one path is available.

# HEURISTIC FUNCTION

- state descriptions → measures of desirability

➢ Some Simple Heuristic Functions

  - TSP: the sum of the distances so far
  - Tic-Tac-Toe: 1 for each row in which we could win and in which we already have one piece plus 2 for each such row in which we have two pieces.

# GENERATE-AND-TEST

Heuristic technique, DFS with Backtracking

Informed search technique

Algorithm

1. Generate a possible solution.
2. Test to see if this is actually a solution.
3. Quit if a solution has been found.
   Otherwise, return to step 1.

# GENERATE-AND-TEST

- Acceptable for simple problems.
- Inefficient  for problems with large space.
- Exhaustive generate-and-test.
- Plan generate-test:
    - Create a list of candidates.
    - Apply generate-and-test to that list.

# GENERATE-AND-TEST

- Properties of Good Generators:

➢Complete

➢Not redundant

➢Informed

# GENERATE-AND-TEST

- Example: Assume a pin number I having, that is: 00 00 00
- Possible solutions are
  - 00 00 00
  - 00 00 01
  - 00 00 02
  - 00 00 03
  - -
  - -
  - 99 99 99
- Total (100) (100) (100)

= (100)^3 possible solutions= 1 Million

# GENERATE-AND-TEST

- Lets say to generate 5 solutions will takes 1 minute.
- So, in 60 minute will generate 300 solutions

Now if we have some knowledge about the problem
Like if the numbers are prime in the given pin 00 00 00
Now in between 0 to 100 there are 25 primes numbers.

So now,
(25) (25) (25)= (25)^3 near to 15000 possible solutions will be there.

Conclusion: If we are having some heuristic information about the problem then we come to the solution in less time.

# HILL CLIMBING

- Variant of Generate and Test
- Searching for a goal state = Climbing to the top of a hill
- Generate-and-test + direction to move.
- Heuristic function to estimate how close a given state is to a goal state.

# SIMPLE HILL CLIMBING

- Local search algorithm: knowledge of local domain only
- Greedy approach: When get best move then will stop to search
- No backtracking: If not get best move then will not do backtrack

# SIMPLE HILL CLIMBING

Algorithm

1. Evaluate the initial state. If it is a goal state, then return it and quit. Otherwise, continue with the initial state as the current state.

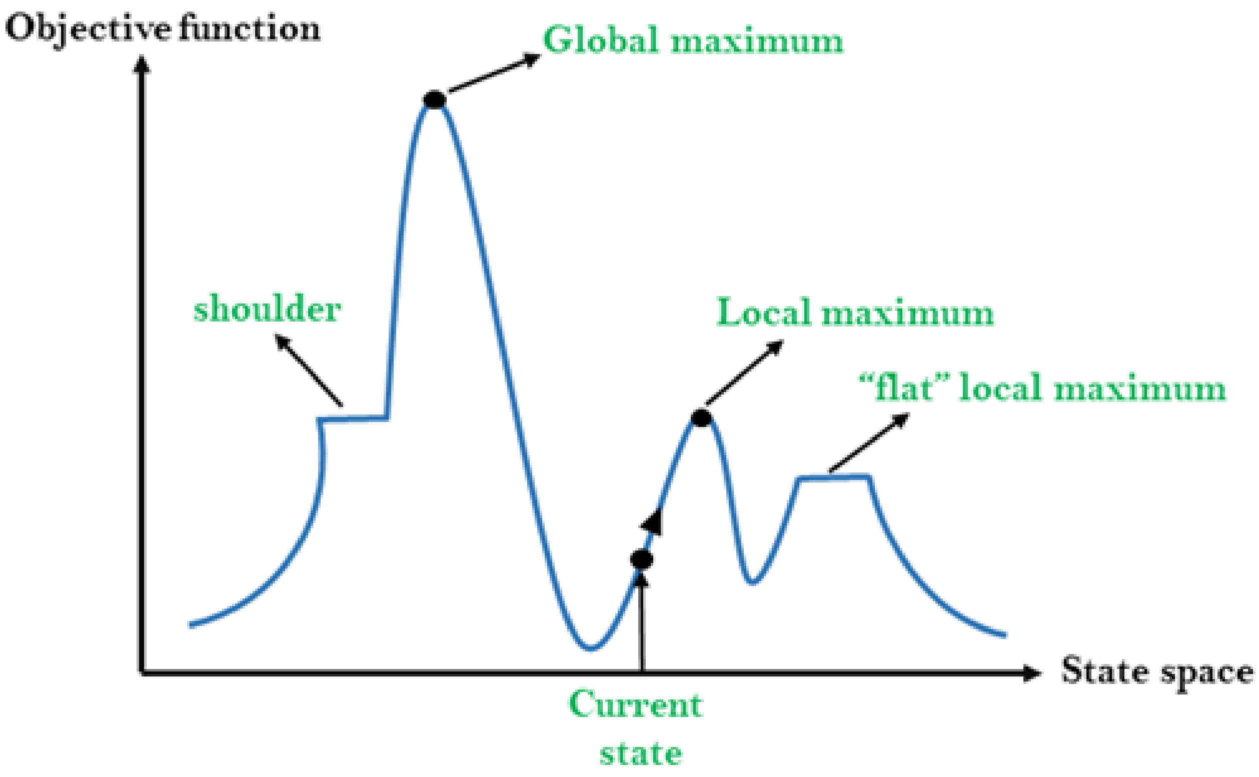
2. Loop until a solution is found or there are no new operators left to be applied:

   – Select and apply a new operator that has not yet been applied to current state

   – Evaluate the new state:

     - if goal → quit
     - if not goal but better than current state → new current state
     - if not better than current state then continue in loop.

# SIMPLE HILL CLIMBING:



# SIMPLE HILL CLIMBING

- Evaluation function as a way to inject task-specific knowledge into the control process.

# SIMPLE HILL CLIMBING

Example: coloured blocks

Heuristic function: the sum of the number of different colours on each of the four sides (solution = 16).

# SIMPLE HILL CLIMBING

- **Hill Climbing & puzzle of four colored block problem:-**

-Heuristic function: sum of no. of different colors on each of four sides=16.

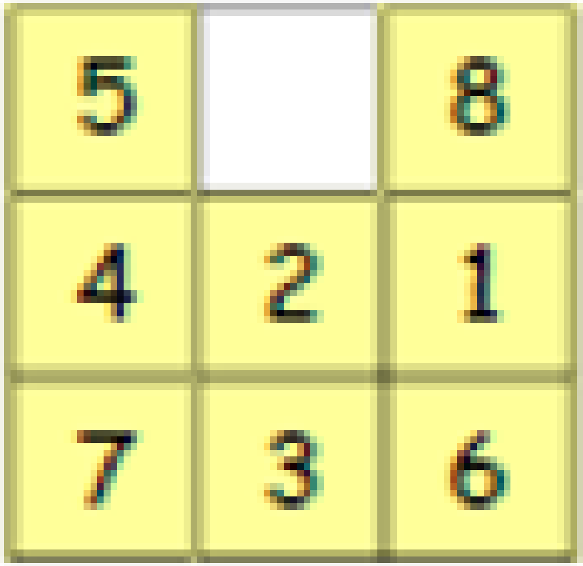
- Set of rules to make a configuration.

i) Pick a block

ii) Rotate 90 degree

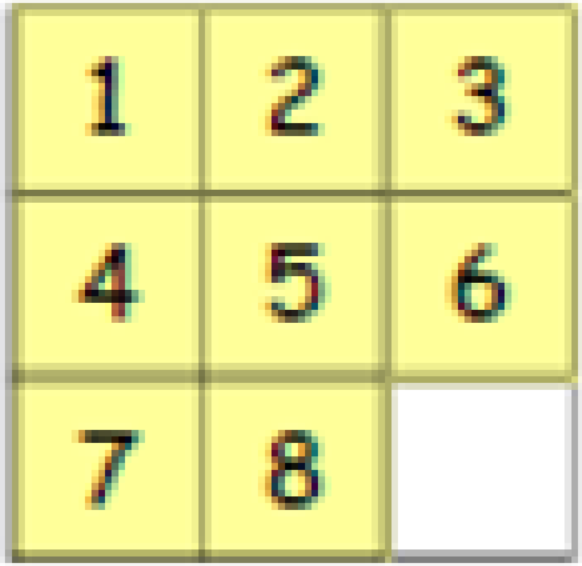
- Based on algorithm generate new state by selecting block and rotating. If result is better then keep otherwise move back and try with different perturbation.

# 8-PUZZLE PROBLEM



|   |   |   |
|---|---|---|
| 5 |   | 8 |
| 4 | 2 | 1 |
| 7 | 3 | 6 |

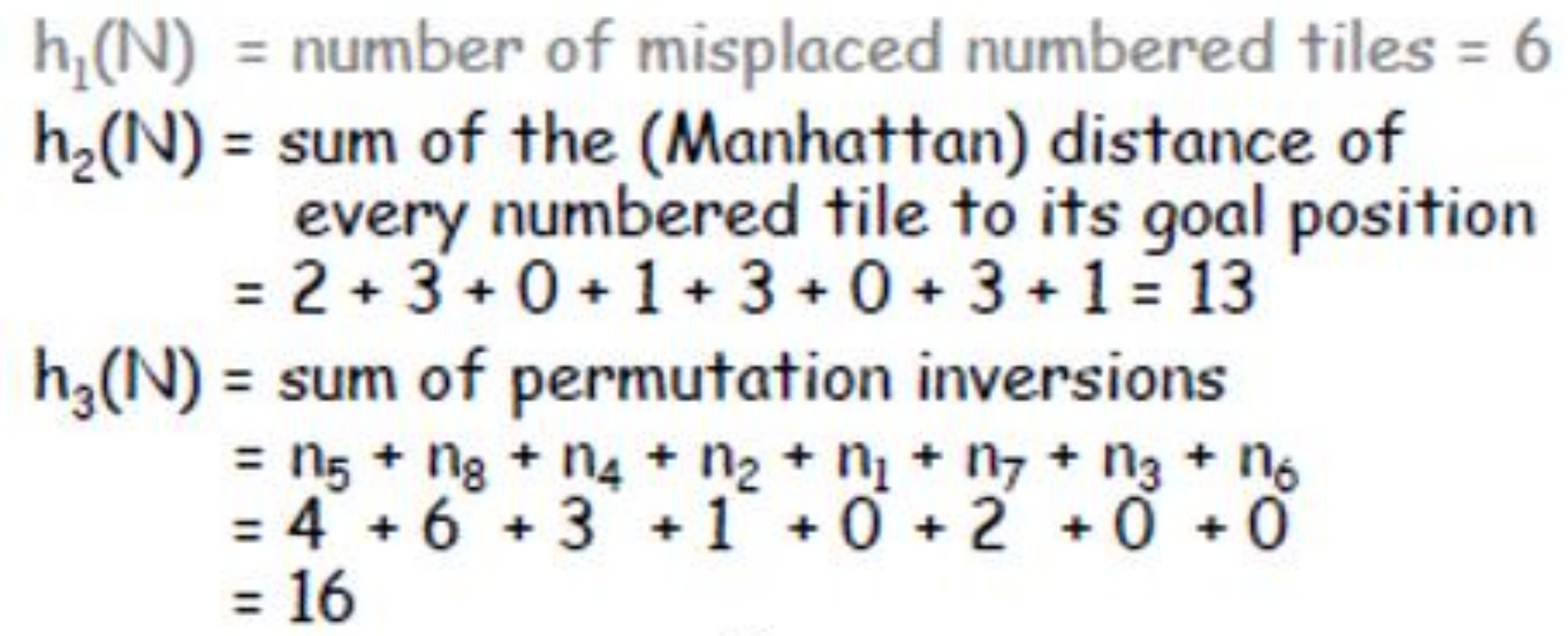
STATE(N)



|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 |   |

Goal state

# 8-PUZZLE PROBLEM



$h_1(N)$ = number of misplaced numbered tiles = 6

$h_2(N)$ = sum of the (Manhattan) distance of every numbered tile to its goal position

$= 2 + 3 + 0 + 1 + 3 + 0 + 3 + 1 = 13$

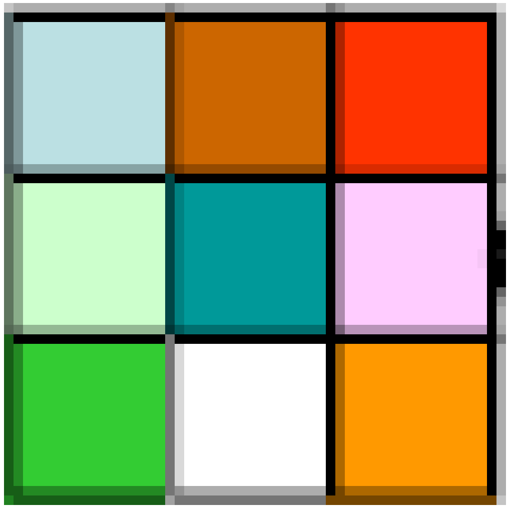
$h_3(N)$ = sum of permutation inversions

$= n_5 + n_8 + n_4 + n_2 + n_1 + n_7 + n_3 + n_6$

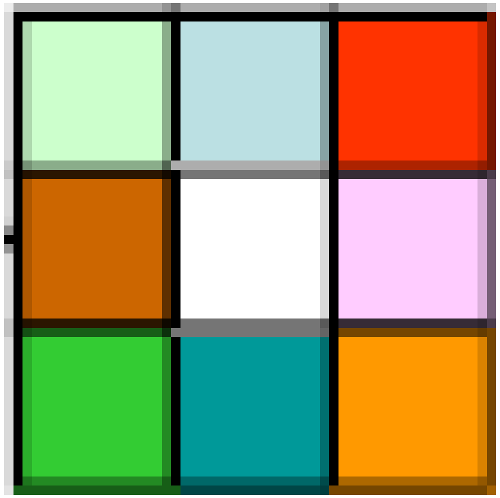
$= 4 + 6 + 3 + 1 + 0 + 2 + 0 + 0$

$= 16$

# 8-PUZZLE PROBLEM



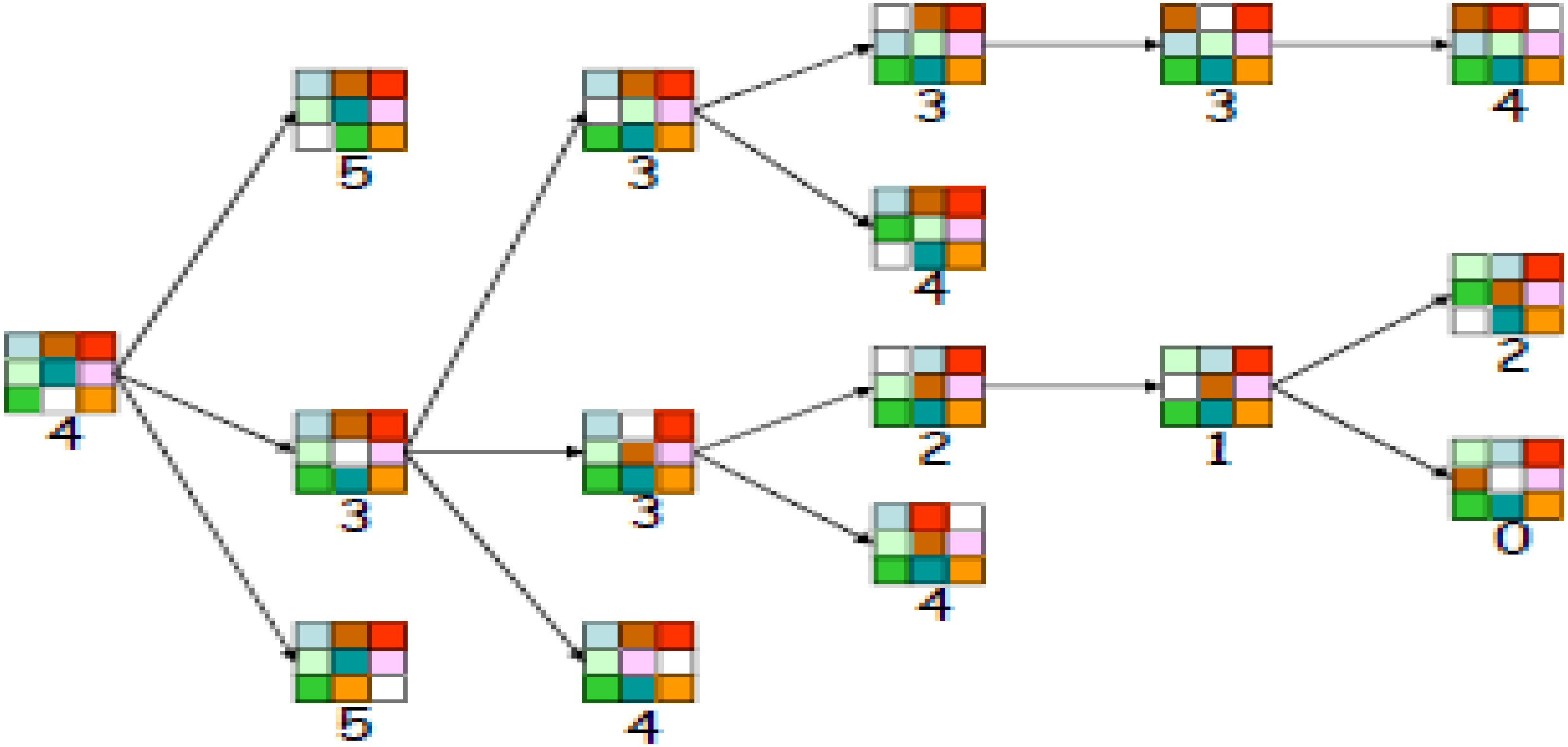
Initial State(N)



Goal State

# 8-Puzzle Problem using Hill Climbing

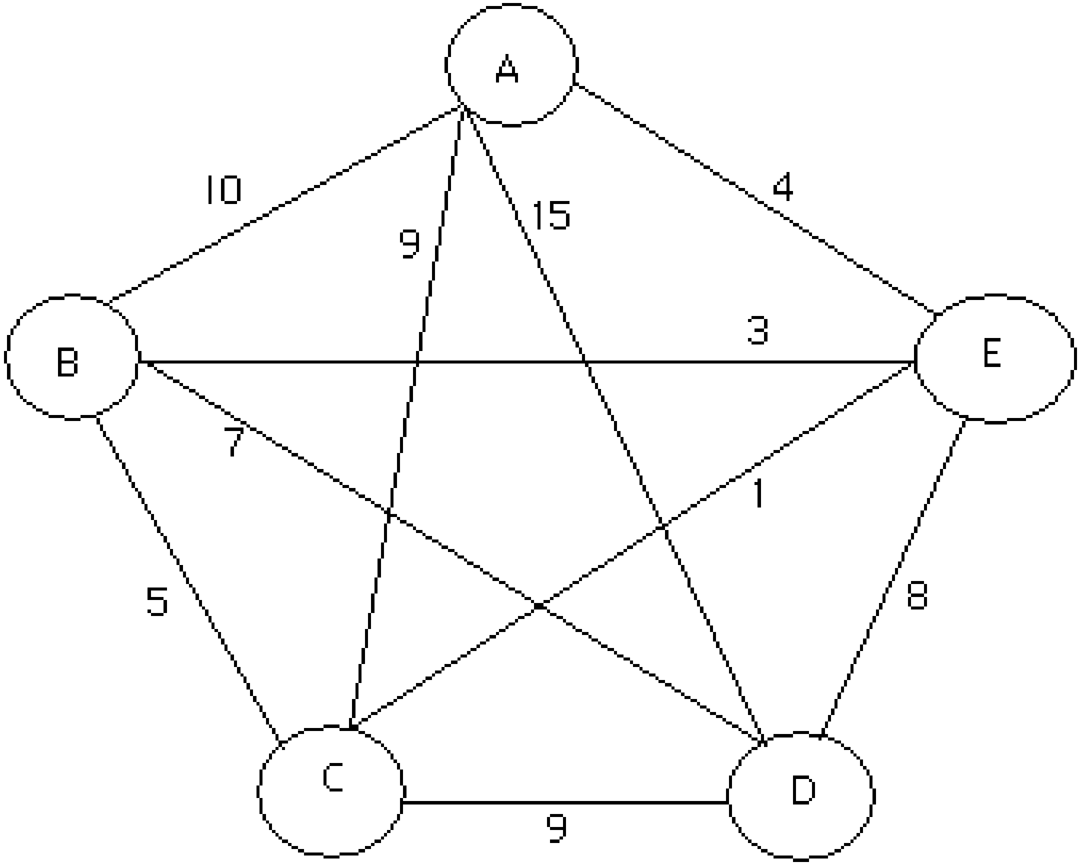
f(N) = h(N) = number of misplaced numbered tiles



The white tile is the empty tile

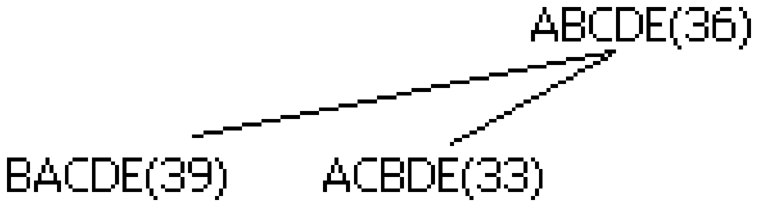
# HEURISTIC SEARCH TECHNIQUES

- Hill Climbing & TSP



# HEURISTIC SEARCH TECHNIQUES

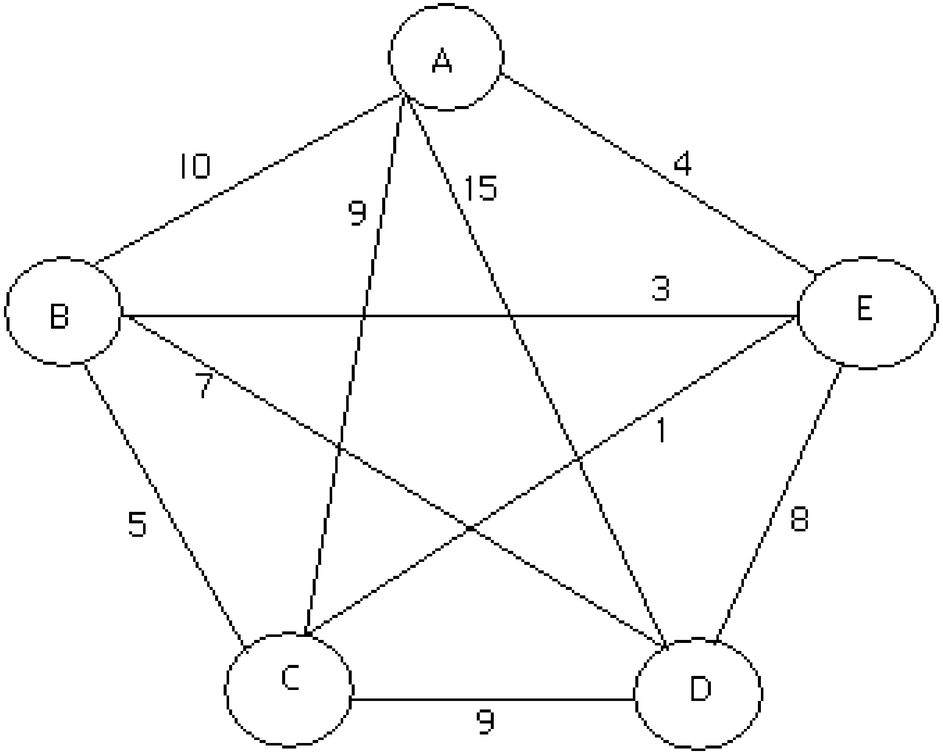
- Hill Climbing & TSP



ABCDE(36)

BACDE(39)

ACBDE(33)



A
10
9
15
4
B
3
E
7
1
5
8
C
9
D