The term Artificial Intelligence was given by John McCarthy. He defined AI as:

**AI Definition**: The science and Engineering of making intelligent machine especially intelligent computer programs

**INRODUCTION:**

1. AI is a set of patterns and algorithms that can generate solutions to everything without being explicitly instructed to do that work.
2. Artificial Intelligence is illustrated as machines. This technology has extended to be very famous in the modern world. It is the process of intelligence in machines that are taught to learn, understand, and copy the actions of humans. These machines are capable of learning new things with time and perform human-like tasks without being explicitly instructed.
3. Artificial Intelligence is:

* An intelligent entity created by humans.
* Capable of performing tasks intelligently without being explicitly instructed.
* Capable of thinking and acting rationally and humanely.

1. AI is constantly evolving, what was considered to be part of AI in the past may now just be looked at as a computer function. **For example**, a calculator may have been considered to be a part of AI in the past. Now, it is considered to be a simple function. Similarly, there are various levels of AI

**HISTORY of AI**

The birth of Artificial Intelligence (1952-1956)

* **Year 1955:** An Allen Newell and Herbert A. Simon created the "first artificial intelligence program"Which was named as **"Logic Theorist"**. This program had proved 38 of 52 Mathematics theorems, and find new and more elegant proofs for some theorems.
* **Year 1956:** The word "Artificial Intelligence" first adopted by American Computer scientist John McCarthy at the Dartmouth Conference. For the first time, AI coined as an academic field.

At that time high-level computer languages such as FORTRAN, LISP, or COBOL were invented. And the enthusiasm for AI was very high at that time.

The golden years-Early enthusiasm (1956-1974)

* **Year 1966:** The researchers emphasized developing algorithms which can solve mathematical problems. Joseph Weizenbaum created the first **chatbot**(a computer program designed to simulate conversation with human users)in 1966, which was named as ELIZA.
* **Year 1972:** The first intelligent humanoid robot was built in Japan which was named as WABOT-1.

**The first AI winter (1974-1980**)

* The duration between years 1974 to 1980 was the first AI winter duration. AI winter refers to the time period where computer scientist dealt with a severe shortage of funding from government for AI researches.
* During AI winters, an interest of publicity on artificial intelligence was decreased.

**A boom of AI (1980-1987)**

* **Year 1980:** After AI winter duration, AI came back with "Expert System". Expert systems were programmed that emulate the decision-making ability of a human expert.
* In the Year 1980, the first national conference of the American Association of Artificial Intelligence **was held at Stanford University**.

The second AI winter (1987-1993)

* The duration between the years 1987 to 1993 was the second AI Winter duration.
* Again Investors and government stopped in funding for AI research as due to high cost but not efficient result. The expert system such as XCON was very cost effective.

The emergence of intelligent agents (1993-2011)

* **Year 1997:** In the year 1997, IBM Deep Blue beats world chess champion, Garry Kasparov, and became the first computer to beat a world chess champion.
* **Year 2002:** for the first time, AI entered the home in the form of Roomba, a vacuum cleaner.
* **Year 2006:** AI came in the Business world till the year 2006. Companies like Facebook, Twitter, and Netflix also started using AI.

Deep learning, big data and artificial general intelligence (2011-present)

* **Year 2011:** In the year 2011, IBM's Watson won jeopardy, a quiz show, where it had to solve the complex questions as well as riddles. Watson had proved that it could understand natural language and can solve tricky questions quickly.
* **Year 2012:** Google has launched an Android app feature "Google now", which was able to provide information to the user as a prediction.
* **Year 2014:** In the year 2014, Chatbot "Eugene Goostman" won a competition in the infamous "Turing test."( The Turing Test is **a method of inquiry in artificial intelligence (AI) for determining whether or not a computer is capable of thinking like a human being**. )
* **Year 2018:** The "Project Debater" from IBM debated on complex topics with two master debaters and also performed extremely well.
* Google has demonstrated an AI program "Duplex" which was a virtual assistant and which had taken hairdresser appointment on call, and lady on other side didn't notice that she was talking with the machine.

Now AI has developed to a remarkable level. The concept of Deep learning, big data, and data science are now trending like a boom. Nowadays companies like Google, Facebook, IBM, and Amazon are working with AI and creating amazing devices. The future of Artificial Intelligence is inspiring and will come with high intelligence.

**INTELLIGENT SYSTEM:**

While studying artificial intelligence, you need to know what intelligence is:

What is Intelligence?

The ability of a system to calculate, reason, learn from experience, store and retrieve information from memory, solve problems, comprehend complex ideas, use natural language fluently, classify, generalize, and adapt new situations.

Intelligent Systems:

In order to design intelligent systems, it is important to categorize them into four categories

1. Systems that think like humans
2. Systems that think rationally(SENSIBLE/LOGICALLY)
3. Systems that behave like humans
4. Systems that behave rationally

**Cognitive Science(reasoning,remembering,thinking): Think Human-Like**

* 1. Requires a model for human cognition. Precise enough models allow simulation by computers.
  2. Focus is not just on behavior and I/O, but looks like reasoning process.
  3. Goal is not just to produce human-like behavior but to produce a sequence of steps of the reasoning process, similar to the steps followed by a human in solving the same task.

**Laws of thought: Think Rationally**

1. The study of **mental faculties** (ability to think clearly) through the use of computational models; that it is, the study of computations that make it possible to perceive reason and act.
2. Focus is on inference mechanisms that are probably correct and guarantee an optimal solution.
3. Goal is to formalize the reasoning process as a system of logical rules and procedures of inference.
4. Develop systems of representation to allow inferences to be like “*Socrates is a man. All men are mortal. Therefore Socrates is mortal”*

**Turing Test: Act Human-Like**

1. The art of creating machines that perform functions requiring intelligence when performed by people; that it is the study of, how to make computers do things which, at the moment, people do better.
2. Focus is on action, and not intelligent behavior centered around the representation of the world
3. Example: Turing Test
   * 3 rooms contain: a person, a computer and an interrogator.
   * The interrogator can communicate with the other 2 by teletype (to avoid the machine imitate the appearance of voice of the person)
   * The interrogator tries to determine which the person is and which the machine is.
   * The machine tries to fool the interrogator to believe that it is the human, and the person also tries to convince the interrogator that it is the human.
   * If the machine succeeds in fooling the interrogator, then conclude that the machine is intelligent.

## Foundations of Artificial Intelligence:

##### Philosophy

e.g., foundational issues (can a machine think?), issues of knowledge and believe, mutual knowledge

* + Psychology and Cognitive Science

e.g., problem solving skills

* + Neuro-Science

e.g., brain architecture

* + Computer Science And Engineering

e.g., complexity theory, algorithms, logic and inference, programming languages, and system building.

* + Mathematics and Physics

e.g., statistical modeling, continuous mathematics,

Sub Areas of AI:

1. **Game Playing**

Deep Blue Chess program beat world champion Gary Kasparov

1. Speech Recognition

PEGASUS spoken language interface to American Airlines' EAASY SABRE reseration system, which allows users to obtain flight information and make reservations over the telephone. The 1990s has seen significant advances in speech recognition so that limited systems are now successful.

1. Computer Vision

Face recognition programs in use by banks, government, etc. The ALVINN system from CMU autonomously drove a van from Washington, D.C. to San Diego (all but 52 of 2,849 miles), averaging 63 mph day and night, and in all weather conditions. Handwriting recognition, electronics and manufacturing inspection, photo interpretation, baggage inspection, reverse engineering to automatically construct a 3D geometric model.

1. Expert Systems

Application-specific systems that rely on obtaining the knowledge of human experts in an area and programming that knowledge into a system.

* 1. **Diagnostic Systems** : MYCIN system for diagnosing bacterial infections of the blood and suggesting treatments. Intellipath pathology diagnosis system (AMA approved). Pathfinder medical diagnosis system, which suggests tests and makes diagnoses. Whirlpool customer assistance center.
  2. System Configuration

DEC's XCON system for custom hardware configuration.

Radiotherapy treatment planning.

* 1. Financial Decision Making

Credit card companies, mortgage companies, banks, and the U.S. government employ AI systems to detect fraud and expedite financial transactions. For example, AMEX credit check.

* 1. Classification Systems

Put information into one of a fixed set of categories using several sources of information. E.g., financial decision making systems. NASA developed a system for classifying very faint areas in astronomical images into either stars or galaxies with very high accuracy by learning from human experts' classifications.

1. Mathematical Theorem Proving

Use inference methods to prove new theorems.

1. Natural Language Understanding

[AltaVista's translation](http://babelfish.altavista.digital.com/cgi-bin/translate) of web pages. Translation of Caterpillar Truck manuals into 20 languages.

1. Scheduling and Planning

Automatic scheduling for manufacturing. DARPA's DART system used in Desert Storm and Desert Shield operations to plan logistics of people and supplies. American Airlines rerouting contingency planner. European space agency planning and scheduling of spacecraft assembly, integration and verification.

1. Artificial Neural Networks:
2. **Machine Learning Application of AI:**

AI algorithms have attracted close attention of researchers and have also been applied successfully to solve problems in engineering. Nevertheless, for large and complex problems, AI algorithms consume considerable computation time due to stochastic feature of the search approaches

1. Business; financial strategies
2. Engineering: check design, offer suggestions to create new product, expert systems for all engineering problems
3. Manufacturing: assembly, inspection and maintenance
4. Medicine: monitoring, diagnosing
5. Education: in teaching
6. Fraud detection
7. Object identification
8. Information retrieval
9. Space shuttle scheduling

##### Defining the problem as State Space Search:

The state space representation forms the basis of most of the AI methods.

* Formulate a problem as a **state space search** by showing the legal problem states, the legal operators, and the initial and goal states.
* A **state** is defined by the specification of the values of all attributes of interest in the world
* An **operator** changes one state into the other; it has a precondition which is the value of certain attributes prior to the application of the operator, and a set of effects, which are the attributes altered by the operator
* The **initial state** is where you start
* The **goal state** is the partial description of the solution

##### Formal Description of the problem:

1. Define a state space that contains all the possible configurations of the relevant objects.
2. Specify one or more states within that space that describe possible situations from which the problem solving process may start **( initial state)**
3. Specify one or more states that would be acceptable as solutions to the problem**. ( goal states)**

Specify a set of rules that describe the actions **(operations)** available

S:{S, A, Action(S), Result(S, a), Cost(S, a)}

**S**=total no.of states (ex: start state, intermediate state, goal state)

Start state Goal state

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

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

**A**=set of all possible actions (ex:8 puzzle problem)

A:-up, down, left, right i.e all possible legal moves

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

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

Up move down move right move

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

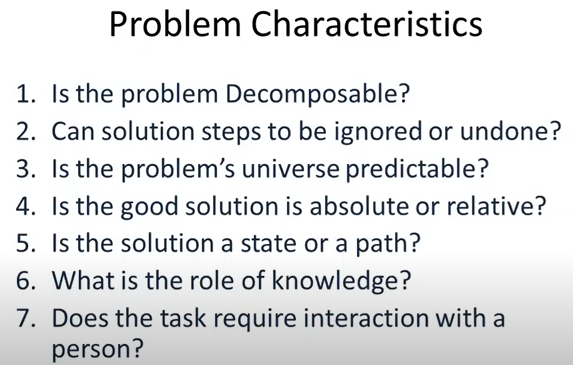
Result(s,a)

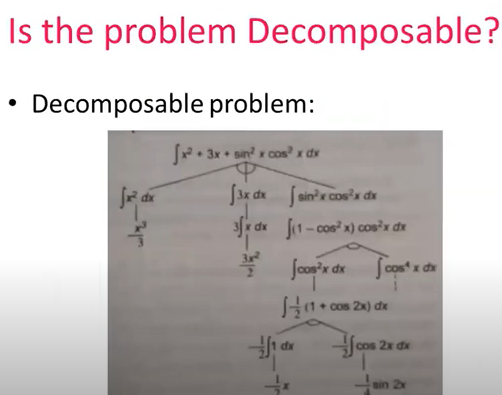
Properties of Search Algorithms

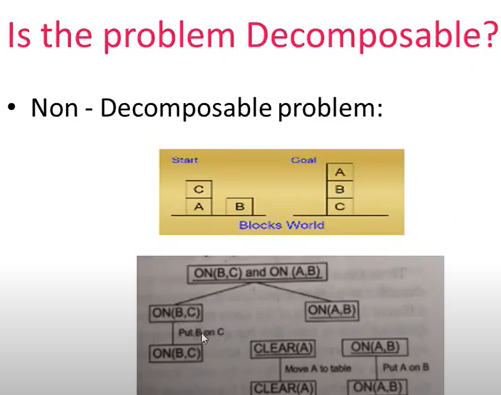
Which search algorithm one should use will generally depend on the problem domain. There are four important factors to consider:

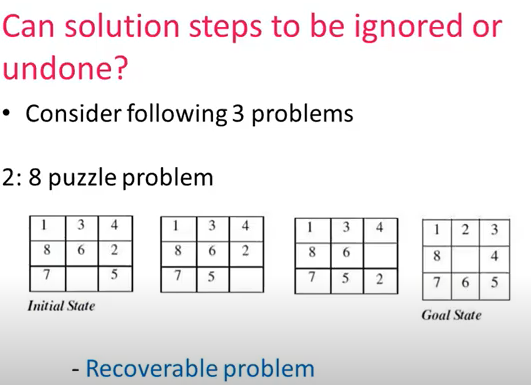
1. ***Completeness*** – Is a solution guaranteed to be found if at least one solution exists?
2. ***Optimality*** – Is the solution found guaranteed to be the best (or lowest cost) solution if there exists more than one solution?
3. ***Time Complexity*** – The upper bound on the time required to find a solution, as a function of the complexity of the problem.
4. ***Space Complexity*** – The upper bound on the storage space (memory) required at any point during the search, as a function of the complexity of the problem.

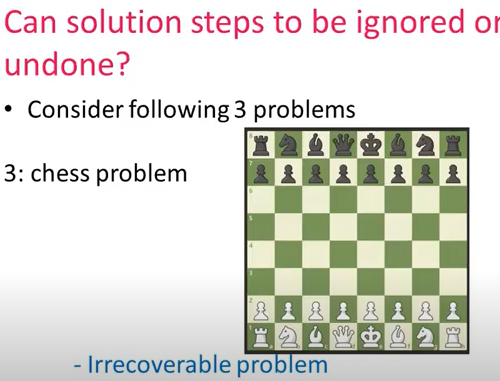
CHARACTERISTICS OF AI PROBLEM:

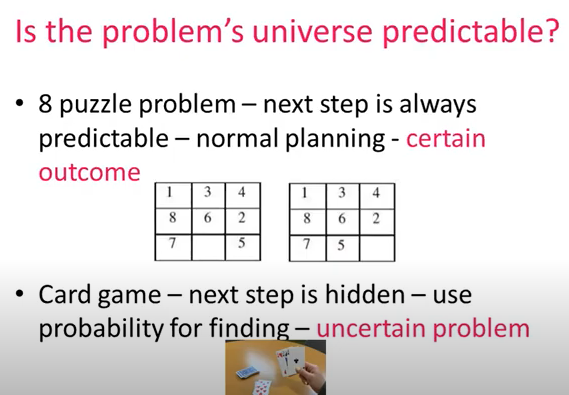


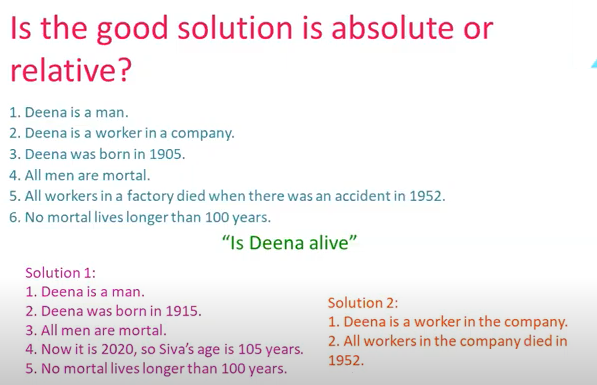


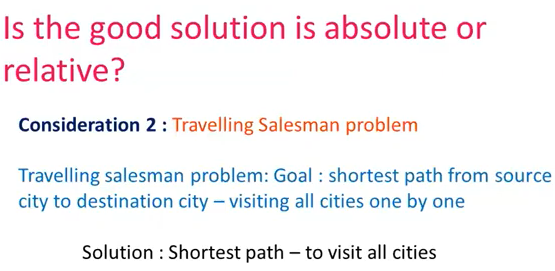


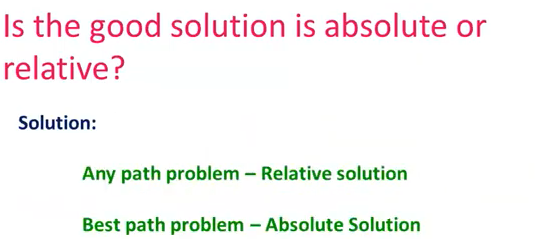


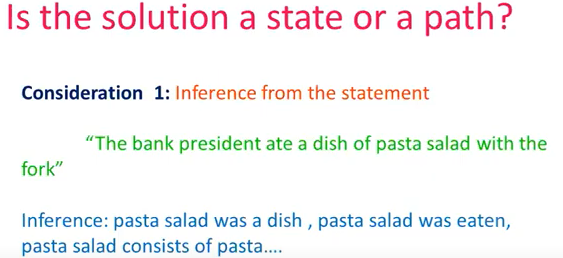


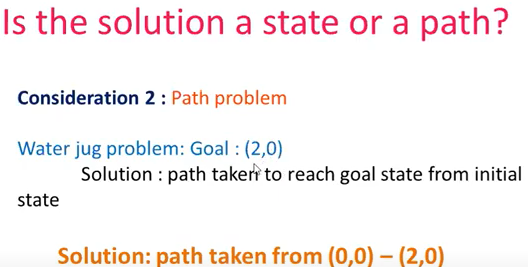


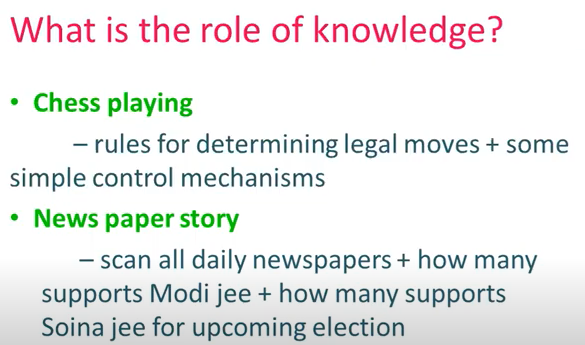


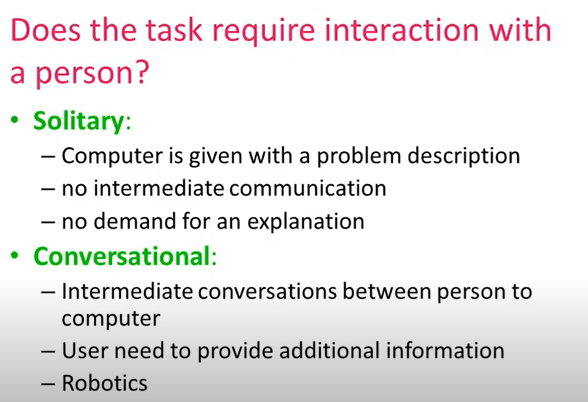










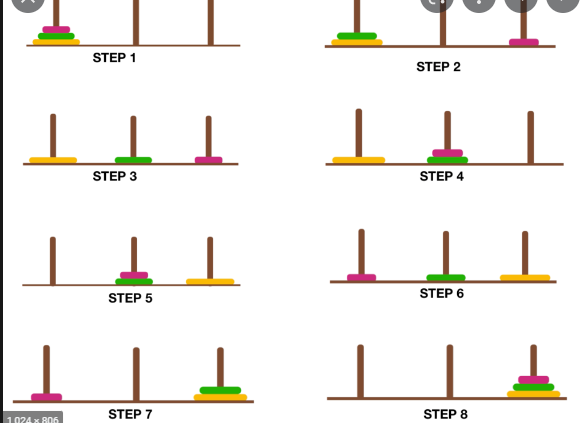


General Problem Solver*:*

The General Problem Solver (GPS) was the first useful AI program, written by Simon, Shaw, and Newell in 1959. As the name implies, it was intended to solve nearly any problem***.***

Newell and Simon defined each problem as a space. At one end of the space is the starting point; on the other side is the goal. The problem-solving procedure itself is conceived as a set of operations to cross that space, to get from the starting point to the goal state, one step at a time.

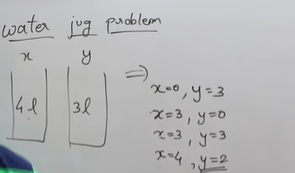
The General Problem Solver, the program tests various actions (which Newell and Simon called operators) to see which will take it closer to the goal state. An operator is any activity that changes the state of the system. The General Problem Solver always chooses the operation that appears to bring it closer to its goal.

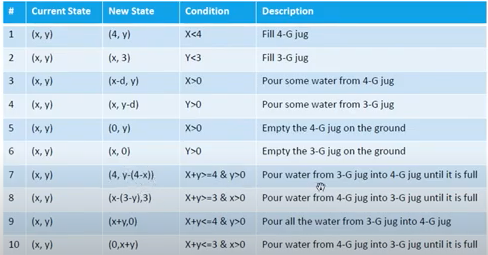


Example: Water Jug Problem

A Water Jug Problem: You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug?

Without applying any rule





Control strategies

Control Strategies means how to decide which rule to apply next during the process of searching for a solution to a problem.

Requirement for a good Control Strategy

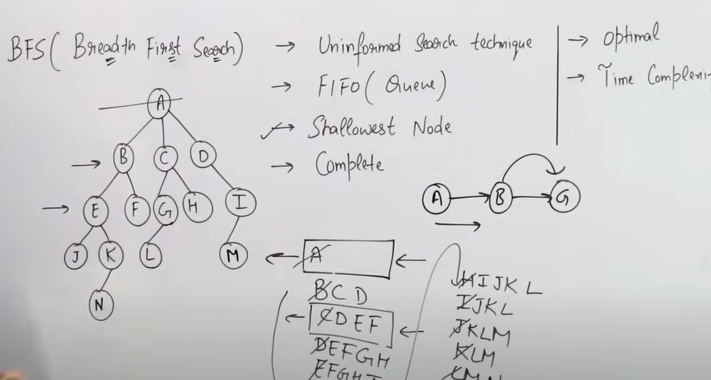
* 1. It should cause motion

In water jug problem, if we apply a simple control strategy of starting each time from the top of rule list and choose the first applicable one, then we will never move towards solution.

* 1. It should explore the solution space in a systematic manner

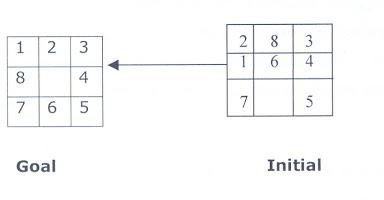
If we choose another control strategy, let us say, choose a rule randomly from the applicable rules then definitely it causes motion and eventually will lead to a solution. But one may arrive to same state several times. This is because control strategy is not systematic.

Systematic Control Strategies (Blind searches):

**Breadth First Search:** 

8 Puzzle Problem.

The 8 puzzle consists of eight numbered, movable tiles set in a 3x3 frame. One cell of the frame is always empty thus making it possible to move an adjacent numbered tile into the empty cell. Such a puzzle is illustrated in following diagram.

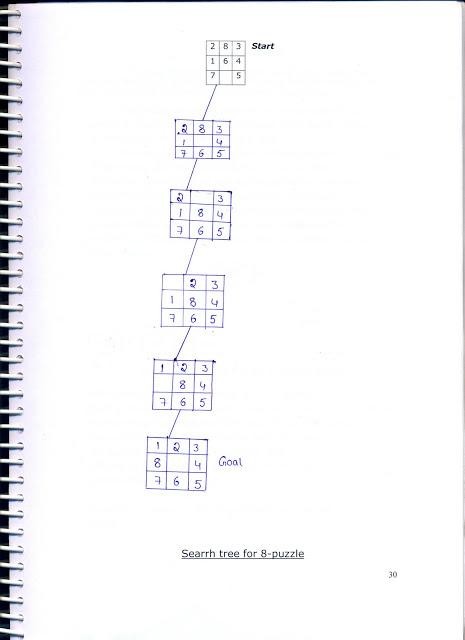


Once the problem states have been conceptually identified, we must construct a computer representation, or description of them. This description is then used as the database of a production system. For the 8-puzzle, a straight forward description is a 3X3 array of matrix of numbers. The initial global database is this description of the initial problem state. Virtually any kind of data structure can be used to describe states.

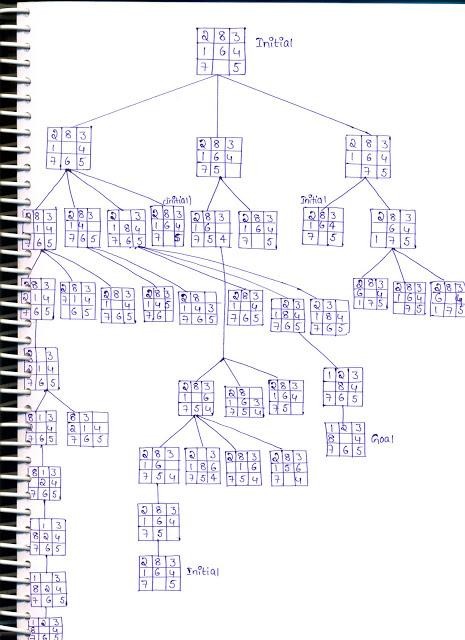
A move transforms one problem state into another state. The 8-puzzle is conveniently interpreted as having the following for moves. Move empty space (blank) to the left, move blank up, move blank to the right and move blank down,. These moves are modeled by production rules that operate on the state descriptions in the appropriate manner

The control strategy repeatedly applies rules to state descriptions until a description of a goal state is produced. It also keeps track of rules that have been applied so that it can compose them into sequence representing the problem solution. A solution to the 8-puzzle problem is given in the following figure.

**Example:- Depth – First – Search traversal (**8-puzzle)

****

**Breadth - First - Search traversal (**8-puzzle)



Exhaustive Searches, BFS and DFS

Search is the systematic examination of states to find path from the start/root state to the goal state.

Many traditional search algorithms are used in AI applications. For complex problems, the traditional algorithms are unable to find the solution within some practical time and space limits. Consequently, many special techniques are developed; using heuristic functions. The algorithms that use heuristic functions are called heuristic algorithms. Heuristic algorithms are not really intelligent; they appear to be intelligent because they achieve better performance.

Heuristic algorithms are more efficient because they take advantage of feedback from the data to direct 2the search path.

Uninformed search

Also called blind, exhaustive or brute-force search, uses no information about the problem to guide the search and therefore may not be very efficient.

Informed Search:

Also called heuristic or intelligent search, uses information about the problem to guide the search, usually guesses the distance to a goal state and therefore efficient, but the search may not be always possible.

Uninformed Search Methods:

**Breadth- First -Search:**

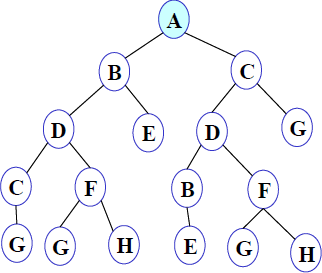
Consider the state space of a problem that takes the form of a tree. Now, if we search the goal along each breadth of the tree, starting from the root and continuing up to the largest depth, we call it *breadth first search*.

* Algorithm:

1. Create a variable called NODE-LIST and set it to initial state
2. Until a goal state is found or NODE-LIST is empty do
   1. Remove the first element from NODE-LIST and call it E. If NODE-LIST was empty, quit
   2. For each way that each rule can match the state described in E do:
      1. Apply the rule to generate a new state
      2. If the new state is a goal state, quit and return this state
      3. Otherwise, add the new state to the end of NODE-LIST

BFS illustrated:

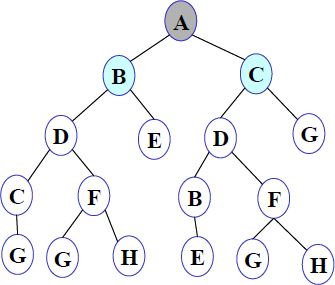
**Step 1:** Initially fringe(edge,border) contains only one node corresponding to the source state A.



##### Figure 1

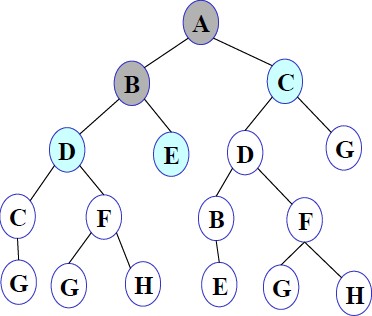
**FRINGE: A**

**Step 2:** A is removed from fringe. The node is expanded, and its children B and C are generated. They are placed at the back of fringe.



##### Figure 2

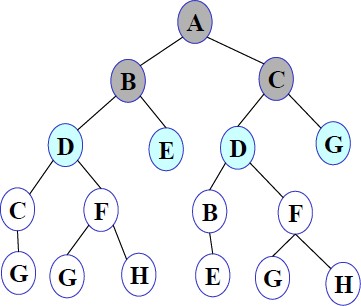
**FRINGE: B C**

**Step 3:** Node B is removed from fringe and is expanded. Its children D, E are generated and put at the back of fringe.

##### Figure 3

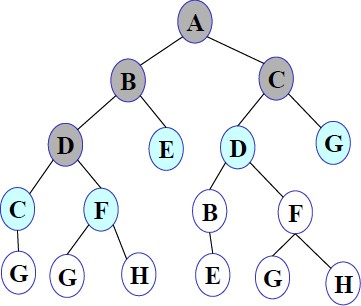
**FRINGE: C D E**

**Step 4:** Node C is removed from fringe and is expanded. Its children D and G are added to the back of fringe.



##### Figure 4

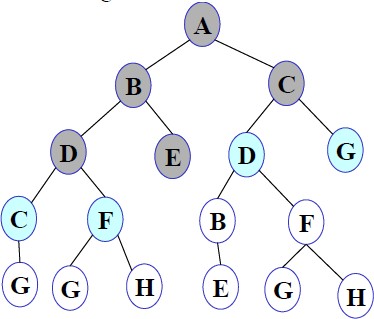
**FRINGE: D E D G**

**Step 5**: Node D is removed from fringe. Its children C and F are generated and added to the back of fringe.

##### Figure 5

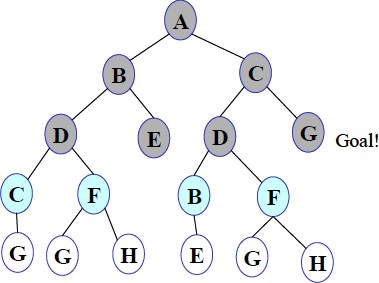
**FRINGE: E D G C F**

**Step 6**: Node E is removed from fringe. It has no children.



##### Figure 6

**FRINGE: D G C F**

**Step 7**: D is expanded; B and F are put in OPEN.

##### Figure 7

**FRINGE: G C F B F**

**Step 8**: G is selected for expansion. It is found to be a goal node. So the algorithm returns the path A C G by following the parent pointers of the node corresponding to G. The algorithm terminates.

Breadth first search is:

* + One of the simplest search strategies
  + Complete. If there is a solution, BFS is guaranteed to find it.
  + If there are multiple solutions, then a minimal solution will be found
  + The algorithm is optimal (i.e., admissible) if all operators have the same cost. Otherwise, breadth first search finds a solution with the shortest path length.
  + **Time complexity** : O(bd )
  + **Space complexity** : O(bd )
  + **Optimality** :Yes

b - branching factor(maximum no of successors of any node), d – Depth of the shallowest goal node

***Maximum length of any path (m) in search space***

**Advantages**: Finds the path of minimal length to the goal.

Disadvantages:

* Requires the generation and storage of a tree whose size is exponential the depth of the shallowest goal node.
* The breadth first search algorithm cannot be effectively used unless the search space is quite small.

Depth- First- Search.

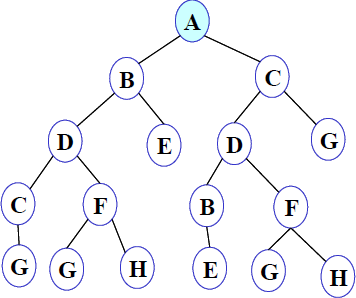
We may sometimes search the goal along the largest depth of the tree, and move up only when further traversal along the depth is not possible. We then attempt to find alternative offspring of the parent of the node (state) last visited. If we visit the nodes of a tree using the above principles to search the goal, the traversal made is called depth first traversal and consequently the search strategy is called *depth first search*.

* Algorithm:

1. Create a variable called NODE-LIST and set it to initial state
2. Until a goal state is found or NODE-LIST is empty do
   1. Remove the first element from NODE-LIST and call it E. If NODE-LIST was empty, quit
   2. For each way that each rule can match the state described in E do:
      1. Apply the rule to generate a new state
      2. If the new state is a goal state, quit and return this state
      3. Otherwise, add the new state in front of NODE-LIST

##### DFS illustrated:

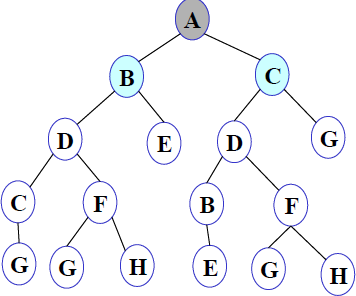
**A State Space Graph Step 1**: Initially fringe contains only the node for A.



##### Figure 1

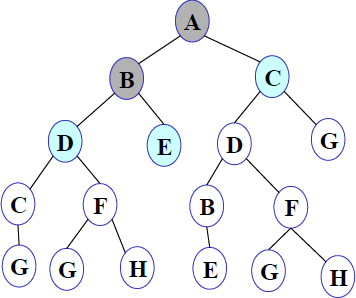
**FRINGE: A**

**Step 2:** A is removed from fringe. A is expanded and its children B and C are put in front of fringe.



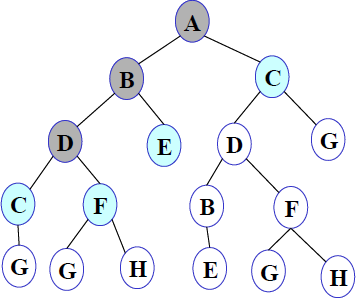
##### Figure 2

**FRINGE: B C**

**Step 3:** Node B is removed from fringe, and its children D and E are pushed in front of fringe.

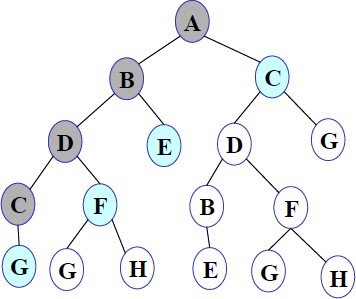
##### Figure 3

**FRINGE: D E C**

**Step 4:** Node D is removed from fringe. C and F are pushed in front of fringe.

##### Figure 4

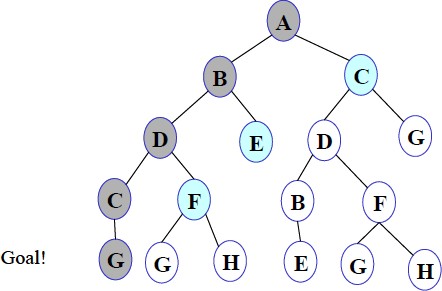
**FRINGE: C F E C**

**Step 5:** Node C is removed from fringe. Its child G is pushed in front of fringe.

##### Figure 5

**FRINGE: G F E C**

**Step 6:** Node G is expanded and found to be a goal node.



##### Figure 6

**FRINGE: *G* F E C**

The solution path A-B-D-C-G is returned and the algorithm terminates.

Depth first searchis:

1. The algorithm takes exponential time.
2. If N is the maximum depth of a node in the search space, in the worst case the algorithm will

d

take time O(b ).

1. The space taken is linear in the depth of the search tree, O(bN).

Note that the time taken by the algorithm is related to the maximum depth of the search tree. If the search tree has infinite depth, the algorithm may not terminate. This can happen if the search space is infinite. It can also happen if the search space contains cycles. The latter case can be handled by checking for cycles in the algorithm. Thus **Depth First Search is not complete.**

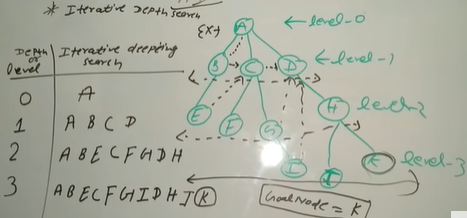
**Exhaustive searches- Iterative Deeping DFS**

**Description:**

* It is a search strategy resulting when you combine BFS and DFS, thus combining the advantages of each strategy, taking the completeness and optimality of BFS and the modest memory requirements of DFS.
* IDS works by looking for the best search depth d, thus starting with depth limit 0 and make a BFS and if the search failed it increase the depth limit by 1 and try a BFS again with depth 1 and so on – first d = 0, then 1 then 2 and so on – until a depth d is reached where a goal is found.
* As it takes the advantages of BFS +DFS

advantages of BFS is: it will find the goal node at any way + advantages of DFS is it

consumes less memory



Disadvantage: repeatedly searching of same node

Heuristic Searches:

1. A Heuristic technique helps in solving problems, even though there is no guarantee that it will never lead in the wrong direction. It is a technique design to solve a problem quickly(a kind of guess)

2. It reduces the time as it will not explore all the nodes but only it explores the nodes which has less heuristic values.

3. If we want to solve the problem in polynomial time then we choose heuristic search technique

A heuristic function is a function that maps 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 solution.

How can we calculate heuristic value?

* 1. Euclidean distance(straight line distance)—sqrt((x2-x1)\*\*2) +(y2-y1)\*\*2))
  2. Manhatlen diatance(no.of misplaced tiles if we take an example of 8 puzzle)

Greedy Best First Search

Greedy best-first search tries to expand the node that is closest to the goal, on the: grounds that this is likely to lead to a solution quickly. Thus, it **evaluates nodes by using just the heuristic function:**

**f *(n)* = *h (n)****.*

Using the **straight-line distance** heuristic **hSLD,** the goal state can be reached faster.

### Best first search algorithm:

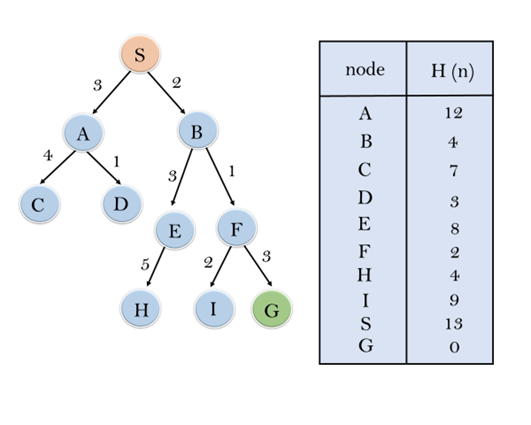
* **Step 1:** Place the starting node into the OPEN list.
* **Step 2:** If the OPEN list is empty, Stop and return failure.
* **Step 3:** Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.
* **Step 4:** Expand the node n, and generate the successors of node n.
* **Step 5:** Check each successor of node n, and find whether any node is a goal node or not. If any successor node is goal node, then return success and terminate the search, else proceed to Step 6.
* **Step 6:** For each successor node, algorithm checks for evaluation function f(n), and then check if the node has been in either OPEN or CLOSED list. If the node has not been in both list, then add it to the OPEN list.
* **Step 7:** Return to Step 2.

### Advantages:

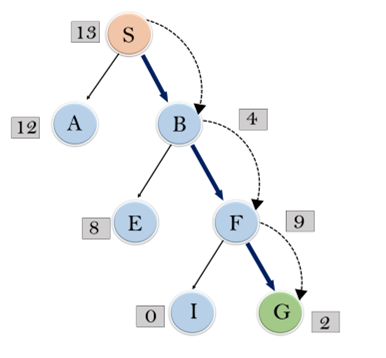
* Best first search can switch between BFS and DFS by gaining the advantages of both the algorithms.
* This algorithm is more efficient than BFS and DFS algorithms.

### Disadvantages:

* It can behave as an unguided depth-first search in the worst case scenario.
* It can get stuck in a loop as DFS.
* This algorithm is not optimal.



In this search example, we are using two lists which are **OPEN** and **CLOSED** Lists. Following are the iteration for traversing the above example.



**Expand the nodes of S and put in the CLOSED list**

**Initialization:** Open [A, B], Closed [S]

**Iteration 1:** Open [A], Closed [S, B]

**Iteration 2:** Open [E, F, A], Closed [S, B]  
                  : Open [E, A], Closed [S, B, F]

**Iteration 3:** Open [I, G, E, A], Closed [S, B, F]  
                  : Open [I, E, A], Closed [S, B, F, G]

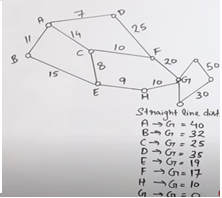
Hence the final solution path will be: **S----> B----->F----> G**

**Time Complexity:** The worst case time complexity of Greedy best first search is O(bm).

**Space Complexity:** The worst case space complexity of Greedy best first search is O(bm). Where, m is the maximum depth of the search space.

**Complete:** Greedy best-first search is also incomplete, even if the given state space is finite.

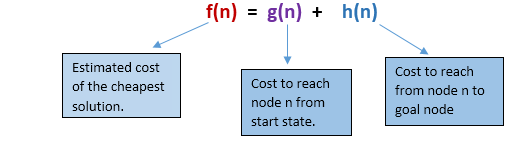
**Optimal:** Greedy best first search algorithm is not optimal.



##### A\* Algorithm

The Best First algorithm is a simplified form of the *A*\* algorithm.

The **A\* search algorithm** (pronounced "Ay-star") is a [tree search algorithm](http://www.fact-index.com/t/tr/tree_search_algorithm.html) that finds a path from a given initial node to a given goal node .A\* search algorithm finds the shortest path through the search space using the heuristic function. This search algorithm expands less search tree and provides optimal result faster.



At each point in the search space, only those node is expanded which have the lowest value of f(n), and the algorithm terminates when the goal node is found.

### Algorithm of A\* search:

**Step1:** Place the starting node in the OPEN list.

**Step 2:** Check if the OPEN list is empty or not, if the list is empty then return failure and stops.

**Step 3:** Select the node from the OPEN list which has the smallest value of evaluation function (g+h), if node n is goal node then return success and stop, otherwise

**Step 4:** Expand node n and generate all of its successors, and put n into the closed list. For each successor n', check whether n' is already in the OPEN or CLOSED list, if not then compute evaluation function for n' and place into Open list.

**Step 5:** Else if node n' is already in OPEN and CLOSED, then it should be attached to the back pointer which reflects the lowest g(n') value.

**Step 6:** Return to **Step 2**.

### Advantages:

* A\* search algorithm is the best algorithm than other search algorithms.
* A\* search algorithm is optimal and complete.
* This algorithm can solve very complex problems.

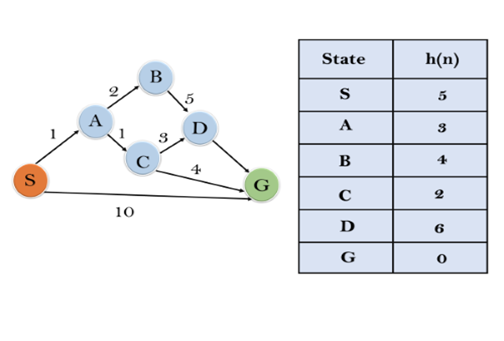
### Disadvantages:

* It does not always produce the shortest path as it mostly based on heuristics and approximation.
* A\* search algorithm has some complexity issues.
* The main drawback of A\* is memory requirement as it keeps all generated nodes in the memory, so it is not practical for
* various large-scale problems.

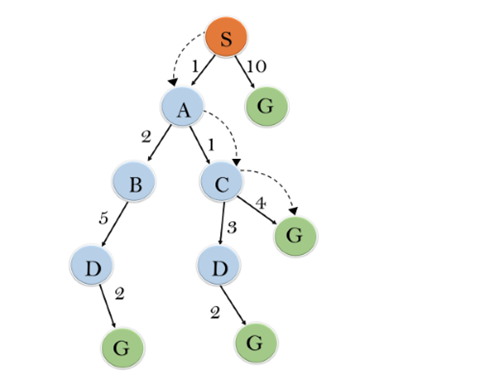
### Example:

In this example, we will traverse the given graph using the A\* algorithm. The heuristic value of all states is given in the below table so we will calculate the f(n) of each state using the formula f(n)= g(n) + h(n), where g(n) is the cost to reach any node from start state.

Here we will use OPEN and CLOSED list.



Solution:



**Initialization:** {(S, 5)}

**Iteration1:** {(S--> A, 4), (S-->G, 10)}

**Iteration2:** {(S--> A-->C, 4), (S--> A-->B, 7), (S-->G, 10)}

**Iteration3:** {(S--> A-->C--->G, 6), (S--> A-->C--->D, 11), (S--> A-->B, 7), (S-->G, 10)}

**Iteration 4** will give the final result, as **S--->A--->C--->G** it provides the optimal path with cost 6.

**Points to remember:**

* A\* algorithm returns the path which occurred first, and it does not search for all remaining paths.
* The efficiency of A\* algorithm depends on the quality of heuristic.
* A\* algorithm expands all nodes which satisfy the condition f(n)<="" li="">

**Complete:** A\* algorithm is complete as long as:

* Branching factor is finite.
* Cost at every action is fixed.

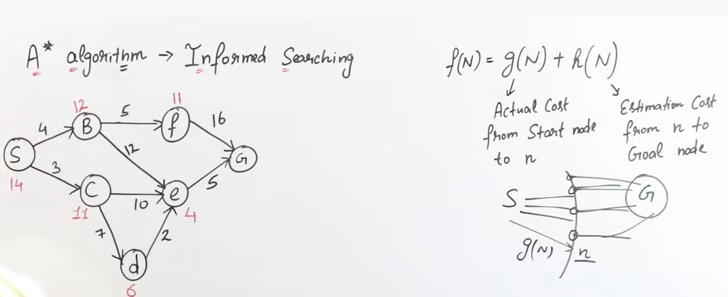
**Optimal:** A\* search algorithm is optimal if it follows below two conditions:

* **Admissible:** the first condition requires for optimality is that h(n) should be an admissible heuristic for A\* tree search. An admissible heuristic is optimistic in nature.
* **Consistency:** Second required condition is consistency for only A\* graph-search.

If the heuristic function is admissible, then A\* tree search will always find the least cost path.

**Time Complexity:** The time complexity of A\* search algorithm depends on heuristic function, and the number of nodes expanded is exponential to the depth of solution d. So the time complexity is O(b^d), where b is the branching factor.

**Space Complexity:** The space complexity of A\* search algorithm is **O(b^d)**



F(S)=0+14=14

S->B S->C

4+12=16 3+11=14 (14 IS MIN COST )

SB->F SB->E SC->E SC->D

4+5+11=20 4+12+4=20 3+10+4=17 3+7+6=16 (16 IS MIN COST)

SC->D IS MIN COST ie 16

S->B HAS GAIN 16 AS MIN COST,SO WE WILL EXPLORE S->B and SCD->E

FROM S->B WE EXPLORE SB->F & SB->E WITH COST 20 AND WHERE AS SCD->E WITH COST 16,SO WE EXPLORE SCDE->G

F(S)=0+14=14

S->B S->C

4+12=16 3+11=14 (14 IS MIN COST )

SC->E SC->D

3+10+4=17 3+7+6=16 (16 IS MIN COST )

SCD->E

3+7+2+4=16

SCDE->G

3+7+2+5=17

ITERATIVE DEEPENING A\* (IDA\*)

1.It performs depth first search with limited to some F-bound

Use f(n)=g(n)+h(n)

Algorithm:

Step 1:First set the threshold a, the least value f(n)

Where f(n)=g(n)+h(n)

Step 2:now by checking if the f-score<=threshold ,expand

Else check for the least f(n)

Step 3:continue it until you reach goal node

Step 4:all rejected or stopped value added into f value list

Advantages:

1.finds an optimal solution

2.less storage space

Disadvantage:

cost and computational time is more

A

1 2

B C

3 9 5 4 3 6 8

D E F

G H I J

1

A - 8

B -10

C - 4

D -15

E - 14

F - 12

G - 7

H - 2

I - 0

J – 4

BY CALCULATING F(N)=G(N)+H(N)

8 f-scores

A

11 6

B C

23 24 18

D E F 13 7 8 14

G H I J

I

6

1 Consider minimum f-score value as threshold

2 Maintain a list ,f-values

3 If f-score > threshold:

Add f-score to the list f-values

Never explore the start node. Go to step 4

4 consider the next minimum f-score value as threshold.Repeat untill

f-score <= threshold.

5.if f-score <= threshold: Explore the node ,else go to step 3

6.Repeat the steps 5 until goal node is reached

CONSTRAINT SATISFACTION PROBLEM(CSP):

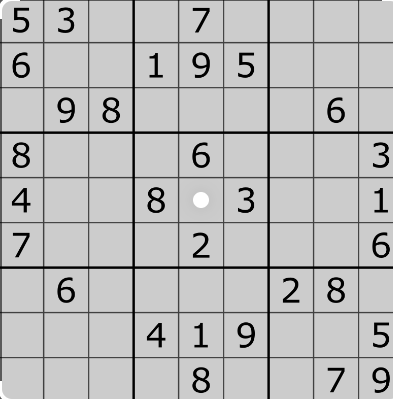
* CSP consists of 3 components V , D, C
* V is set of variables {v1,v2,v3……vn}
* D is set of domains{D1,D2,……Dn}
* C is set of constraints that specify allowable combination of values

Ci=(scope, rel) where

**scope** is set of variables that participate in constraint

**rel** is Relation that define the values that variable can take

**Example**: sudoku which is 9x9 total 81 squares



**Total no.of spaces are variables** (where some spaces have some values- means(some square boxes have values and some may have no values)

**Domain** means in each square box which number can be place(ie 1 to 9)

Means each box can be placed range from 1 to 9

**Constraint** means that row or column should not contain the repeated number

WORKING OF CSP ALGORITHM WITH AN EXAMPLE

1 2

|  |
| --- |
|  |

3 4

V={1,2,3,4}

D={Red,Green,Blue}

C={1!=2,1!=3,1!=4,2!=4,3!=4}

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Intial Dom | R,G,B | R,G,B | R,G,B | R,G,B |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Intial D  1=R | R,G,B  R | R,G,B  G,B | R,G,B  G,B | R,G,B  G,B |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Intial D  1=R  2=G | R,G,B  R  R | R,G,B  G,B  G | R,G,B  G,B  G,B | R,G,B  G,B  B |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Intial D  1=R  2=G  3=B | R,G,B  R  R  R | R,G,B  G,B  G  G | R,G,B  G,B  G,B  B | R,G,B  G,B  B  B |

Now Backtracking will be applied because here the constraint is not satisfying as 3=B & 4=B

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Intial D  1=R  2=G  3=G | R,G,B  R  R  R | R,G,B  G,B  G  G | R,G,B  G,B  G,B  G | R,G,B  G,B  B  B |