Artificial intelligence for robotics engineering

Unit objectives

After completing this unit, you should be able to:

- Gain knowledge on the basics of Artificial Intelligence (AI)
- Gain an insight into the AI problems and techniques
- Learn about the state space search and production systems
- Understand the concept of problem characteristics and search paradigm
- Learn about heuristic search techniques and knowledge representation

Artificial intelligence

- Artificial intelligence (AI) is the study and design of intelligent agents.
- AI is defined as the science and engineering of making machines intelligent.
- Science of making machines do things that would require intelligence.
- It is the intelligence of machines and the branch of computer science that aims to create it.
- It focuses on developing hardware and software systems that solve problems and accomplish tasks.

Well-known definitions for AI

- Intelligence is the computational part of the ability to achieve goals in the world.
- Varying kinds and degrees of intelligence occur in people, many animals.
- AI is the study of the mental faculties through the use of computational models.
- An intelligent agent is a system that studies and acts on the basis of environment.

Birth of AI

- Another influential figure in AI is john McCarthy.
- McCarthy and Minsky, Claude Shannon, and Nathaniel Rochester brought together us researchers interested in the study of intelligence.
- Two researchers from Carnegie tech, Allen newel and Herbert Simon created a reasoning program, the logic theorist (LT)

Basic terminologies

- An agent is something that perceives and acts in an environment.
- The performance measure evaluates the behavior of the agent in an environment.
- The agent program implements the agent function.
- Goal directed agent is a type of intelligent agent.

Applications of AI

- Applications of artificial intelligence:
  - Autonomous planning and scheduling.
  - Game playing.
  - Autonomous control.
  - Diagnosis.
  - Logistics planning.
  - Robotics.
  - Language understanding and problem solving.

The AI problems and techniques

- What is an AI technique?
- Properties of knowledge which are less desirable.
- A problem can be defined formally by following components:
  - The initial state.
  - A description of the possible actions.
  - The goal test.
  - A path cost.
  - A solution.

Real world problems

- Any real-world problem specified by the following:
  - States.
  - Initial state.
  - Successor function.
  - Goal test.
  - Path cost.

- Example problems:
  - TSP: traveling salesperson problem.
  - Routing in computer networks.
  - VLSI layout problem.
  - Robot navigation.

State space search

- State space search is a process used in the field of computer science, including artificial intelligence (AI)

- A state space search includes:
  - A state.
  - An operator.
  - The initial state.
  - The goal state.

- Explicit state space: One possible representation of the effect and precondition of actions is to explicitly enumerate the states.

- Search is the process of considering various possible sequences of operators applied to the initial state and finding out a sequence which culminates in a goal state.

Explicit vs. Implicit state space

- Explicit state space:
  - This would require a table such as the following:

State	Action	Resulting State
s_7	act_{47}	s_{94}
s_7	act_{14}	s_{83}
s_{94}	act_5	s_{33}
...	...	...

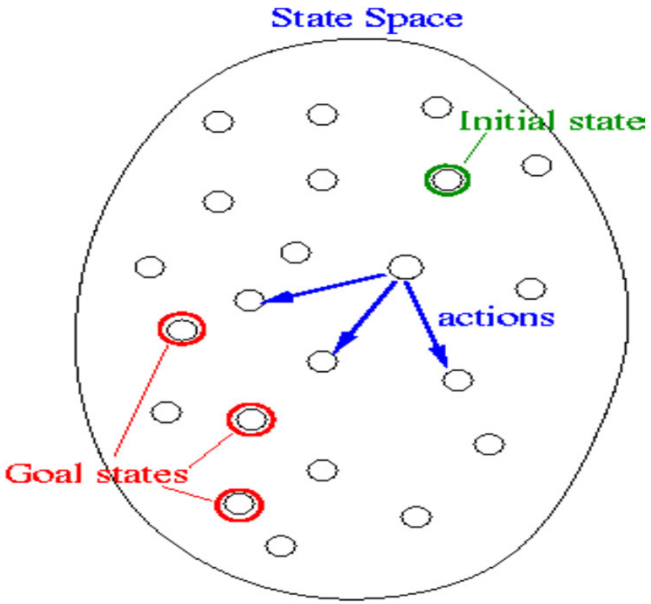
- To generate the state space implicitly, the agent needs to know both initial state and operators.

State space search notations

- An initial state is agent's configuration at the beginning.
- An *action* or an operator takes to the successor state.
- A state can have a number of successor states.
- A *plan* is a sequence of actions.
- The path cost represents cost of a plan.

Search problem and problem space

- Search plays a major role in solving many artificial intelligence (AI) problems.
- The problems that are addressed by AI search algorithms fall into three general classes:
  - Single-agent path-finding problems.
  - Two-player games, and
  - Constraint-satisfaction problems.
- Problem space represents connections between states.
- Schematic representation of search problem:



Representation of search problems

- A search problem is represented using a directed graph.

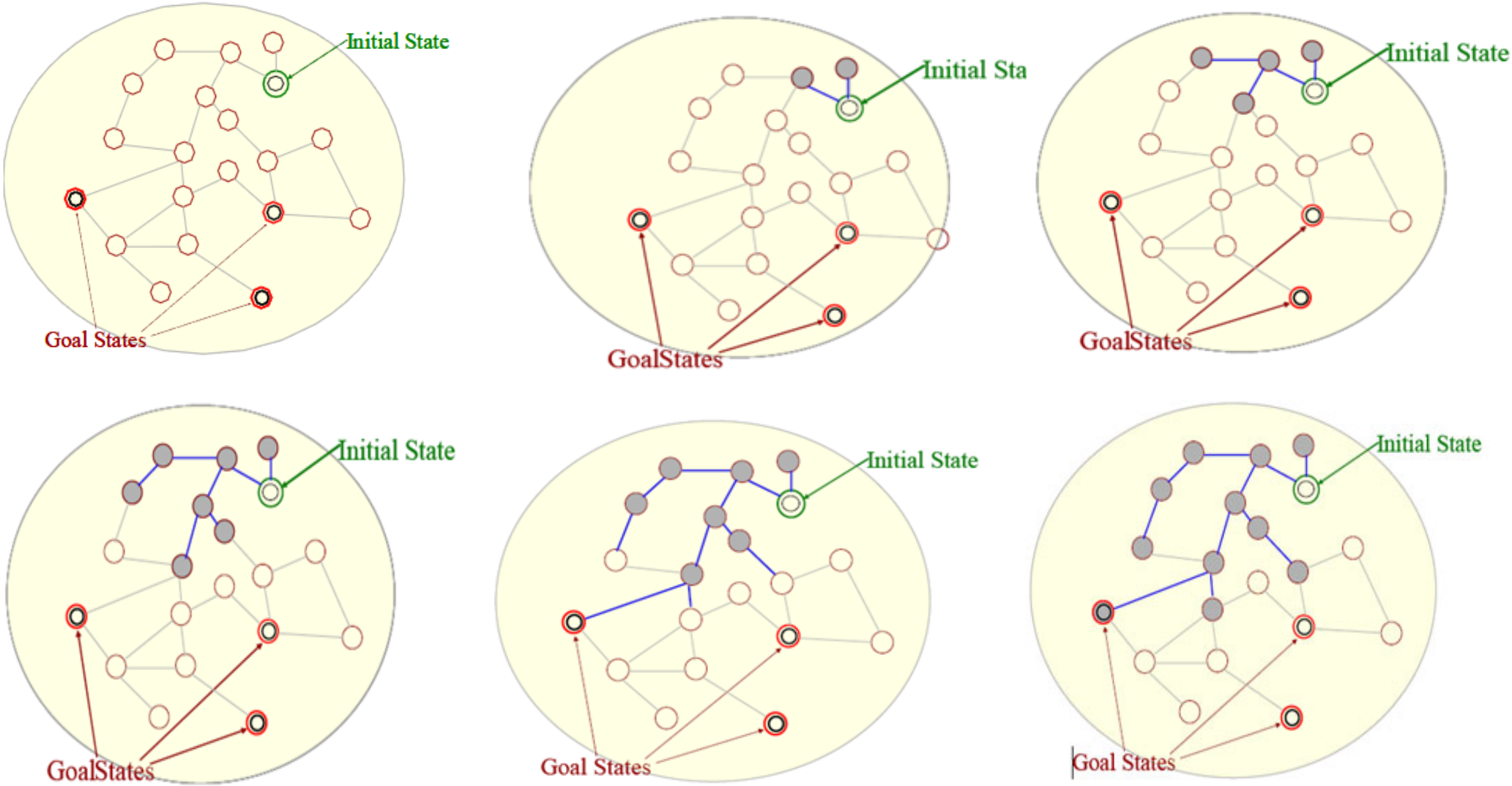
- Searching process: the generic searching process can be very simply described in terms of the following steps:

Do until a solution is found or the state space is exhausted.

1. Check the current state.
2. Execute allowable actions to find the successor states.
3. Pick one of the new states.
4. Check if the new state is a solution state.
5. If it is not, the new state becomes the current state and the process is repeated.

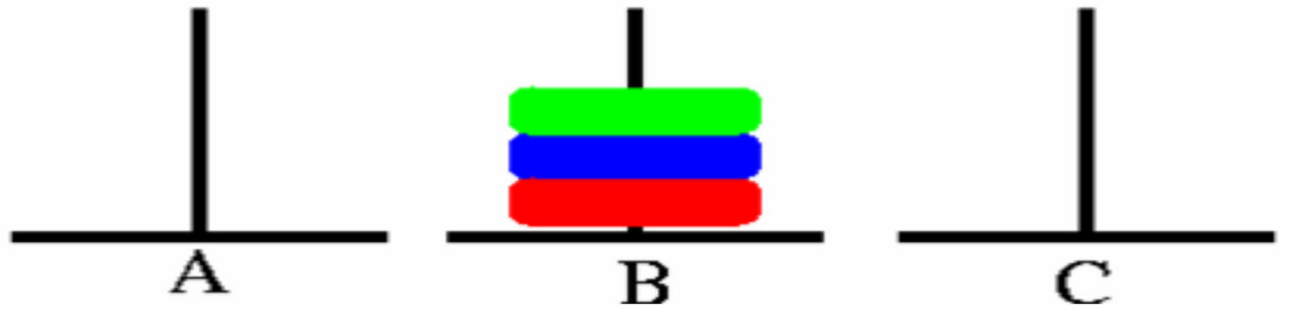
State space search example

- We will now illustrate the searching process with the help of an example. Consider the problem depicted in Figure.

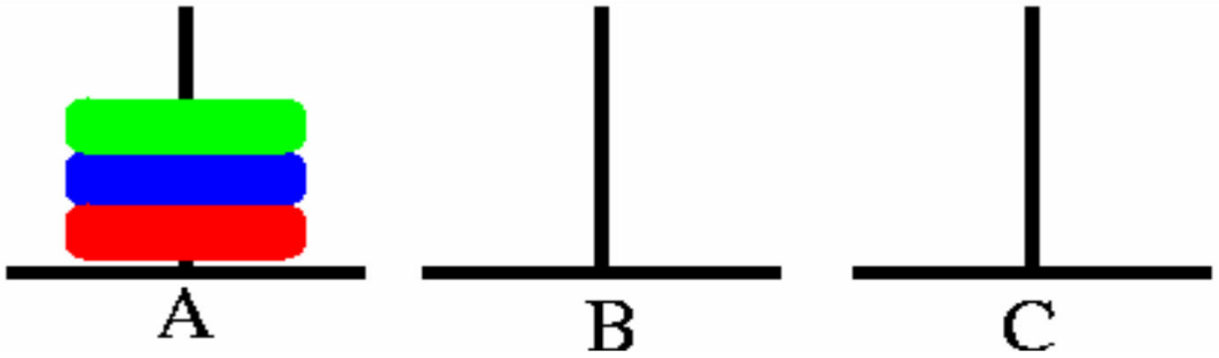


Pegs and disks problem (1 of 4)

- Figure: Pegs and Disks problem.

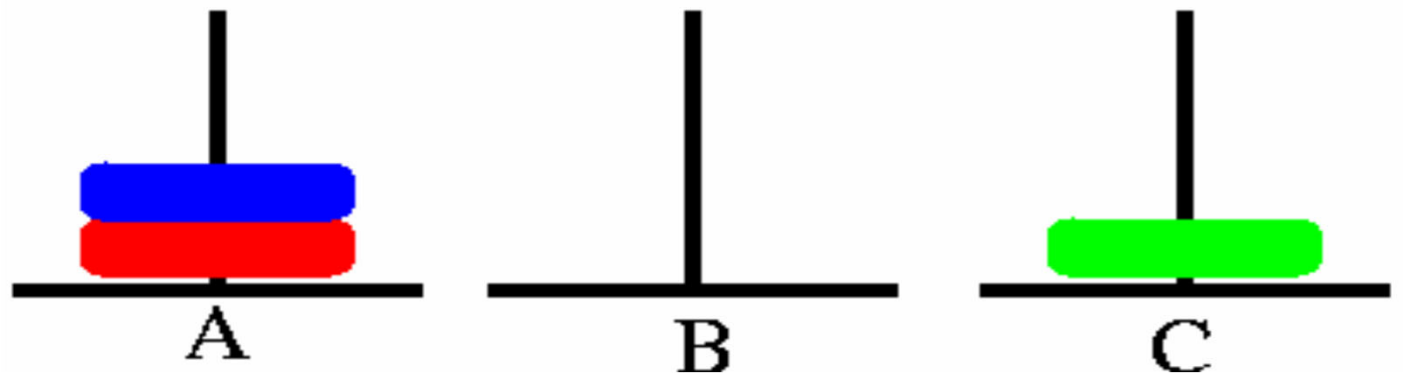


- Figure: The initial state is illustrated below.

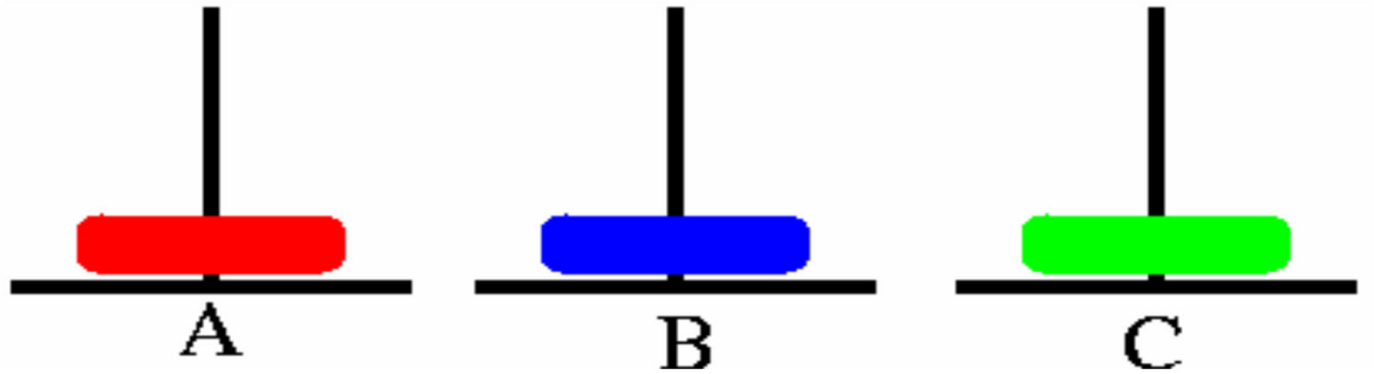


Pegs and disks problem (2 of 4)

- Now we will describe a sequence of actions that can be applied on the initial state.
- Step 1: Move A → C

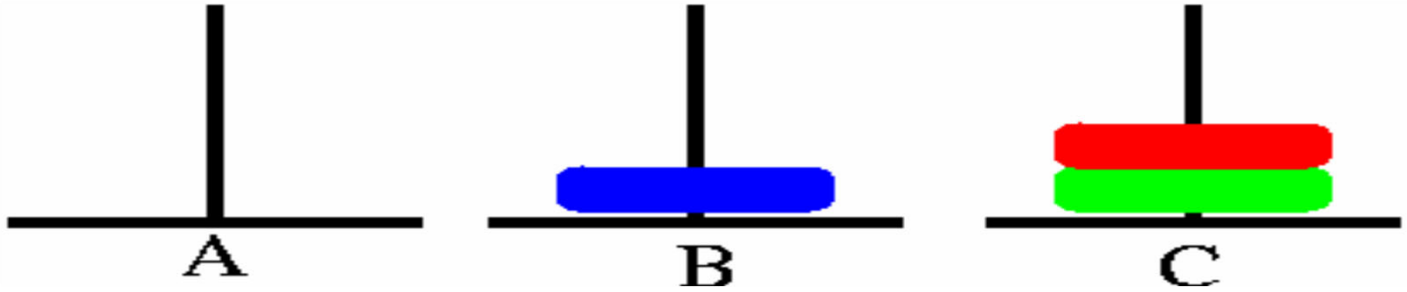


- Step 2: Move A → B

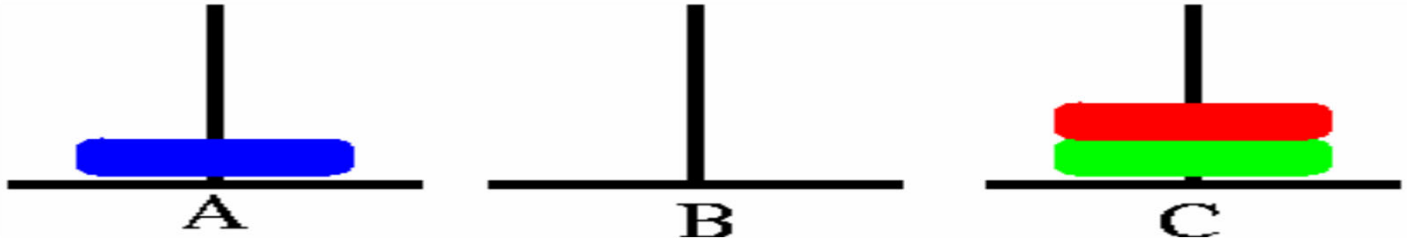


Pegs and disks problem (3 of 4)

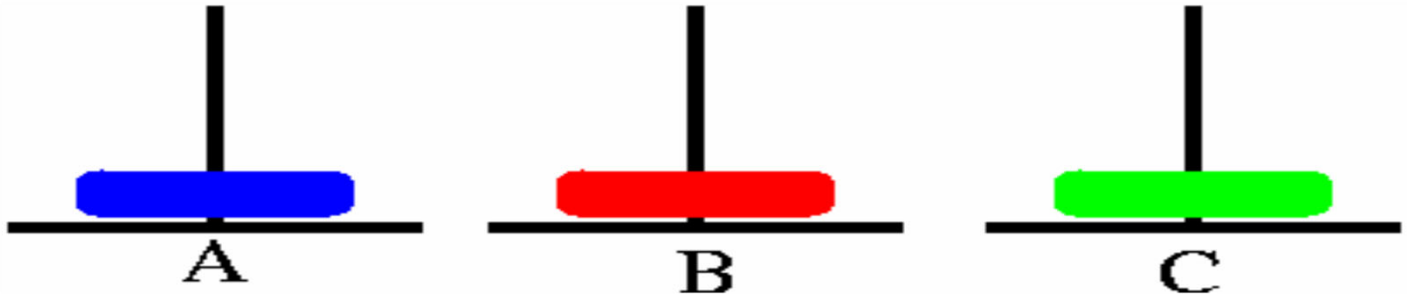
- Step 3: Move A → C



- Step 4: Move B→ A

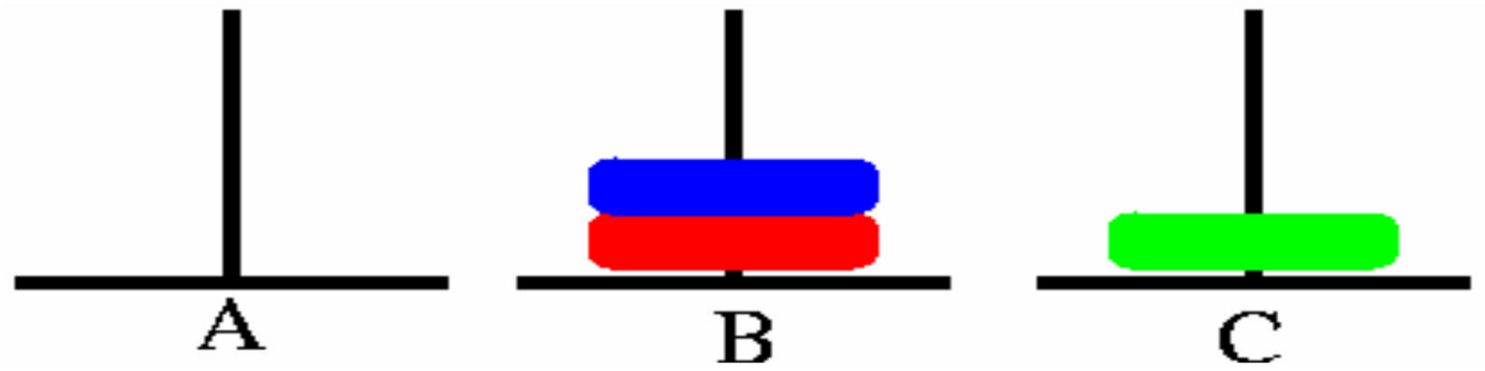


- Step 5: Move C → B

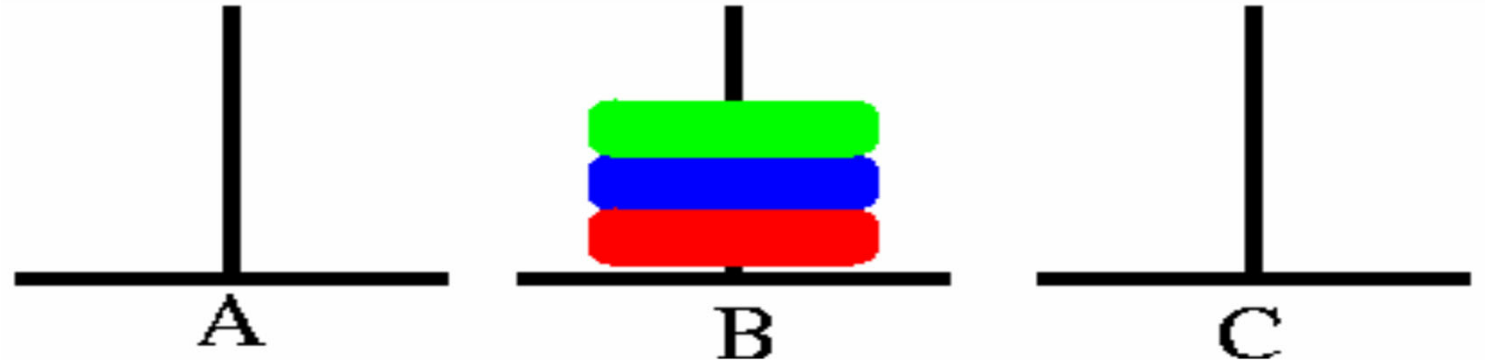


Pegs and disks problem (4 of 4)

- Step 6: Move A → B

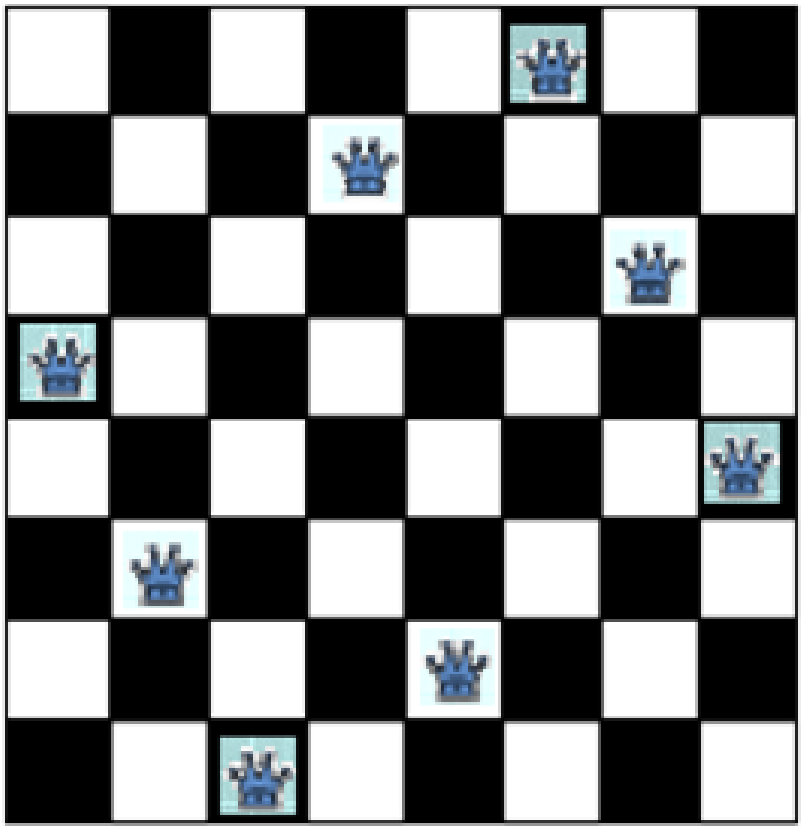


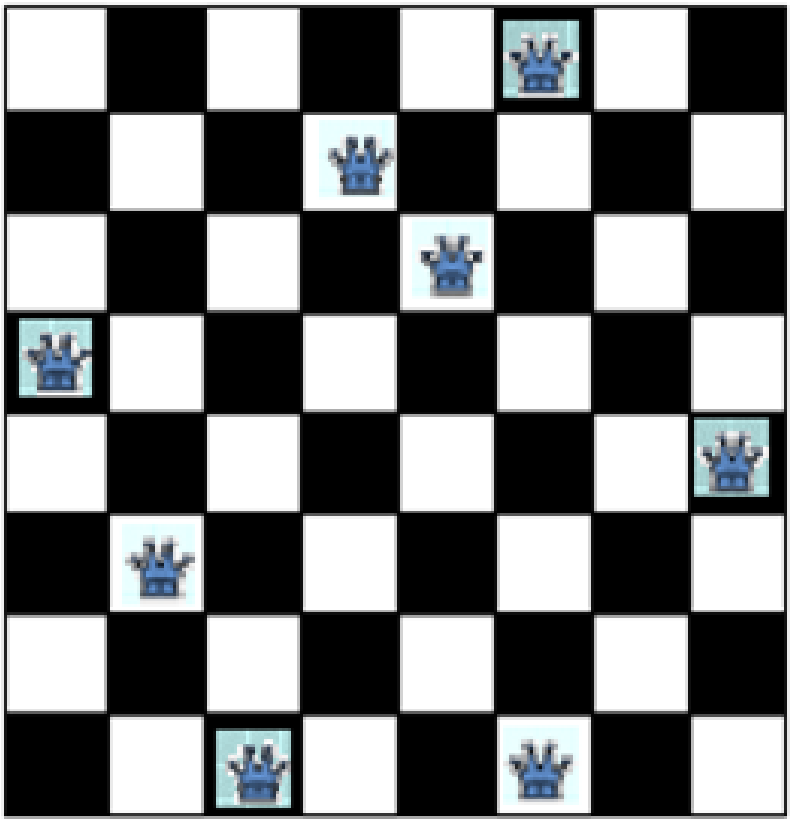
- Step 7: Move C→ B



8 queen's problem

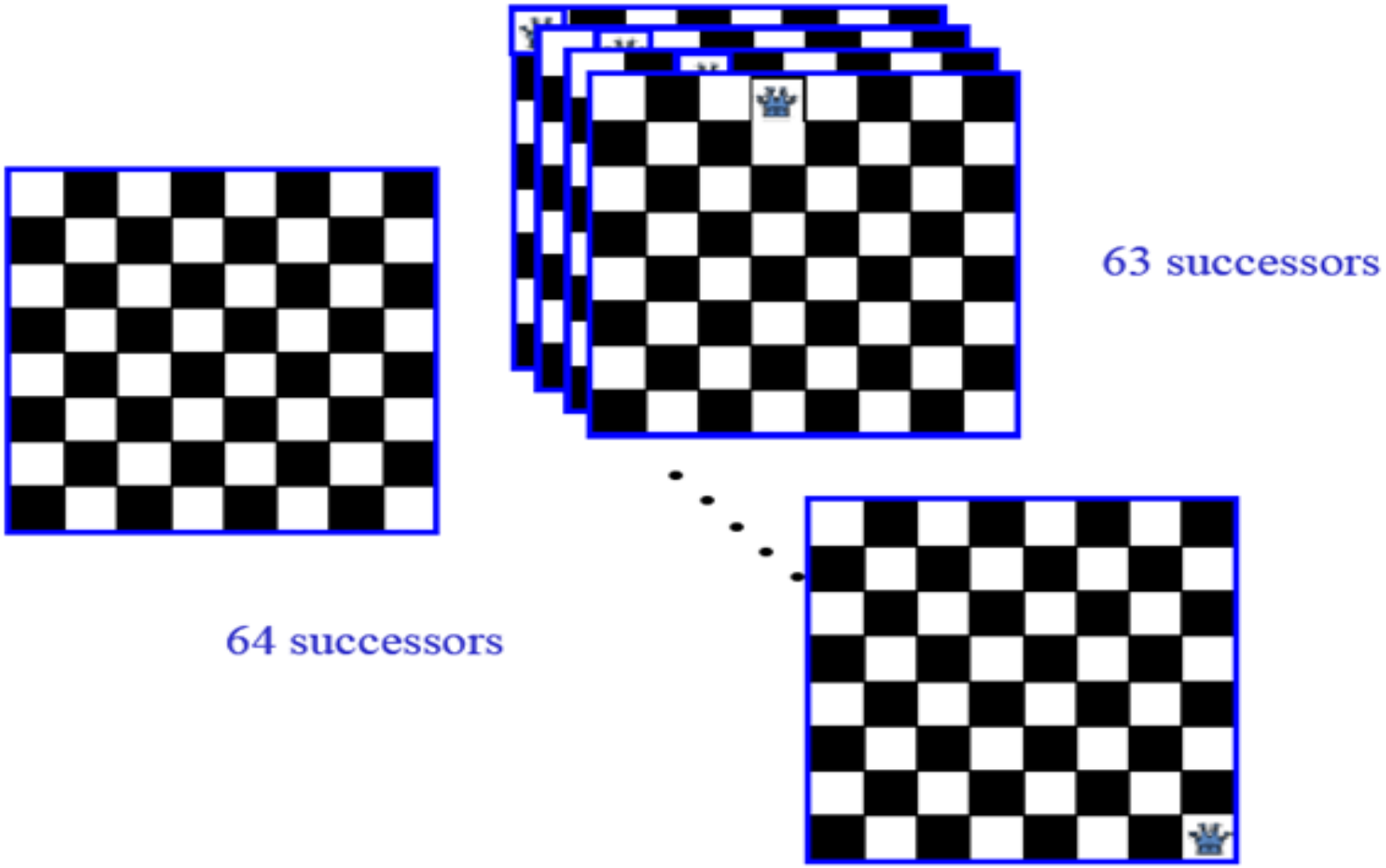
- Eight Queen's problem:





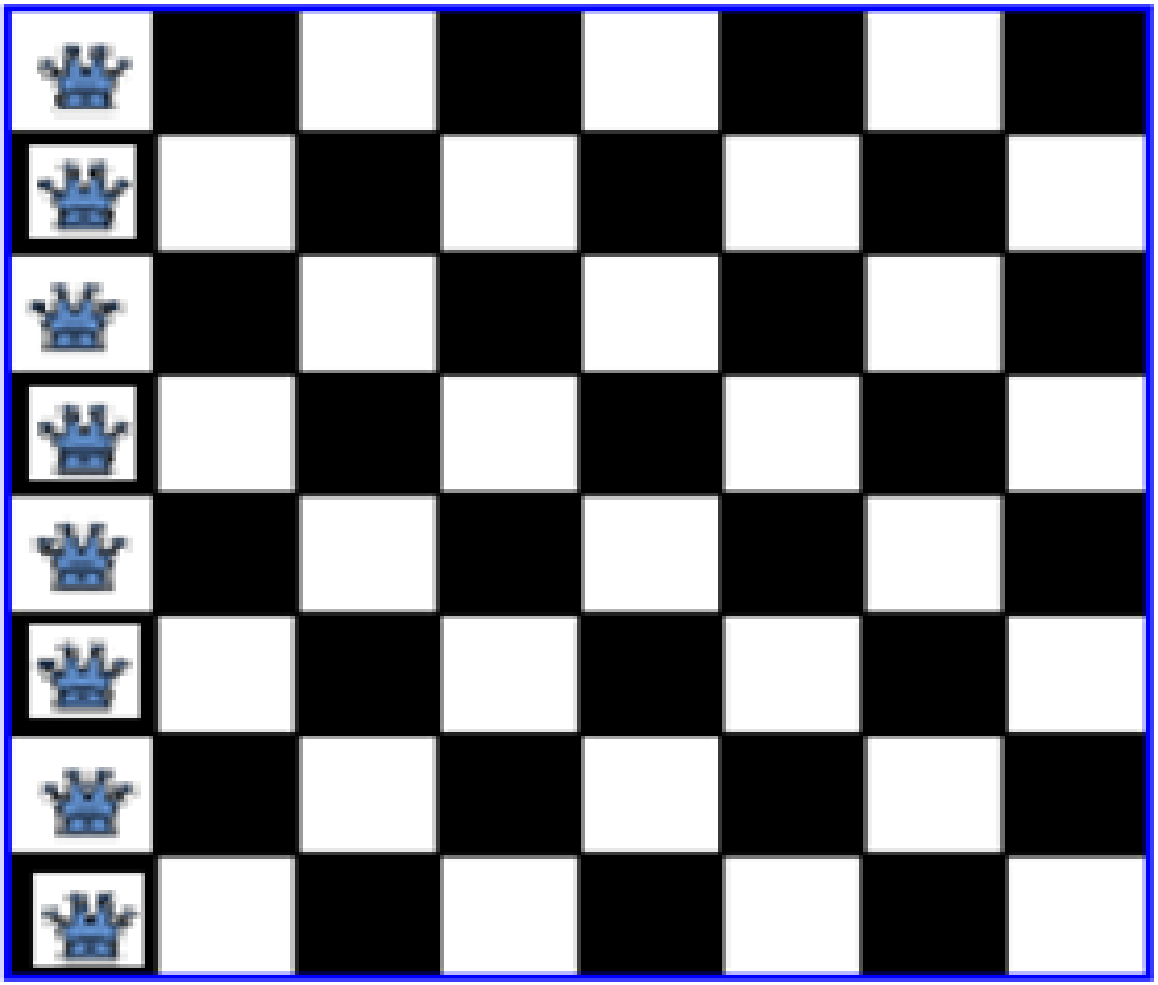
N queens problem formulation (1 of 3)

- In this problem, initially there will not be any queen in the board.

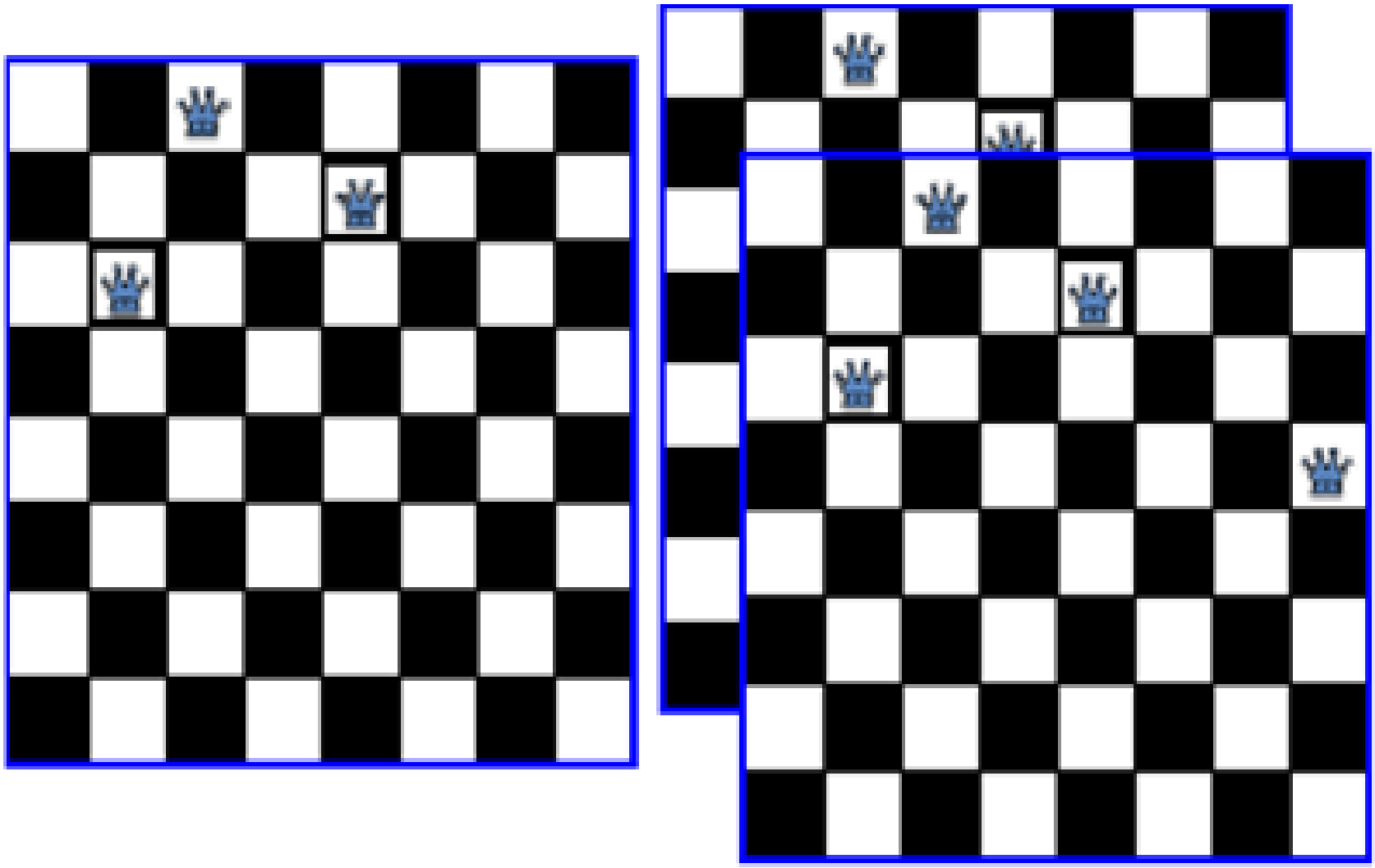


N queens problem formulation (2 of 3)

- In this problem, all the queens will be at column 1 as shown in the figure:



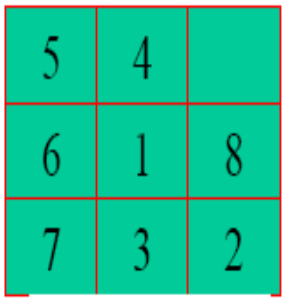
N queens problem formulation (3 of 3)



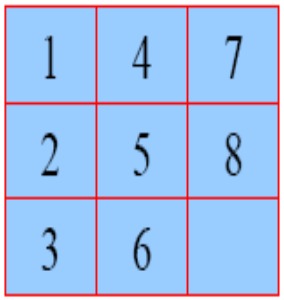
8 puzzle problem

- 8 puzzle problem:
  - States
  - Operators/Action
  - Goal Test
  - Path Cost

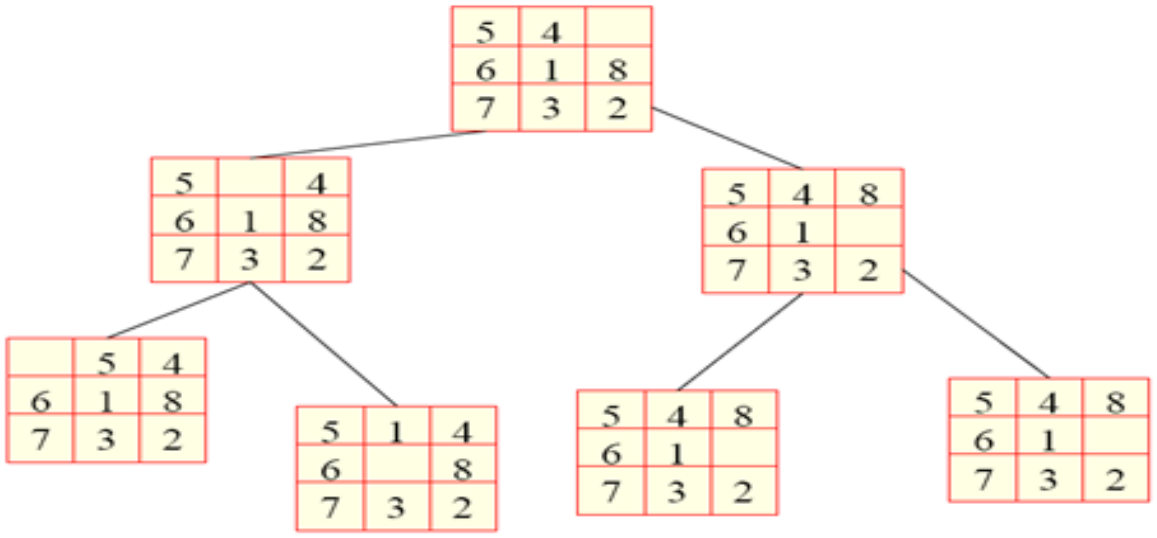
- A small portion of the state space of 8-puzzle is shown below.



Initial State



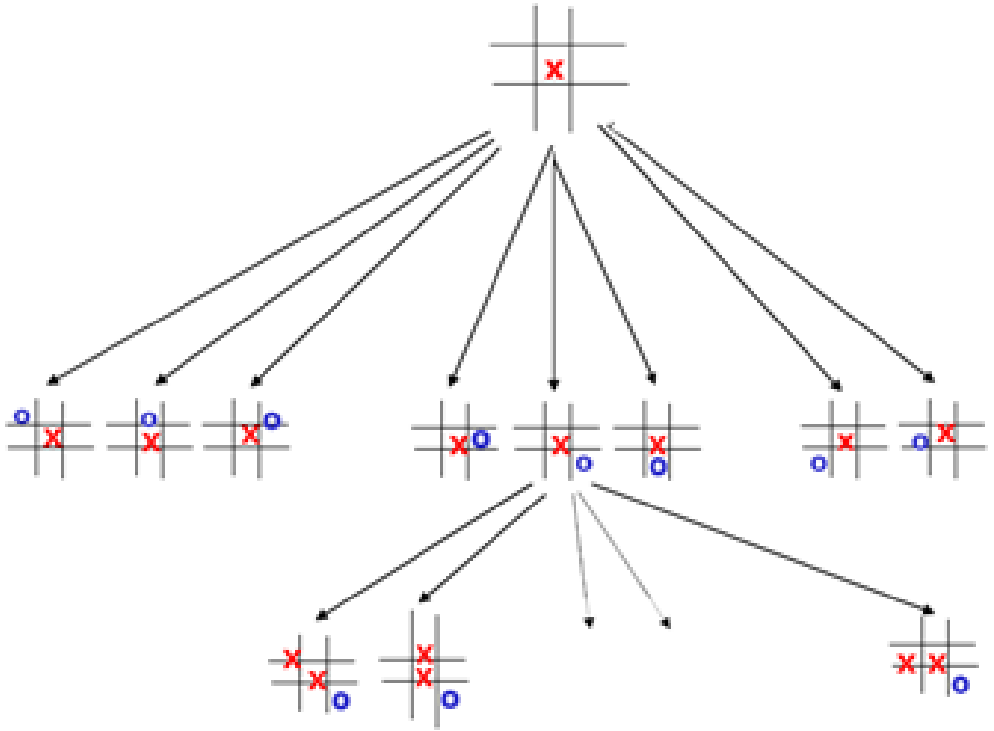
Goal State



8-puzzle partial state space

State space search example: Tic-tac-toe problem

- Tic-tac-toe is a game that involves two players who play alternately.
- Player one puts a X in an empty position.
- Player 2 places an O in an unoccupied position.
- The player who can first get three of his symbols in the same row, column or diagonal wins.
- Water jug problem:
  - Initial state.
  - Goal test.
  - Successor function.



Production systems (1 of 2)

- Productions consist of two parts: a sensory precondition (or "IF" statement) and an action (or "THEN").
- The production is triggered when production's precondition and current state matches.
- A production system consists of four basic components:
  - A set of rules.
  - One or more knowledge databases.
  - A control strategy.
  - A rule applier.
- A production system consists of rules and factors. Knowledge is encoded in a declarative from which comprises of a set of rules of the form:
  - Situation → Action; SITUATION that implies ACTION.
  - Example: IF the initial state is a goal state THEN quit.

Production systems (2 of 2)

- The major components of an AI production system are:
  - A global database.
  - A set of production rules and
  - A control system.

- Advantages of production systems:
  - Provide an excellent tool for structuring AI programs.

- Disadvantages of Production Systems:
  - Very difficult to analyze the flow of control within a production system.

- Production systems describe the operations that can be performed in a search for a solution to the problem:
  - Monotonic production system.
  - Commutative Production system.
  - Partially Commutative, Monotonic.

Commutative production system

- A Commutative production system is a production system that is both monotonic and partially commutative.

- Partially Commutative, Monotonic:
  - Useful for solving ignorable problems.
  - Can be implemented without backtrack capability.

- Non-Monotonic, Partially Commutative:
  - Useful for problems in which changes occur but can be reversed.
  - Example: Robot Navigation.

- Not partially Commutative:
  - Problems in which irreversible change occurs.
  - Example: chemical synthesis.

Problem characteristics

- Different characteristics that are required include the following abilities:
  - The ability to act intelligently, as a human.
  - The ability to behave following "general intelligent action."

- The AI problem is analyzed by the following key dimensions:
  - Is the problem is decomposable into set of independent smaller or easier sub problems?
  - Can solution steps be ignored or at least undone if they prove unwise?
  - Is the problem’s universe predictable?

Search paradigm

- Search is a problem-solving technique that systematically explores a space of problem states.
- Search methods can be divided into systematic and nonsystematic algorithms.
- The importance of search and their properties:
  - Many goal based agents are essentially problem solving agents.
  - We need to search for a sequence of rule applications.
  - For neural network systems, we need to search for the set of connection weights.
- Four important factors/properties to consider:
  - Completeness.
  - Optimality.
  - Time complexity.
  - Space complexity.

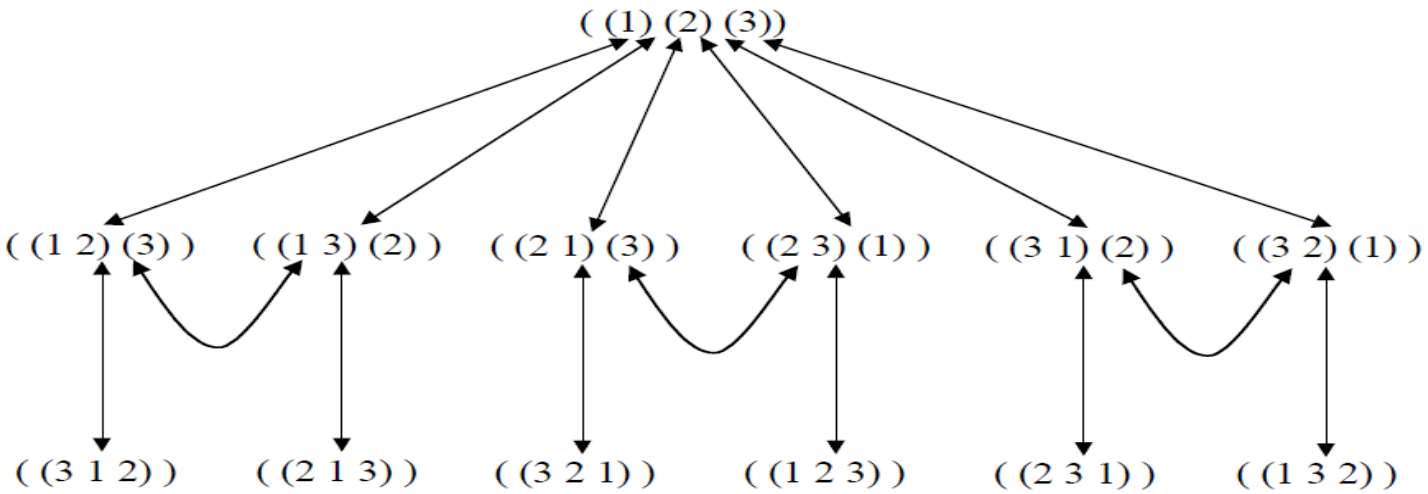
Classification of search algorithms

- The different search strategies that we will consider include the following:
  - Blind Search strategies or uninformed search:
    - Depth first search.
    - Breadth first search.
    - Iterative deepening search.
    - Iterative broadening search.
  - Informed Search.
  - Constraint Satisfaction Search.
  - Adversary Search.

- Some general terminologies:
  - State Space Representations.
  - State Space Graphs.
  - Routes through State Space.
  - Search Trees.
  - Node data structure.

General terminologies (1 of 3)

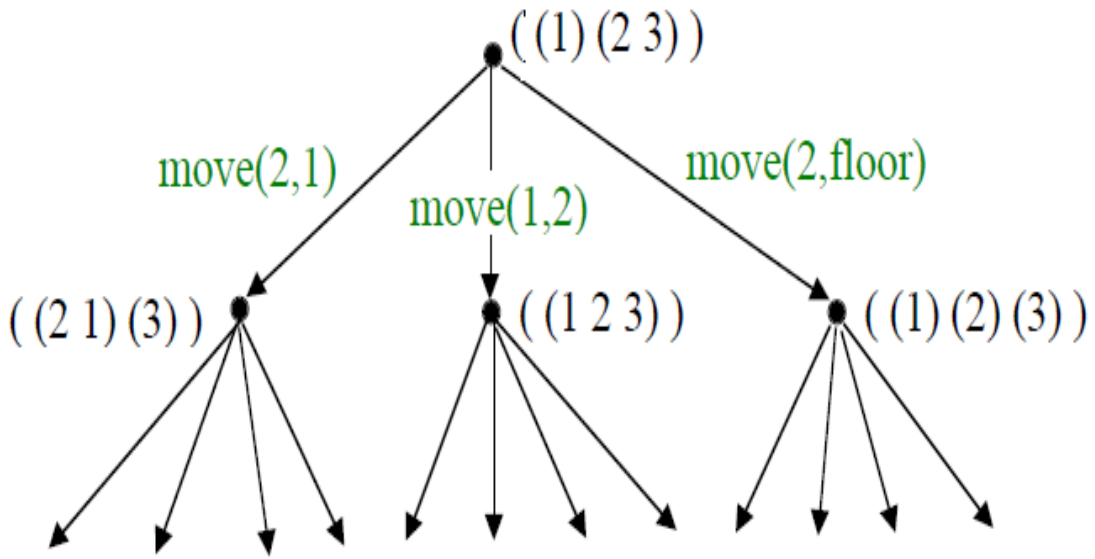
- State space graphs:
  - Example shown below.



- Routes through State Space:
  - Aim is to search for a route, or sequence of transitions, through the state space graph.

General terminologies (2 of 3)

- Search Trees:
  - The root of the search tree is the search node corresponding to the initial state as shown below:



- Search tree contains:
  - Root Node.
  - Leaf Node.
  - Ancestor/Descendant.
  - Branching factor.
  - Path.

General terminologies (3 of 3)

- Node data structure:
  - A node used in the search algorithm is a data structure.
- The nodes that the algorithm has generated are kept in a data structure called OPEN
- Expanding a node from open results in a closed node.
- The search process constructs a search tree, where:
  - root is the initial state and
  - Leaf nodes are nodes

Uninformed search algorithms

- Types of uniformed search algorithms:
  - Breadth First Search (BFS).
  - Depth First Search (DFS).
  - Depth Limited Search (DLS).
  - Depth First Iterative Deepening Search.
  - Bi-Directional Search.

Breadth first search

- BFS expands the leaf node with the lowest path cost.
- Algorithm and BFS tree is as shown below.

Breadth first search

Let fringe be a list containing the initial state

Loop

if fringe is empty return failure

Node ←remove-first (fringe)

if Node is a goal

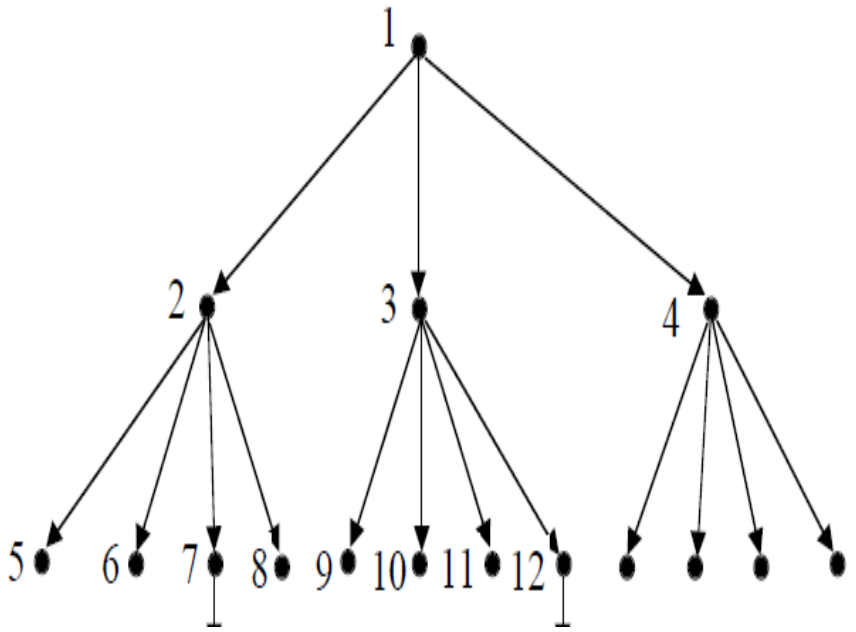
then return the path from initial state to Node

else generate all successors of Node, and

(merge the newly generated nodes into fringe)

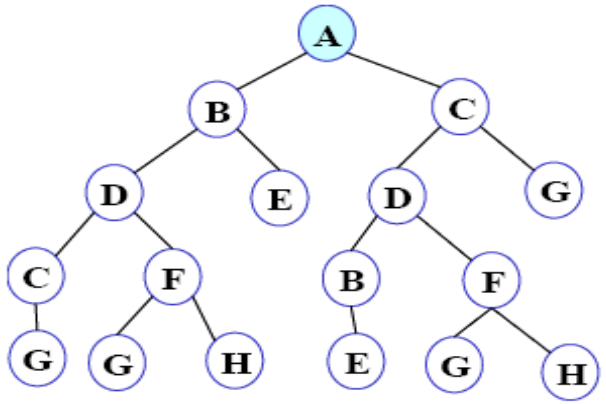
add generated nodes to the back of fringe

End Loop

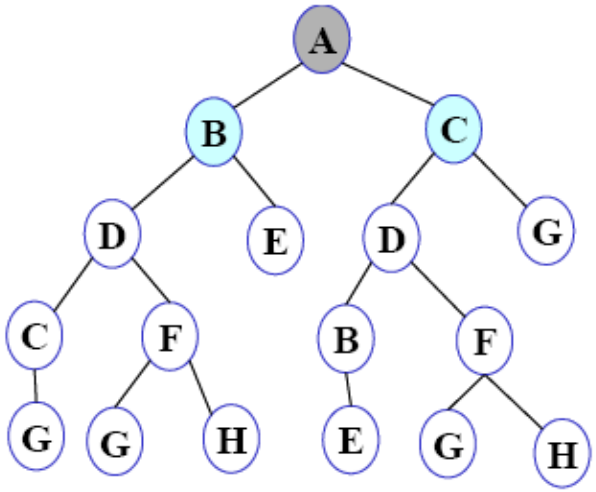


BFS illustration (1 of 3)

- We will now consider the search space in the following figure:
- Step 1: initially fringe contains only one node corresponding to the source state A.

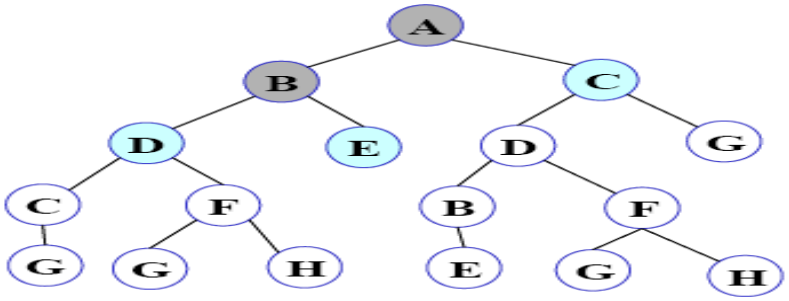


- Step 2: A is removed from fringe.

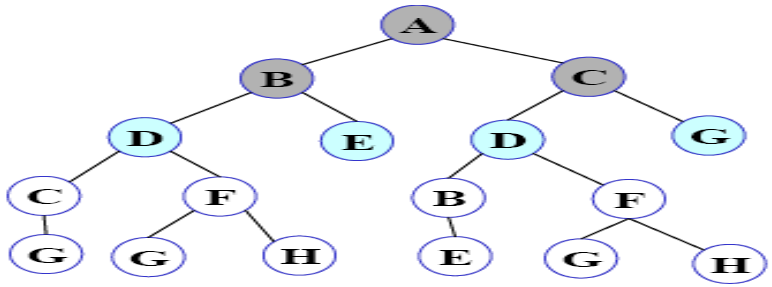


BFS illustration (2 of 3)

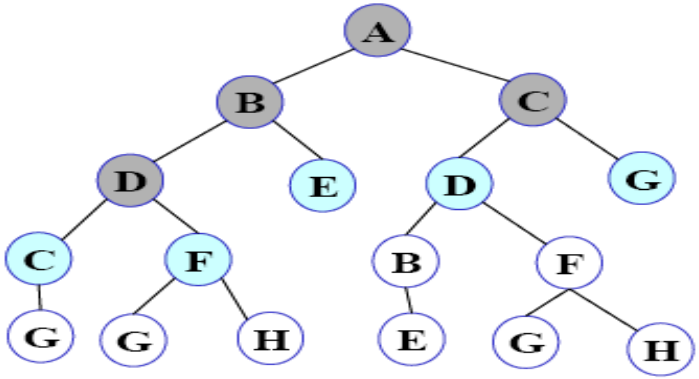
- Step 3: node B is removed from fringe and is expanded.



- Step 4: node C is removed from fringe and is expanded.

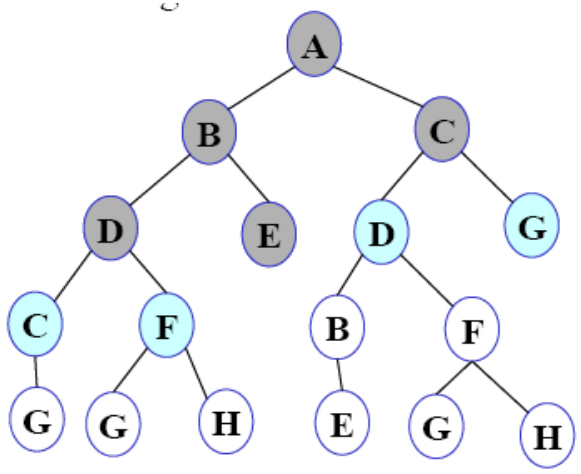


- Step 5: node D is removed from fringe.

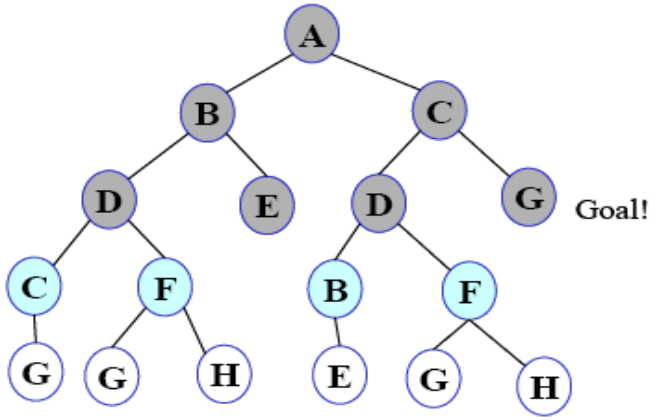


BFS illustration (3 of 3)

- Step 6: node E is removed from fringe.



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



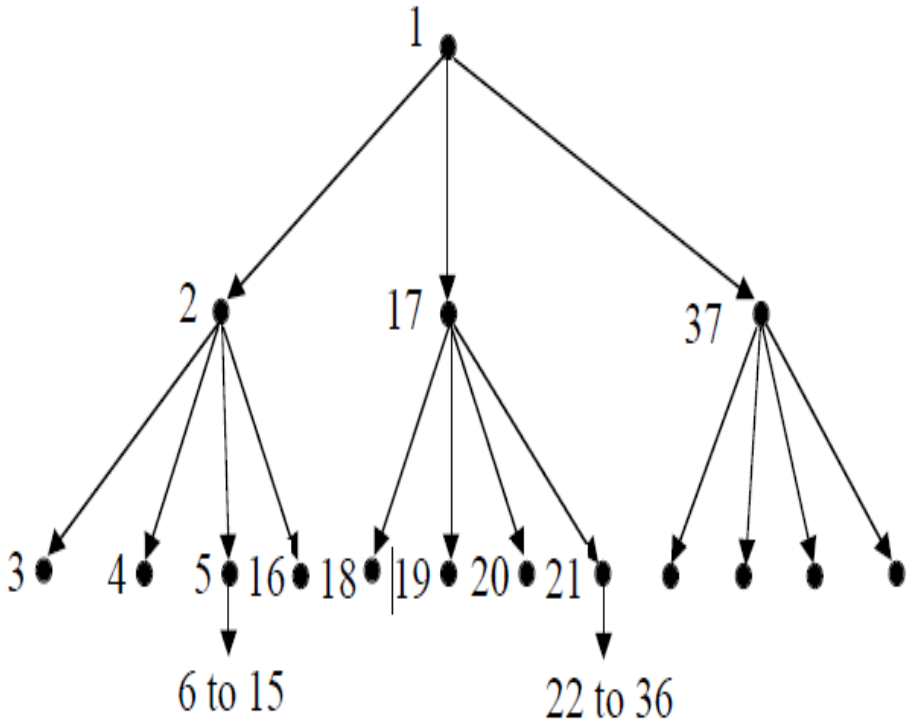
- Step 8: g is selected for expansion.

Depth first search

- DFS expands the leaf node with the highest path cost.
- Algorithm and DFS tree are shown below:

Depth First Search

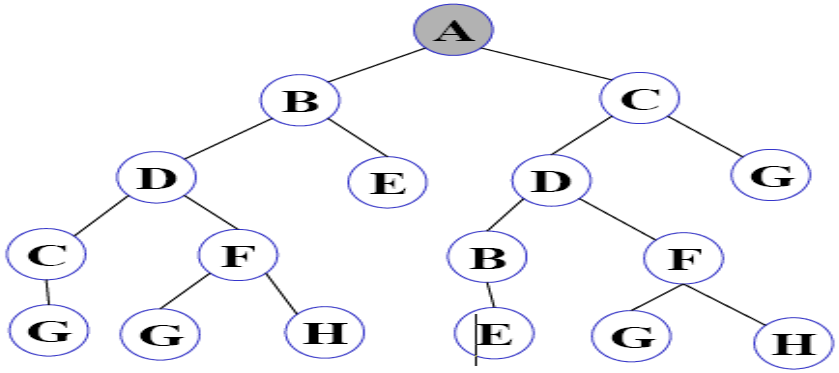
Let fringe be a list containing the initial state
Loop
if fringe is empty return failure
Node←remove-first (fringe)
if Node is a goal
then return the path from initial state to Node
else
generate all successors of Node, and
merge the newly generated nodes into fringe
add generated nodes to the front of fringe
End Loop



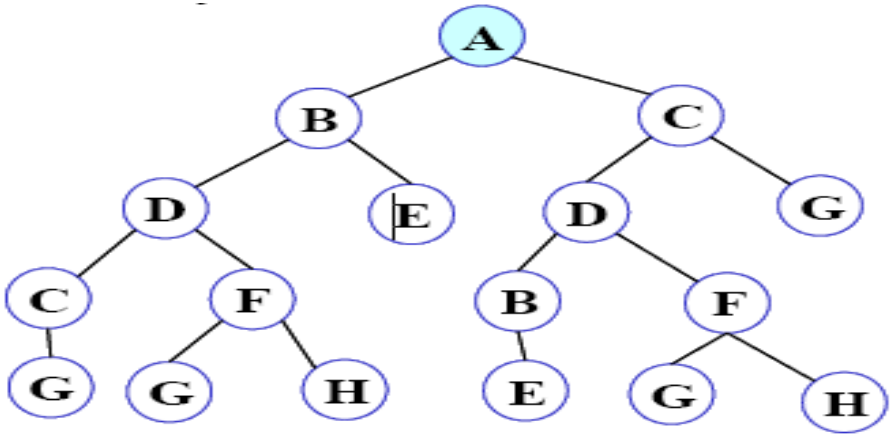
DFS illustrations (1 of 3)

- Let us now run depth first search on the search space for the following graph.

- Step 1: initially fringe contains only the node for A.

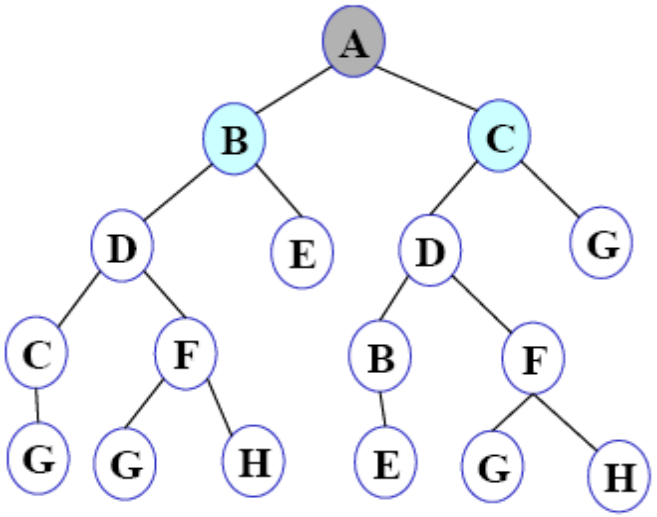


- Step 2: A is removed from fringe.

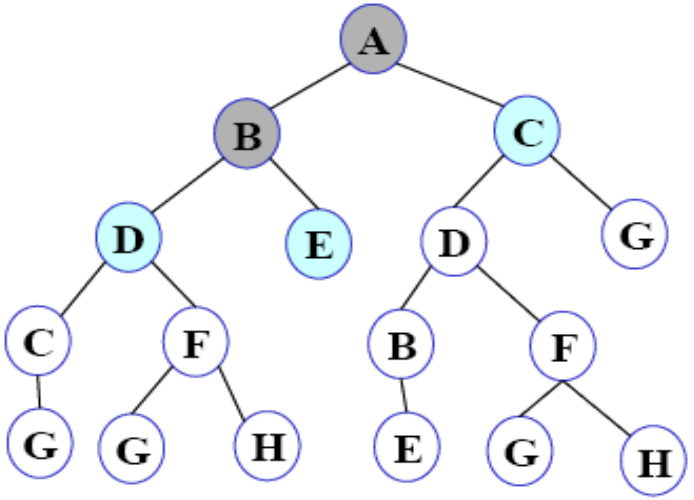


DFS illustrations (2 of 3)

- Step 3: Node B is removed from fringe.

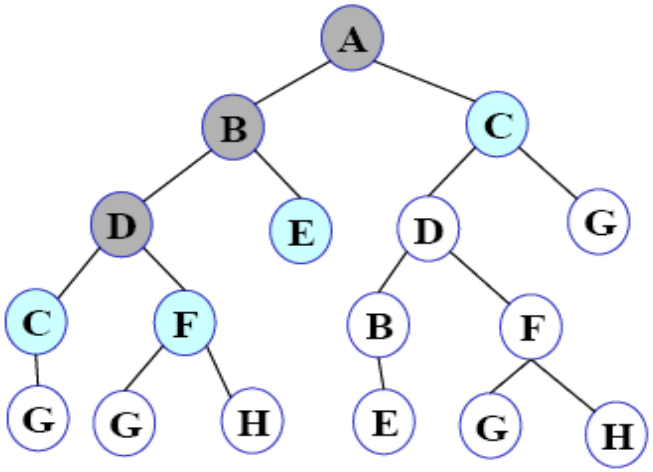


- Step 4: Node D is removed from fringe.

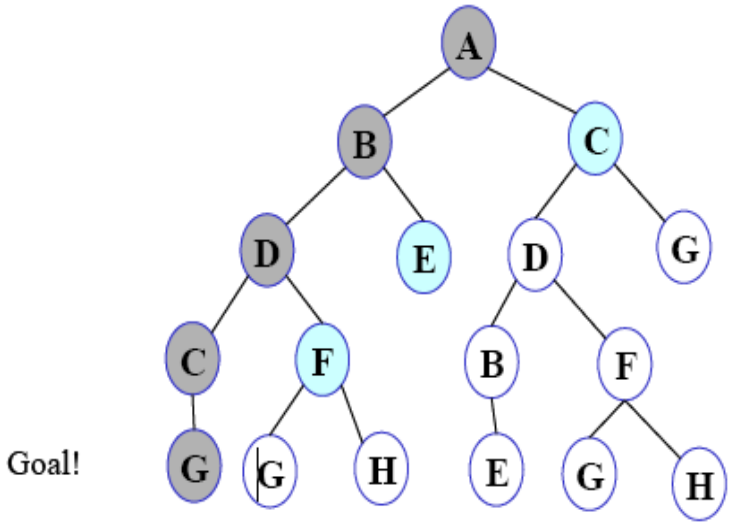


DFS illustrations (3 of 3)

- Step 5: Node C is removed from fringe.



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



Goal!

Depth limited search (DLS)

- DLS is a variation of DFS.
- If we put a limit *l* on how deep a depth first search can go, then the search will terminate.

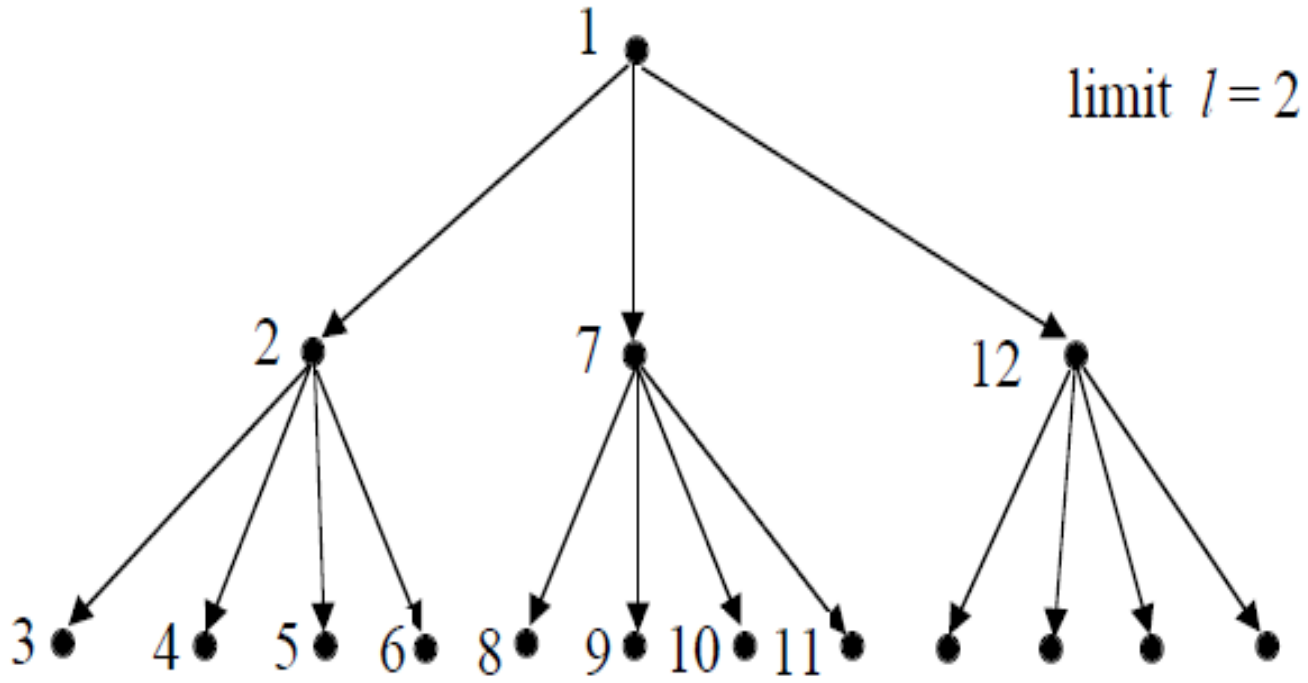
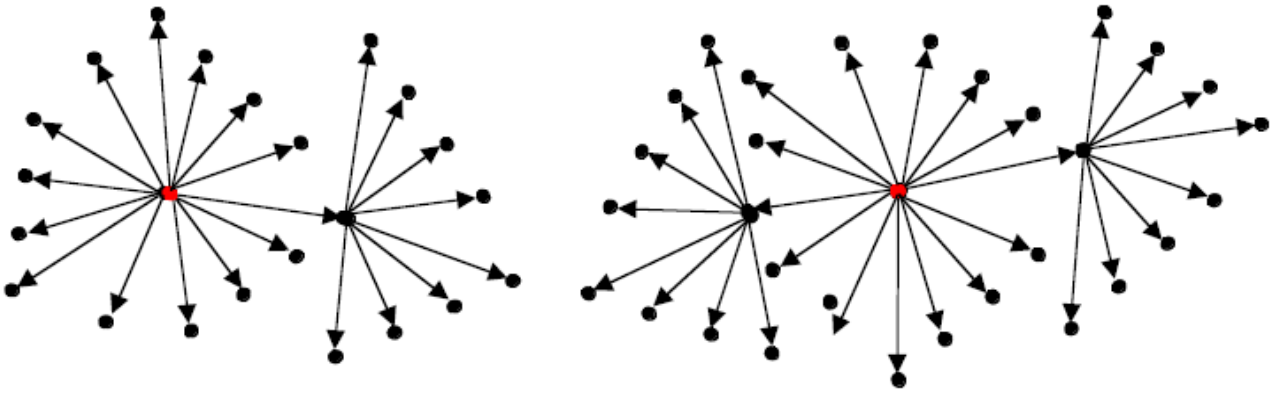


Figure: DLS tree

Depth first iterative deepening search and bi-directional search

- Depth first iterative deepening search (DFIDS):
  - DFIDS is a variation of DLS.
- Bi-directional search (bds):
  - The idea behind bi-directional search is to search simultaneously both forward and backwards.
- Repeated states:
  - The possibility that we will waste time by expanding states that have already been expanded.
- Fig. Bi-directional search tree:



Comparing the uninformed search algorithms

- We can now summarize the properties of our five uninformed search strategies:

Strategy	Complete	Optimal	Time Complexity	Space Complexity
BFS	Yes	Yes	$O(b^d)$	$O(b^d)$
DFS	No	No	$O(b^m)$	$O(b^m)$
DLS	If $l \geq d$	No	$O(b^l)$	$O(b^l)$
DFIDS	Yes	Yes	$O(b^d)$	$O(b^d)$
BDS	Yes	Yes	$O(b^{d/2})$	$O(b^{d/2})$

Informed search algorithms

- Informed search uses some kind of evaluation function.
- Heuristic is a rule of thumb that probably leads to a solution.
- Heuristic Information:
  - Information about the problem includes the nature of states, cost of transforming etc.
  - Often expressed in the form of heuristic evaluation function.
- Some of the heuristic search techniques are:
  - Pure Heuristic Search.
  - A* algorithm.
  - Iterative-Deepening A*.

Heuristic search techniques

- Heuristic search is an AI search technique.
- In artificial intelligence, heuristic search has both general meaning, and specialized technical meaning.
- Some of the variety of heuristic search are:
  - Generate and test.
  - Hill climbing.
  - Means ends analysis.

Generate-and-test

- The generate-and-test strategy is the simplest of all the approaches.

- The following is an algorithm for generate-and-test:
  - Generating a particular point in problem space.
  - Generating a path form start state.
  - Test to see if this is the actual solution by comparing
  - If a solution has been found, quit. Otherwise return to step 1.

- The generate-and-test is a:
  - It is a depth first search.
  - It is an exhaustive search of the problem space.
  - Operate by generating solutions randomly.

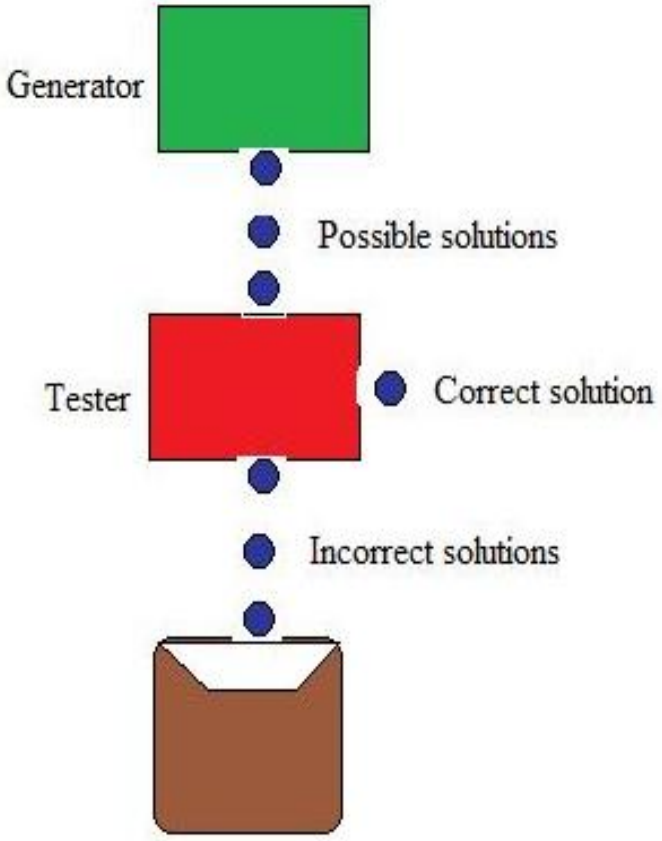


Figure: generate-and-test procedure

Hill climbing

- Hill climbing is a mathematical optimization technique.
- Hill climbing is good for finding a local optimum solution.
- It is used widely in artificial intelligence.
- Hill climbing can often produce a better result than other algorithms.
- Computation of heuristic function can be done.
- Hill climbing is often used when a good heuristic function is available.

Algorithm for simple hill climbing

- Evaluate the initial state
- Loop until a solution
- Algorithm for steepest-ascent hill climbing
- Simulated annealing

Best first search

- Best-first search is a search algorithm which explores a graph by expanding the most promising node chosen according to a specified rule.
- The a* search algorithm is an example of best-first search.
- Best-first algorithms are often used for path finding in combinatorial search.
- Algorithm:

```
OPEN = [initial state]
CLOSED = []
while OPEN is not empty
do
  1. Remove the best node from OPEN, call it n, add it to CLOSED.
  2. If n is the goal state, backtrace path to n (through recorded parents) and return
     path.
  3. Create n's successors.
  4. For each successor do:
        a. If it is not in CLOSED: evaluate it, add it to OPEN, and record its parent.
        b. Otherwise: change recorded parent if this new path is better than previous
           one.
done
```

Knowledge representation (1 of 2)

- Knowledge representation (KR) is an area of artificial intelligence research.
- Knowledge representation (KR) research involves analysis of how to reason accurately and effectively.
- The fundamental goal of knowledge representation is to facilitate reasoning, inferencing, or drawing conclusions.
- Knowledge representation in terms of five distinct roles:
  - It is most fundamentally a surrogate.
  - It is a set of ontological commitments.
  - It is a fragmentary theory of intelligent reasoning.
  - It is a medium for pragmatically efficient computation.
  - It is a medium of human expression.

Knowledge representation (2 of 2)

- Characteristics of knowledge representation:
  - Coverage.
  - Understandable by humans.
  - Consistency.
  - Efficient.
  - Easiness for modifying and updating.
  - Supports the intelligent activity which uses the knowledge base.

- Knowledge representation techniques:
  - Lists.
  - Trees.
  - Semantic networks.
  - Schemas.
  - Rule-based representations.
  - Logic-based representations.

Knowledge representation languages

- Programming is a process of representing knowledge.
- A KR language must unambiguously represent any interpretation of a sentence (logical adequacy), have a method for translating from natural language to that representation, and must be usable for reasoning.
- The main features of KR language are:
  - Object-oriented.
  - Generalization/specialization.
  - Reasoning.
  - Classification.
- Object orientation and generalization help to make the represented knowledge more understandable to humans.
- Production systems allow for the simple and natural expression of if-then rules.

Framework for knowledge representation

- Domain modelling:
  - Domain modelling is the field in which the application of KR to specific domains is studied and performed.
- Ontological analysis:
  - Ontological analysis is the process of defining this part of the model.
- In general, ontology consists of three parts:
  - Concept definitions.
  - Role definitions.
  - Inference definitions.
- A role definition may have up to three parts as well:
  - The role taxonomy.
  - Role inverses.
  - Role restrictions.
- Classic:
  - Classic is a frame-based knowledge representation language that belongs to the family of description logics.

Knowledge representation schemes

- There are four types of Knowledge representation namely Relational, Inheritable, Inferential, and Declarative/Procedural:

  - Relational based knowledge representation scheme.
  - Inheritable knowledge representation scheme.
  - Inferential knowledge representation scheme.
  - Declarative knowledge representation scheme.
  - Procedural knowledge representation scheme.

Properties and schemes for knowledge representation

- Properties for knowledge representation systems:
  - Representational adequacy.
  - Inferential adequacy.
  - Inferential efficiency.
  - Acquisition efficiency.

- Knowledge representation schemes
  - There are four types of knowledge representation namely:
    - Relational.
    - Inheritable.
    - Inferential.
    - Declarative/procedural..

Relational based knowledge representation scheme

- The simplest way of storing facts is to use a relational method.

- This scheme is a:
  - Simple way to store facts.
  - Each fact about a set of objects is set out systematically in columns.
  - Little opportunity for inference.
  - Knowledge basis for inference engines.

- The table below shows a simple way to store facts. The facts about a set of objects are put systematically in columns.

Player	Height	Weight	Bats - Throws
Aaron	6-0	180	Right – Right
Mays	5-10	170	Right - Right
Ruth	6-2	215	Left - Left
Williams	6-3	205	Left - Right

Inheritable knowledge representation scheme

- The knowledge is embodied in the design hierarchies found in all the three domains.
- The inheritance is a powerful form of inference, but not adequate.

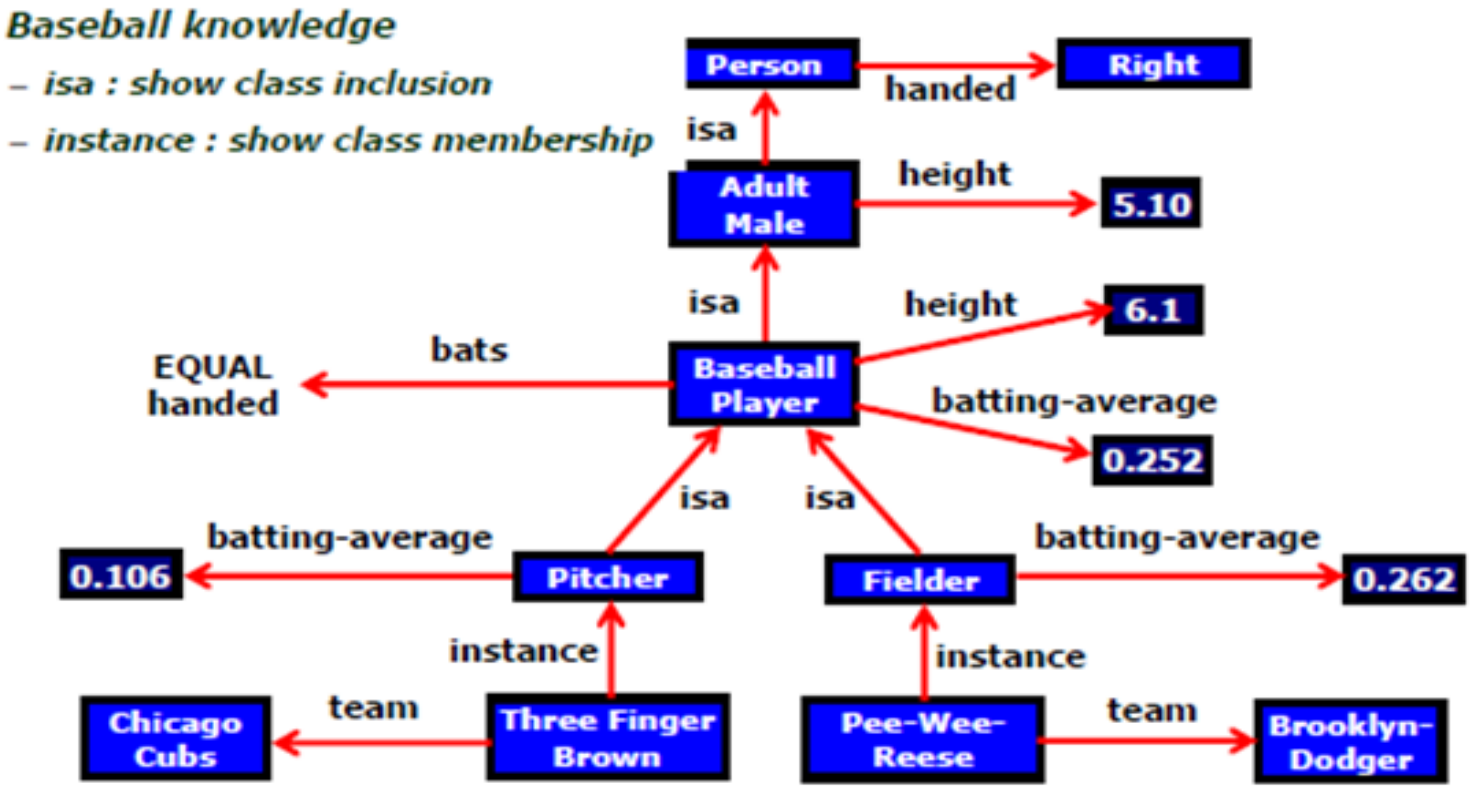


Figure: Inheritable knowledge representation (KR)

Inferential knowledge representation scheme

- This knowledge generates new information from the given information.

- This new information require analysis of the given information to generate new knowledge.
  - Example:
    - Given a set of relations and values, one may infer other values or relations.

- The symbols used for the logic operations are: " → " (implication), " ¬ " (not), " ∨ " (or), " ∧ " (and), " ∀ " (for all), " ∃ " (there exists).
  - Examples of predicate logic statements:
    - "Wonder" is a name of a dog : dog (wonder).
    - All dogs belong to the class of animals : ∀ x : dog (x) → animal(x).

Declarative/procedural knowledge

- Declarative knowledge:
  - Based on declarative facts about axioms and domains.
- Procedural knowledge:
  - A mapping process between domains that specify “what to do when”.
- The procedural knowledge:
  - May have inferential efficiency.
  - Represented as small programs.
- Advantages:
  - Heuristic or domain specific knowledge can be represented.
- Disadvantages:
  - Completeness.
  - Consistency.

Planning (1 of 2)

- The term 'planning' is typically used in AI when the problem is a "real-world" problem.
- It is very difficult and usually impossible to solve real "real-world" problems using the method of state space search.
- State space search requires complete description of the states searched and requires the search to be carried out locally.
- Planning is an important activity for autonomous agents. It is used for more than just moving objects around.

Planning (2 of 2)

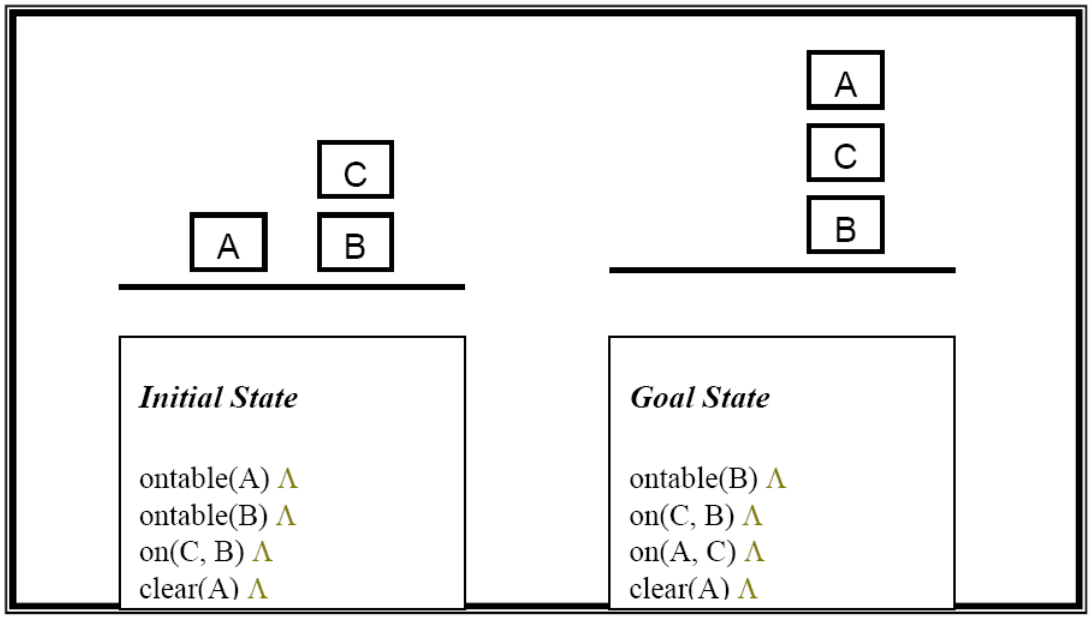
- A sequence of operator instances.
- A simple planning agent.
- Problem solving agents + knowledge-based agents = planning agents.
- The pseudo-code is as shown below:

```
function SIMPLE-PLANNING-AGENT (percept) returns an action
  static: KB, a knowledge base(includes action descriptions)
          p , a plan, initially NoPlan
          t, a counter, initially 0, indicating time
  local variables: G, a goal
                   current, a current state description

  TELL(KB, MAKE-PERCEPT-SENTENCE (percept,t))
  current <- STATE-DESCRIPTION(KB, t)
  if p = NoPlan then
      G <- ASK (KB, MAKE-GOAL-QUERY(t))
      p <- IDEAL-PLANNER(current, G, KB)
  if p = NoPlan or p is empty then action <- NoOp
  else
      action <- FIRST(p)
      p <- REST(p)
  TELL(KB, MAKE-ACTION-SENTENCE(action,t))
  t <- t+ 1
  return action
```

Representation of states, goals and actions

- States are represented by conjunctions of function-free ground literals.
- Representation of actions:
  - Strips operators consist of three components.
- Preconditions and effects are restrictive.
- The set of operators for the “box world” example problem is shown below:



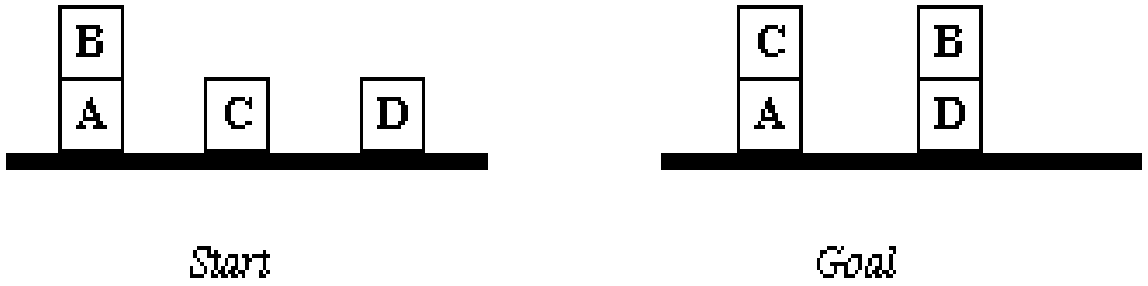
Goal stack planning (1 of 2)

- Basic idea to handle interactive compound goals uses goal stacks, here the stack contains:
  - Goals.
  - Operators -- ADD, DELETE and PREREQUISITE lists.
  - A database maintaining the current situation for each operator used.

- Goal stack planning algorithm is given in the notes.

Goal stack planning (2 of 2)

- Goal stack planning example is as shown in figure:



We can describe the *start* state:

ON (*B*, *A*) ∧ONTABLE (*A*) ∧ONTABLE(*C*) ∧ONTABLE (*D*)
∧ARMEMPTY

and *goal* state:

ON(*C*, *A*) ∧ON (*B*, *D*) ∧ONTABLE (*A*) ∧ONTABLE (*D*)

Checkpoint (1 of 2)

Multiple choice questions:

1. BFS stands for _____.
   a) Binary first search
   b) Breadth first search
   c) Block first statement
   d) All of these above

2. The major components of an AI production system are _____.
   a) A global database
   b) A set of production rules
   c) A control system
   d) All of these above

3. The generate-and-test is a _____.
   a) Depth first search
   b) Exhaustive search
   c) Binary first search
   d) Both a) and b)

Checkpoint solutions (1 of 2)

Multiple choice questions:

1. BFS stands for _____.
   a) Binary first search
   b) Breadth first search
   c) Block first statement
   d) All of these above

2. The major components of an AI production system are _____.
   a) A global database
   b) A set of production rules
   c) A control system
   d) All of these above

3. The generate-and-test is a: _____.
   a) Depth first search
   b) Exhaustive search
   c) Binary first search
   d) Both a) and b)

Checkpoint (2 of 2)

Fill in the blanks:

1. A state can have a number of_____ states.
2. _____ describes a case where a leading expert on lymph-node pathology scoffs at a program's diagnosis of an especially difficult case.
3. _____ is helpful to think of the search process as building up a search tree of routes through the state space graph.
4. _____ expands the leaf node with the highest path cost so far, and keeps going until a goal node is generated.

True or False:

1. AI is defined as the science and engineering of making machines intelligent with the help of intelligent agents' programs. True/ False
2. Computation of heuristic function can be done with infinite amount of computation. True/ False
3. Medical diagnosis programs based on probabilistic analysis have been able to perform at the level of an expert physician in several areas of medicine. True/ False

Checkpoint solutions (2 of 2)

Fill in the blanks:

1. A state can have a number of **successor** states.
2. **Heckerman** describes a case where a leading expert on lymph-node pathology scoffs at a program's diagnosis of an especially difficult case.
3. **Search tree** is helpful to think of the search process as building up a search tree of routes through the state space graph.
4. **DFS** expands the leaf node with the highest path cost so far, and keeps going until a goal node is generated.

True or False:

1. AI is defined as the science and engineering of making machines intelligent with the help of intelligent agents' programs. **True**
2. Computation of heuristic function can be done with infinite amount of computation. **False**
3. Medical diagnosis programs based on probabilistic analysis have been able to perform at the level of an expert physician in several areas of medicine. **True**

Question bank

Two mark questions:

1. Define artificial intelligence and what are its applications?
2. Define problem and problem space. Illustrate.
3. Define knowledge representation. What are its characteristics?
4. Explain search paradigm and its importance in robotics.

Four mark questions:

1. Discuss pegs and disks problem considering three disks.
2. Define BFS. Illustrate with an example.
3. Discuss any two heuristic search techniques with an example.
4. Discuss blocks of world problem with example.

Eight mark questions:

1. Explain 8 queens' problem?
2. Write about goal stack planning algorithm with examples?

Unit summary

Having completed this unit, you should be able to:

- Gain knowledge on the basics of Artificial Intelligence (AI)
- Gain an insight into the AI problems and techniques
- Learn about the state space search and production systems
- Understand the concept of problem characteristics and search paradigm
- Learn about heuristic search techniques and knowledge representation