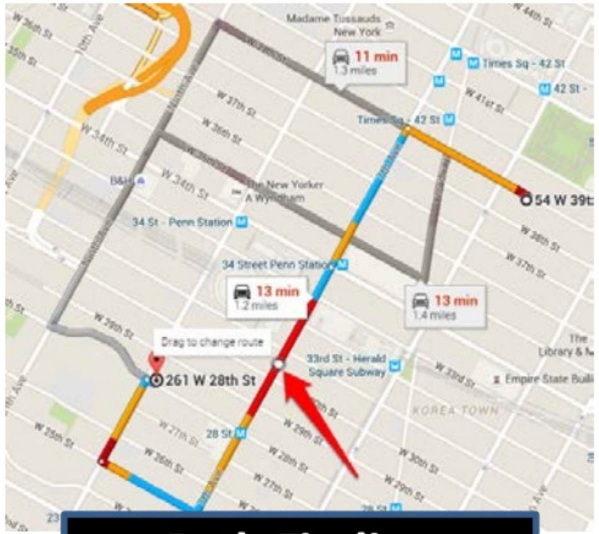
ARTIFICIAL INTELLIGENCE (AI)
STATE SPACE AND SEARCH ALGORITHMS

Shreya Ghosh

Assistant Professor, Computer Science and Engineering

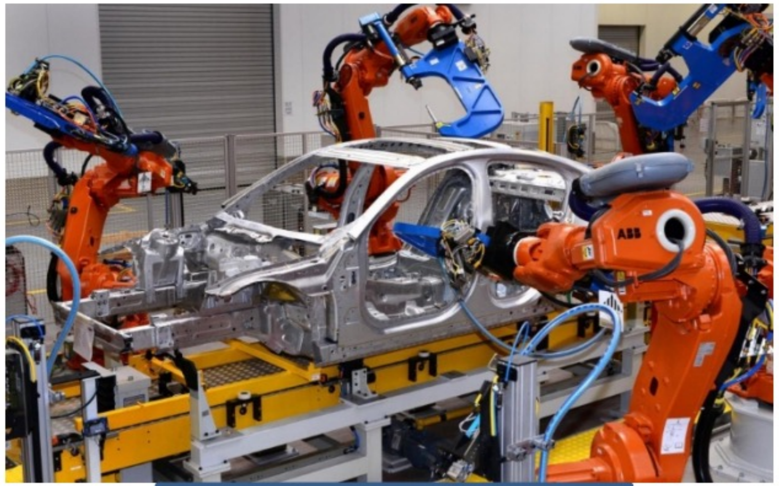
IIT Bhubaneswar

COMPLEX PROBLEMS AND SOLUTIONS



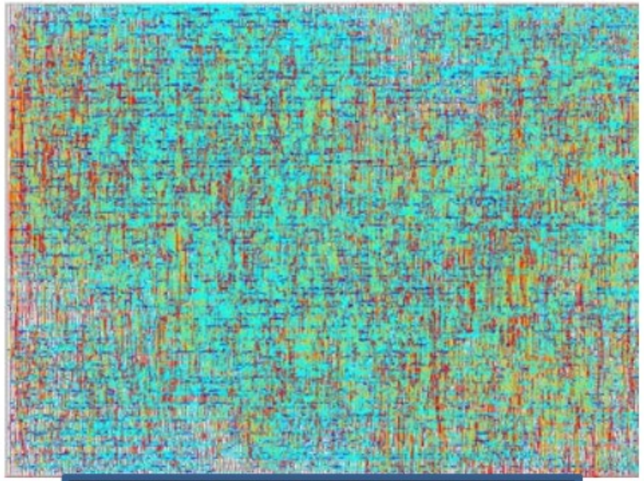
Path Finding

Chess Playing

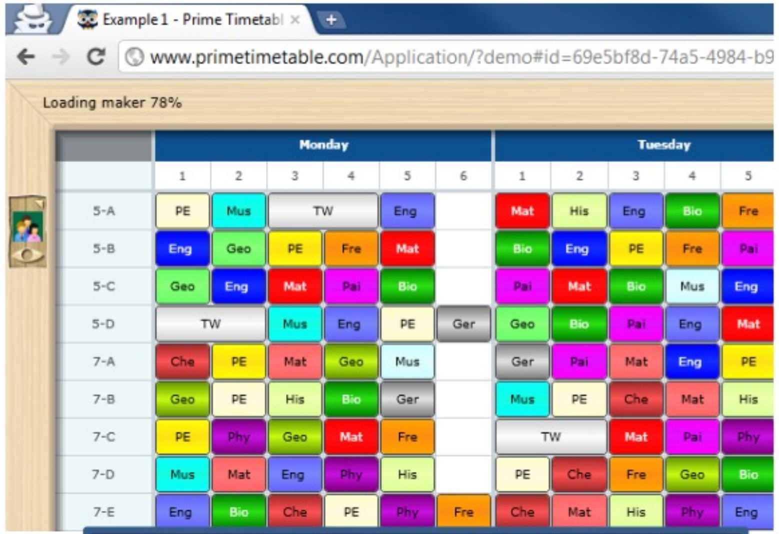


Robot Assembly

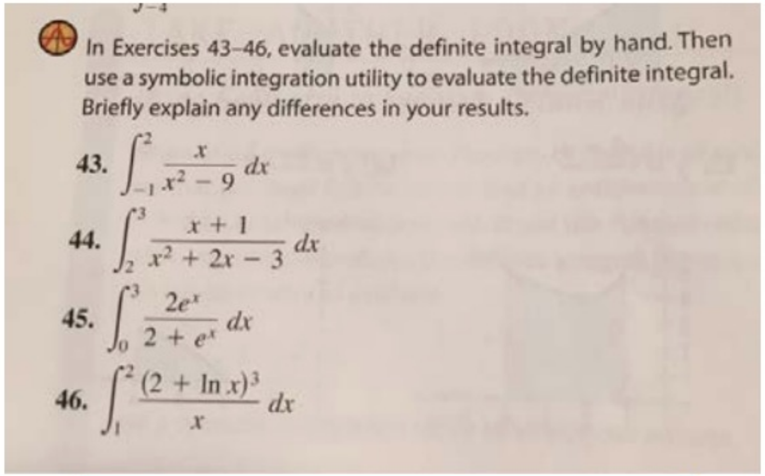
COMPLEX PROBLEMS AND SOLUTIONS



VLSI Chip Design



Time-Table Scheduling



In Exercises 43–46, evaluate the definite integral by hand. Then use a symbolic integration utility to evaluate the definite integral. Briefly explain any differences in your results.

43. $\int_{-1}^{2} \frac{x}{x^2 - 9}\,dx$

44. $\int_{2}^{3} \frac{x + 1}{x^2 + 2x - 3}\,dx$

45. $\int_{0}^{3} \frac{2e^x}{2 + e^x}\,dx$

46. $\int_{1}^{2} \frac{(2 + \ln x)^3}{x}\,dx$

Symbolic Integration

AUTOMATED PROBLEM SOLVING BY SEARCH

- Problem Statement is the Input and solution is the Output, sometimes even the problem specific algorithm or method could be the Output

- Problem Formulation by AI Search Methods consists of the following key concepts
  - Configuration or State
  - Constraints or Definitions of Valid Configurations
  - Rules for Change of State and their Outcomes
  - Initial or Start Configurations
  - Goal Satisfying Configurations
  - An Implicit State or Configuration Space
  - Valid Solutions from Start to Goal in the State Space
  - General Algorithms which SEARCH for Solutions in this State Space

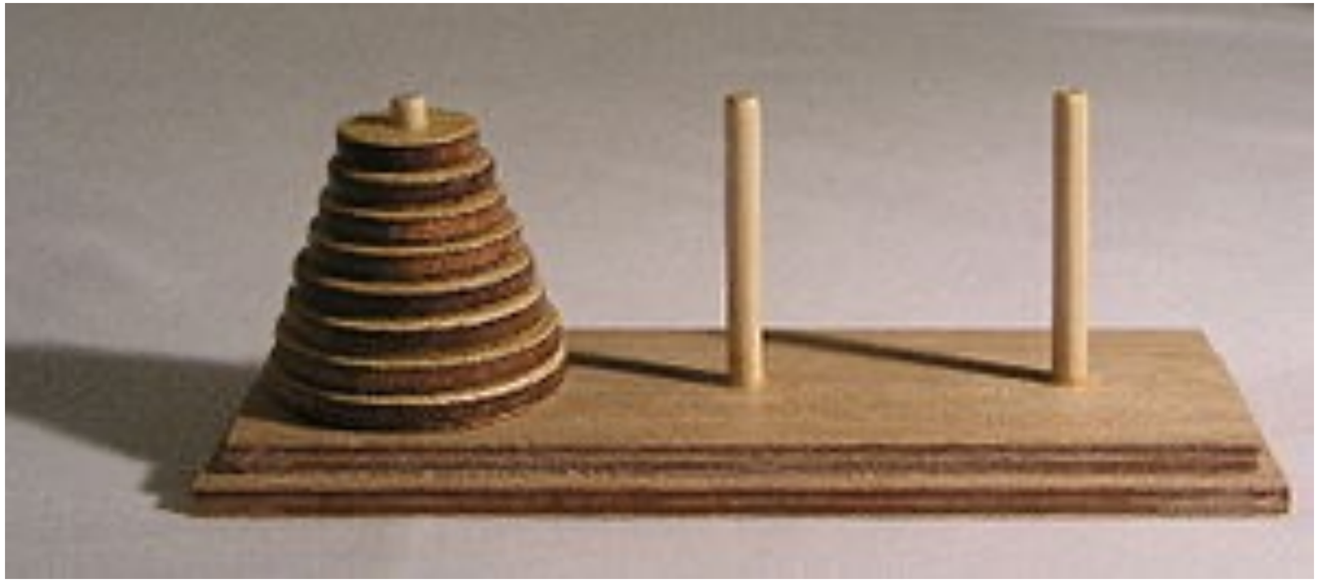
AUTOMATED PROBLEM SOLVING BY SEARCH

- Problem Formulation by AI Search Methods consists of the following key concepts
  - Configuration or State
  - Constraints or Definitions of Valid Configurations
  - Rules for Change of State and their Outcomes
  - Initial or Start Configurations
  - Goal Satisfying Configurations
  - An Implicit State or Configuration Space
  - Valid Solutions from Start to Goal in the State Space
  - General Algorithms which SEARCH for Solutions in this State Space

Size of the Implicit Space,
Capturing Domain Knowledge,
Intelligent Algorithms that work in reasonable time and Memory,
Handling Incompleteness and Uncertainty

BASICS OF STATE SPACE MODELLING

- **STATE or CONFIGURATION:**
  - A set of variables which define a state or configuration
  - Domains for every variable and constraints among variables to define a valid configuration
- **STATE TRANSFORMATION RULES or MOVES:**
  - A set of RULES which define which are the valid set of NEXT STATE of a given State
  - It also indicates who can make these Moves (OR Nodes, AND nodes, etc)
- **STATE SPACE or IMPLICIT GRAPH**
  - The Complete Graph produced out of the State Transformation Rules.
  - Typically too large to store. Could be Infinite.
- **INITIAL or START STATE(s), GOAL STATE(s)**
- **SOLUTION(s), COSTS**
  - Depending on the problem formulation, it can be a PATH from Start to Goal or a Sub-graph of And-ed Nodes
- **SEARCH ALGORITHMS**
  - Intelligently explore the Implicit Graph or State Space by examining only a small sub-set to find the solution



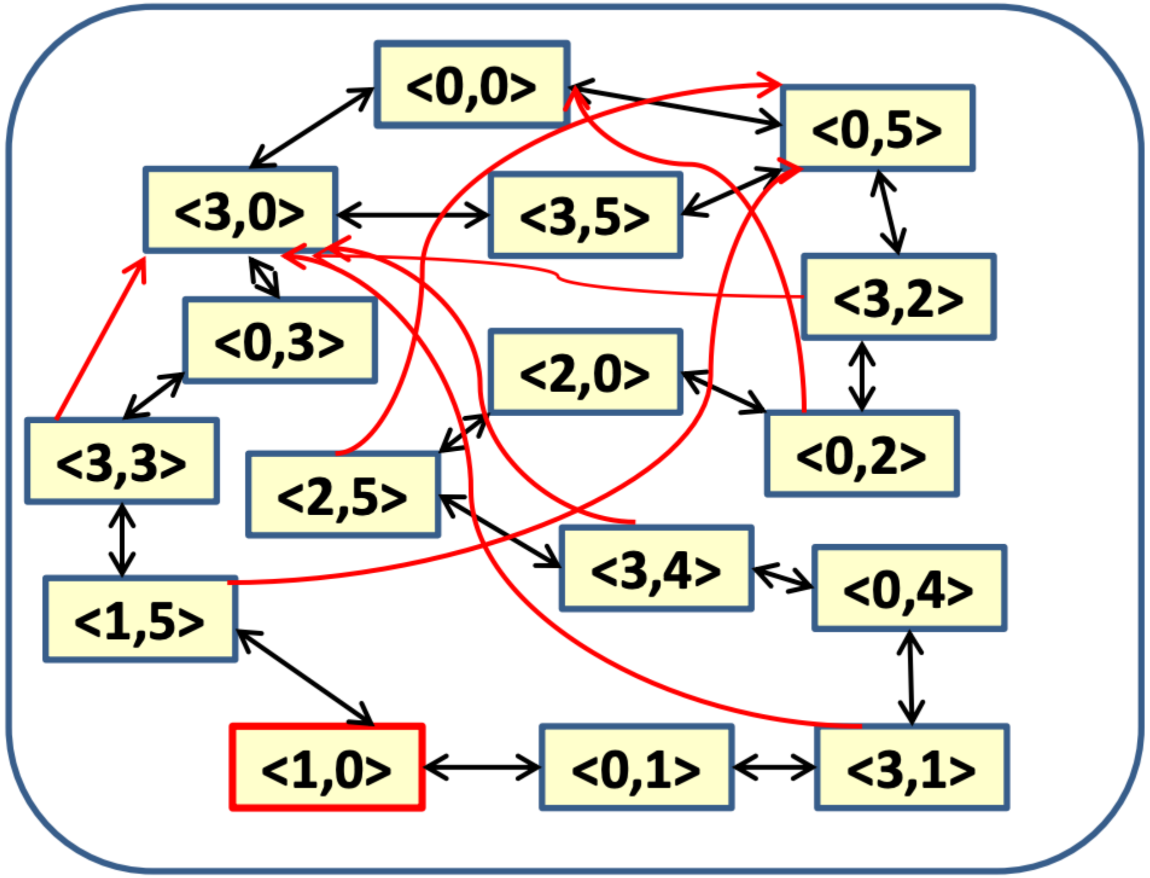
TOWER OF HANOI

TWO JUG PROBLEM

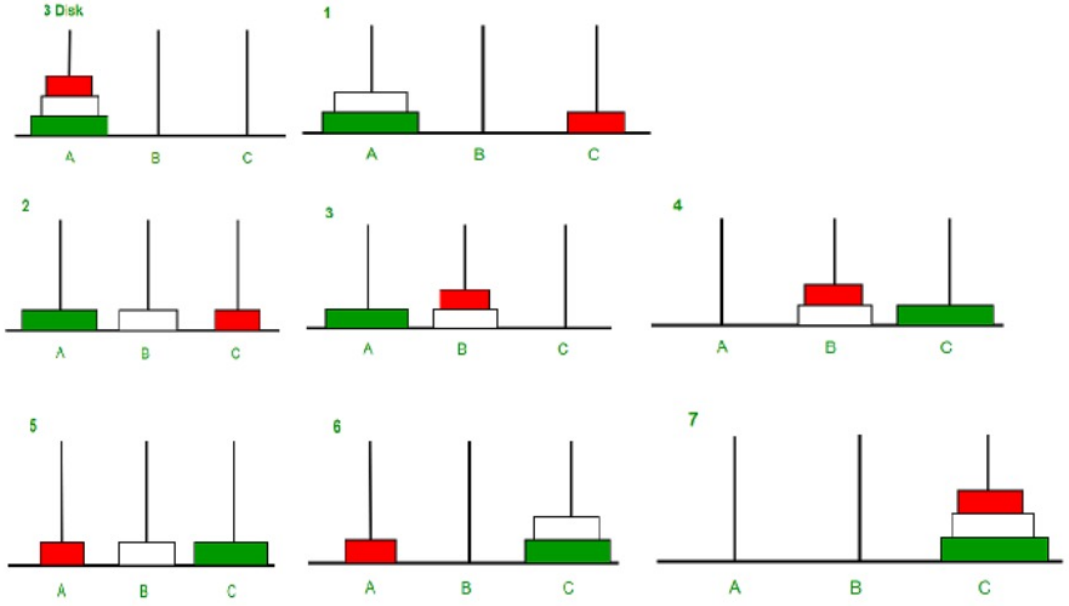
Two Jug Problem

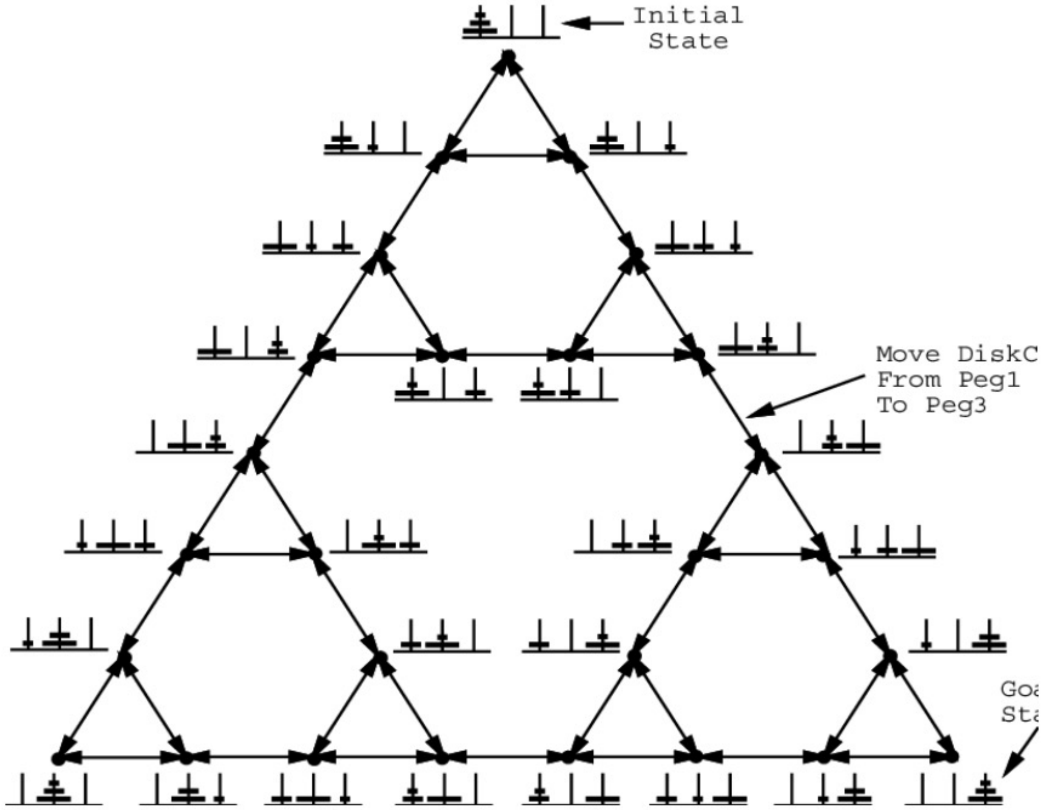
- There is a large bucket B full of water and Two (02) jugs, J1 of volume 3 litre and J2 of volume 5 litre. You are allowed to fill up any empty jug from the bucket, pour all water back to the bucket from a jug or pour from one jug to another. The goal is to have jug J1 with exactly one (01) litre of water
- State Definition: <J1, J2>
- Rules:
  - Fill (J1): <J1, J2> to <3,J2>
  - Fill (J2): <J1, J2> to <J1, 5>
  - Empty (J1), Empty (J2): Similarly defined
  - Pour (J1, J2): <J1, J2> to <X,Y>, where
    - X = 0 and Y = J1 + J2 if J1+J2 $\leq$ 5,
    - Y = 5 and X = (J1+ J2) – 5, if J1+J2 > 5
  - Pour (J2, J2): Similarly defined
- Start: <0,0>, Goal: <1,0>
- Part of State Space Shown on the right

(Not all Links shown here)



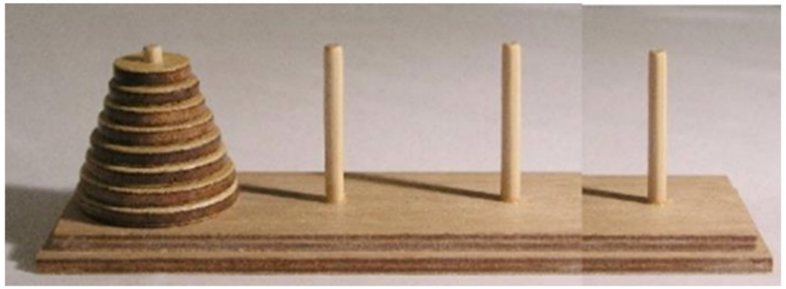
3 DISK, 3 PEG TOWER of HANOI STATE SPACE





STATES, SPACES, SOLUTIONS, SEARCH

- States
  - Full / Perfect Information and Partial Information States
- State Transformation Rules
  - Deterministic Outcomes
  - Non-Deterministic / Probabilistic Outcomes
- State Spaces As Generalized Games
  - Single Player: OR Graphs
  - Multi-Player: And / Or, Adversarial, Probabilistic Graphs
- Solutions
  - Paths
  - Sub-graphs
  - Expected Outcomes
- Costs
- Sizes
- Domain Knowledge
- Algorithms for Heuristic Search



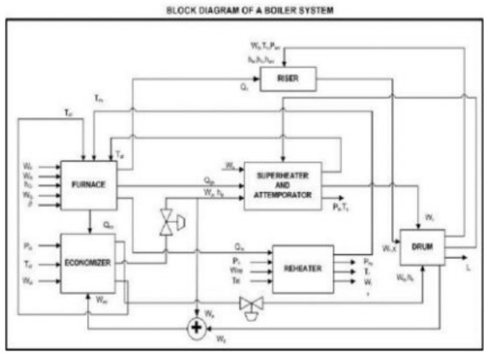
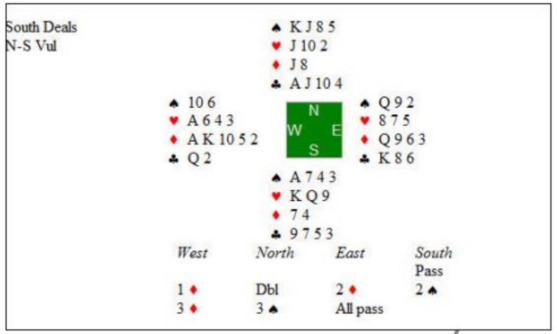


Plate Cutting



BASIC ALGORITHMS in OR GRAPHS: IDS

1. [Initialize] Initially the OPEN List contains the Start Node s. CLOSED List is Empty.
2. [Select] Select the first Node n on the OPEN List. If OPEN is empty, Terminate
3. [Goal Test] If n is Goal, then decide on Termination or Continuation / Cost Updation
4. [Expand]
   a) Generate the successors n_1, n_2, ... n_k, of node n, based on the State Transformation Rules
   b) Put n in LIST CLOSED
   c) For each n_i, not already in OPEN or CLOSED List, put n_i in the FRONT of OPEN List
   d) For each n_i already in OPEN or CLOSED decide based on cost of the paths
5. [Continue] Go to Step 2

Algorithm IDS Performs DFS Level by Level Iteratively (DFS (1), DFS (2), ... and so on)

BASIC ALGORITHMS in OR GRAPHS: BFS

1. [Initialize] Initially the OPEN List contains the Start Node s. CLOSED List is Empty.
2. [Select] Select the first Node n on the OPEN List. If OPEN is empty, Terminate
3. [Goal Test] If n is Goal, then decide on Termination or Continuation / Cost Updation
4. [Expand]
   a) Generate the successors n_1, n_2, …. n_k, of node n, based on the State Transformation Rules
   b) Put n in LIST CLOSED
   c) For each n_i, not already in OPEN or CLOSED List, put n_i in the END of OPEN List
   d) For each n_i already in OPEN or CLOSED decide based on cost of the paths
5. [Continue] Go to Step 2