Problem Reduction

Dr. Asaad Sabah Hadi

Problem Reduction

- Sometimes problems only seem hard to solve.
- A hard problem may be one that can be reduced to a number of simple problems...and, when each of the simple problems is solved, then the hard problem has been solved.
- This is the basic intuition behind the method of problem reduction.

Problem Reduction

- If we are looking for a sequence of actions to achieve some goal, then one way to do it is to use state-space search, where each node in you search space is a state of the world, and you are searching for a sequence of actions that get you from an initial state to a final state.

Problem Reduction

- Another way is to consider the different ways that the goal state can be decomposed into simpler sub-goals.
- For example, when planning a trip to London you probably don't want to search through all the possible sequences of actions that might get you to London.
- You're more likely to decompose the problem into simpler ones - such as getting to the station, then getting a train to London.

Problem Reduction

- There may be more than one possible way of decomposing the problem - an alternative strategy might be to get to the airport, fly to Heathrow, and get the tube from Heathrow into London.
- These different possible plans would have different costs (and benefits) associated with them, and you might have to choose the best plan.

Problem Reduction

- The Simple State-space Search Techniques Described In The Above Slides Could All Be Represented Using A Tree Where Each Successor Node Represents An *Alternative* Action To Be Taken.
- The Graph Structure Being Searched Is Referred To As An *Or* Graph.
- In *Or* Graph We Want To Find A Single Path To A Goal.

Problem Reduction

- This is due to the fact that we will know how to get to from a node to a goal state if we can discover how to get from that node to a goal state along any one of the branches leaving it.

- To represent problem reduction techniques we need to use an AND-OR graph/tree.

Problem Reduction

- Problem reduction technique using AND-OR graph is useful for representing a solution of a problems that can be solved by decomposing them into a set of smaller problems all of which must be solved.
- Here, you can have *and* nodes whose successors must *all* be achieved, and *or* nodes where *one* of the successors must be achieved (ie, they are alternatives).

Problem Reduction

- This decomposition or reduction, generates arcs that we call AND arcs.
- One AND arc may point to a number of successor nodes, all of which must be solved in order for the arc to point to a solution.
- As in OR graph, several arcs may emerge from a single node, indicating the variety of ways in which the original problem might be solved. That is why is called AND-OR graph

Problem Reduction

- This Allows Us To Represent Both Cases Where ALL Of A Set Of Sub-goals Must Be Satisfied To Achieve Some Goal, And Where There Are Alternative Sub-goals, Any Of Which Could Achieve The Goal.

Problem Reduction

- To find a way to get to London we need to search this tree to find a set of simple goals that we trivially know how to satisfy. Maybe ordering a taxi is a primitive goal - we could decompose it into ``pick up phone, dial number .." but this wouldn't be very helpful. Anyway, we need to apply our basic ideas of tree/graph search to searching AND-OR trees/graphs.

Problem Reduction

- The typical problem that is used to illustrate problem reduction search is the Tower of Hanoi problem because this problem has a very elegant solution using this method.
- 64 size ordered disks occupy one of 3 pegs and must be transferred to one of the other pegs. But, only one disk can be moved at a time; and a larger disk may never be placed on a smaller disk.

Problem Reduction

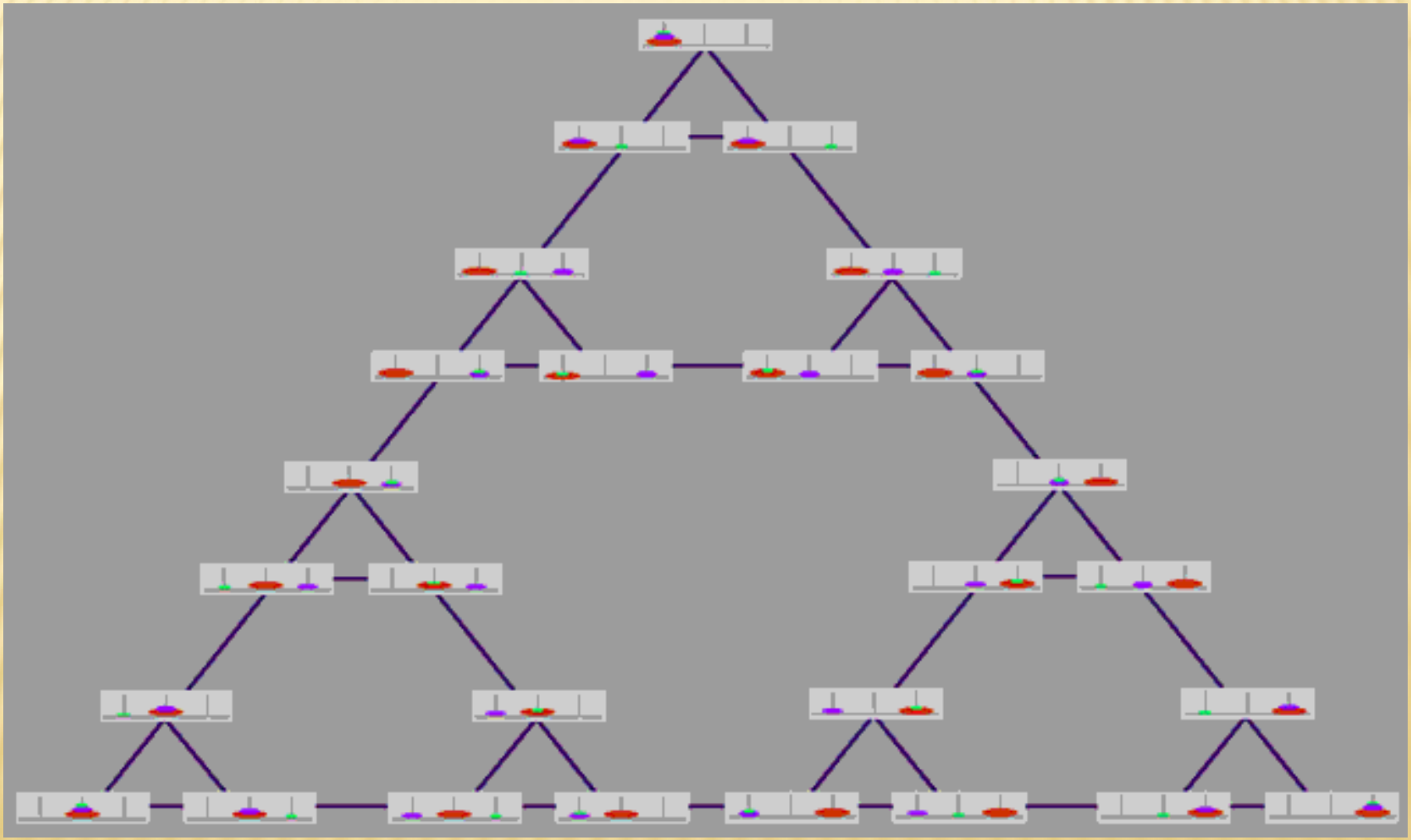
- The figure below shows the **state space** associated with a 3-disk Tower of Hanoi Problem. The problem involves moving from a state where the disks are stacked on one of the pegs and moving them so that they end up stacked on a different peg.
- In this case, we will consider the state at the top of the figure the *starting state*. In this case all three disks are on the left-most peg.

Problem Reduction

- And we will consider the state at the bottom right to be the *goal state*. In this state the three disks are now all stacked on the right-most peg.

Problem Reduction

State Space For The 3 Disk Tower Of Hanoi Problem



Problem Reduction

- Recall that in state space search the generators correspond to moves in the state space.
- Thus, the two states below the top state in the triangle of states corresponds to the movement of the smallest disk either to the rightmost peg or to the middle peg.
- The shortest solution to this problem corresponds to the path down the right side of the state space.

Problem Reduction

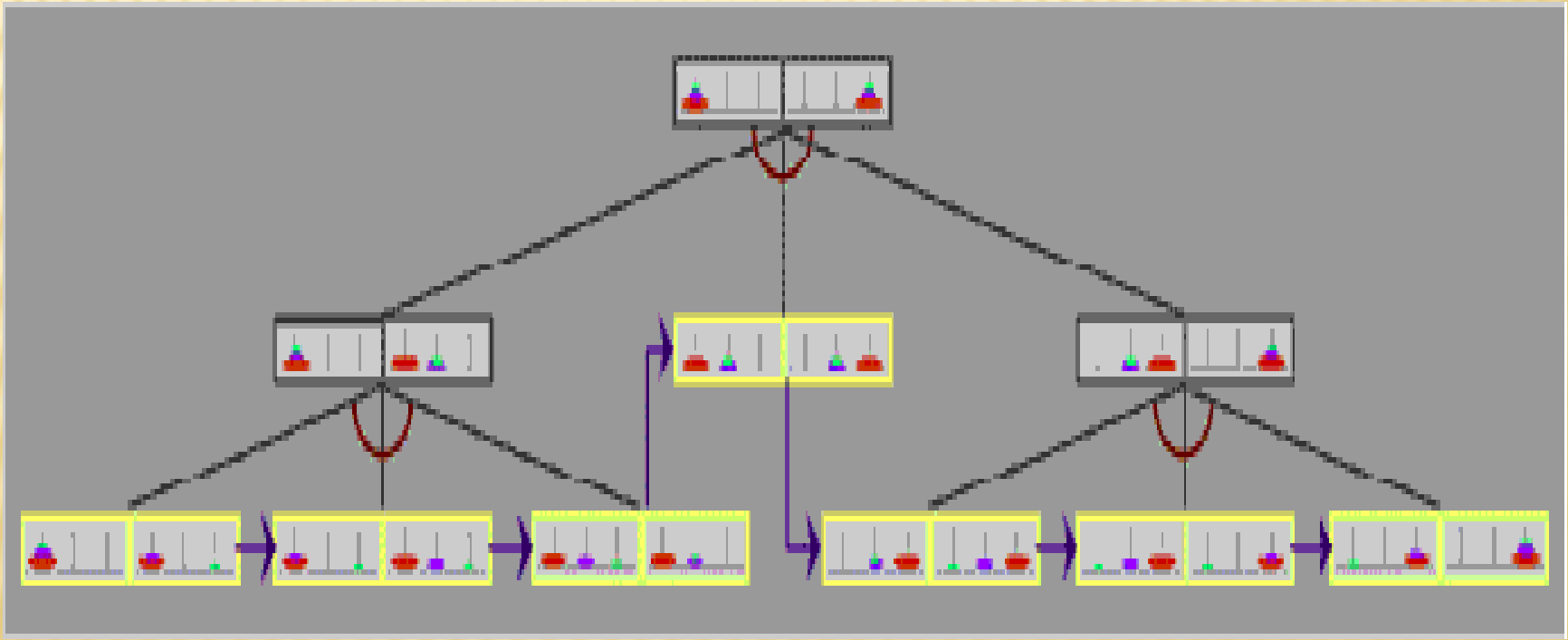
- In problem reduction search the problem space consists of an AND/OR graph of (partial) state pairs. These pairs are referred to as *(sub)problems*.
- The first element of the pair is the starting state of the (sub)problem and the second element of the pair is the goal state (sub)problem.

Problem Reduction

- The symmetry of the state space shown above may have led you to suspect that the Tower of Hanoi problem can be elegantly solved using the method of problem decomposition.
- The AND tree that solves the 3 disk problem is shown below

Problem Reduction

- AND Tree Showing the Problem Reduction Solution to the 3 Disk Tower of Hanoi Problem



Types of Problem Spaces

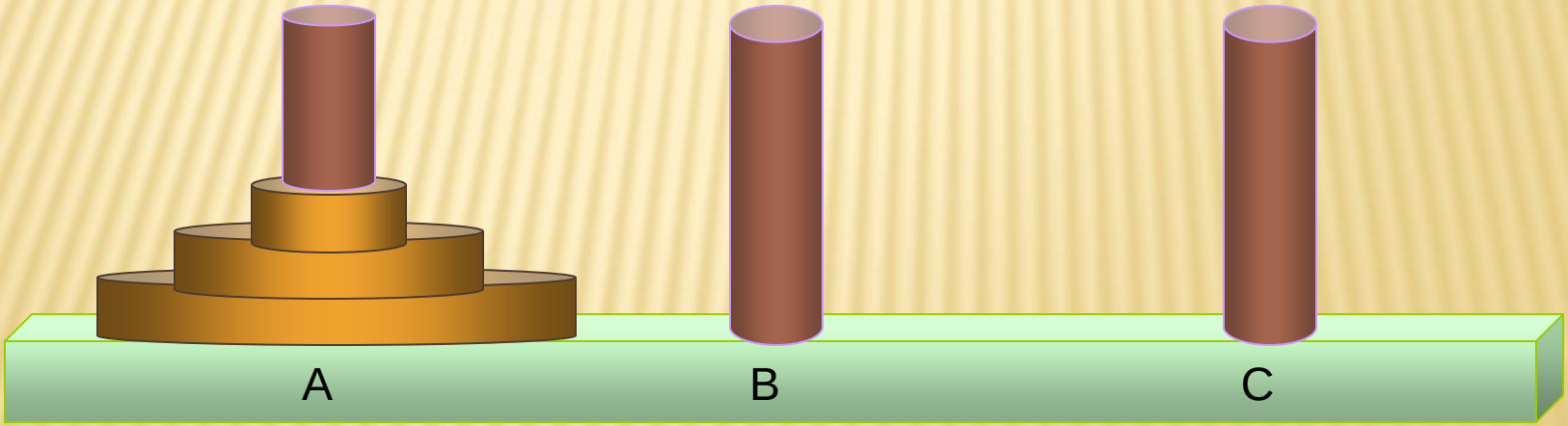
- There are several types of problem spaces:
  - State space
  - Problem Reduction Space
  - AND/OR Graphs

State Space

- The **states** represent situations of the problem.
- The **operators** represent actions in the world.
  - **forward search**: the root of the problem space represents the start state, and the search proceeds forward to a goal state.
  - **backward search** : the root of the problem space represents the goal state, and the search proceeds backward to the initial state.
- *For example:* in Rubik's Cube and the Sliding-Tile Puzzle, either a forward or backward search are possible.

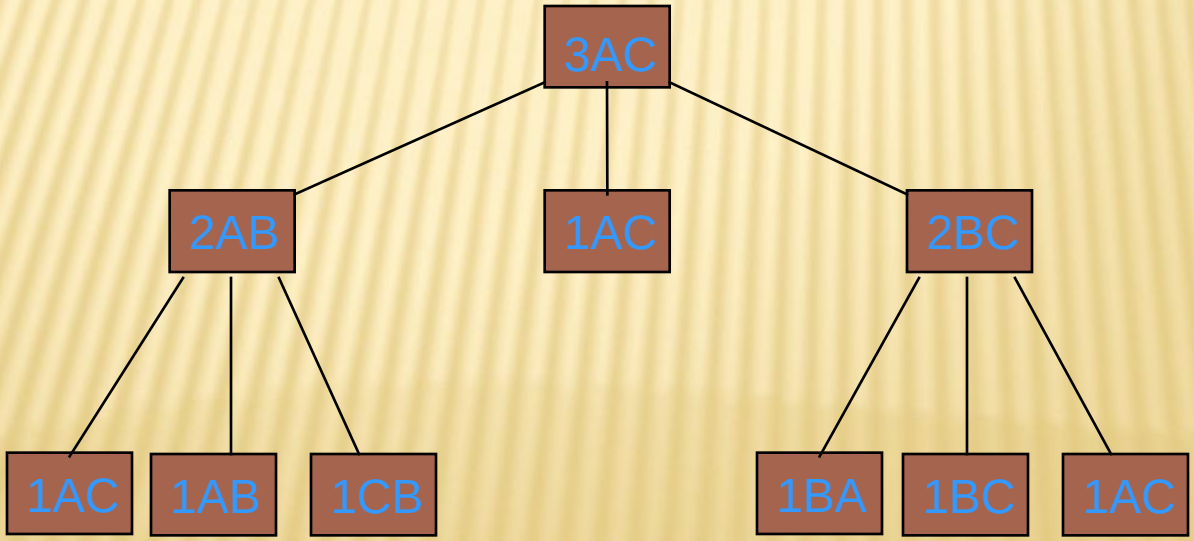
Problem Reduction Space

- In a problem reduction space, the nodes represent problems to be solved or goals to be achieved, and the edges represent the decomposition of the problem into subproblems.
- This is best illustrated by the example of the Towers of Hanoi problem.

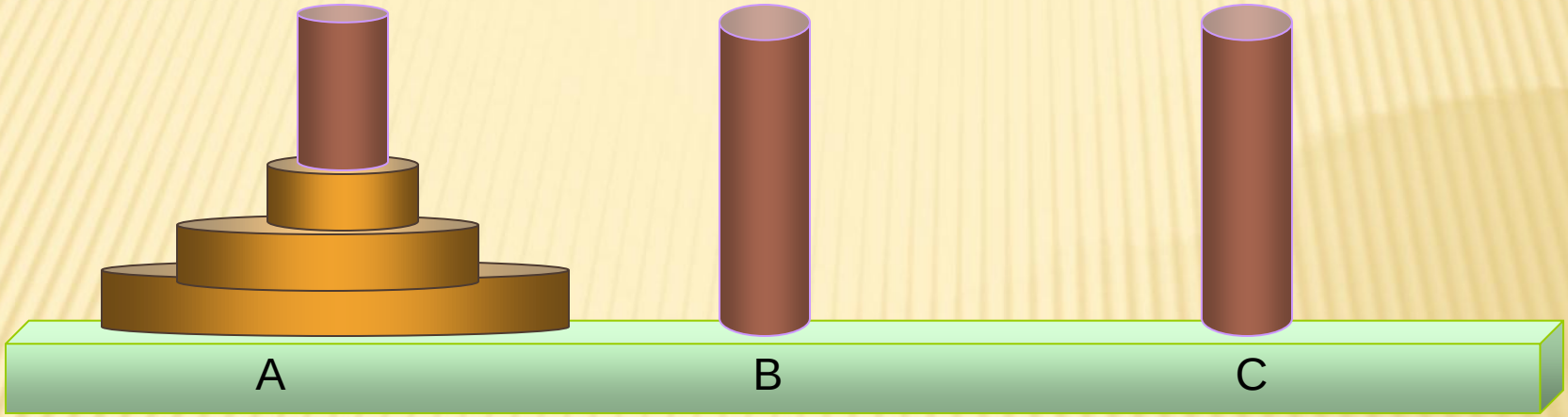


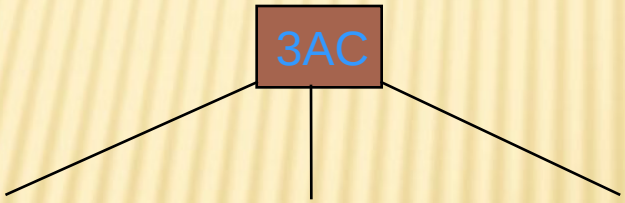
Problem Reduction Space

- The root node, labeled “3AC” represents the original problem of transferring all 3 disks from peg A to peg C.
- The goal can be decomposed into three subgoals: 2AB, 1AC, 2BC. In order to achieve the goal, all 3 subgoals must be achieved.

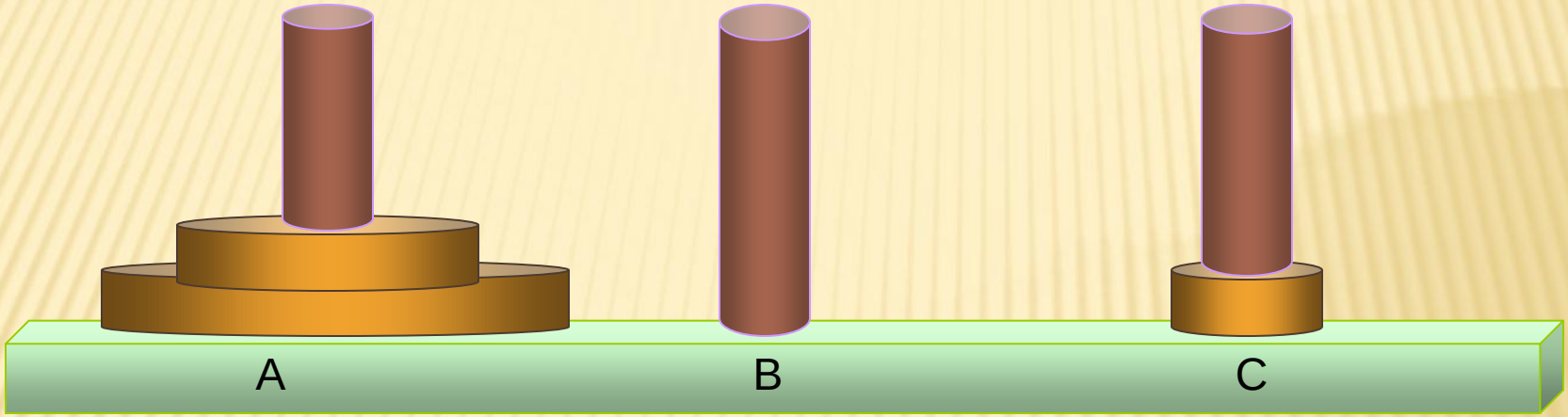


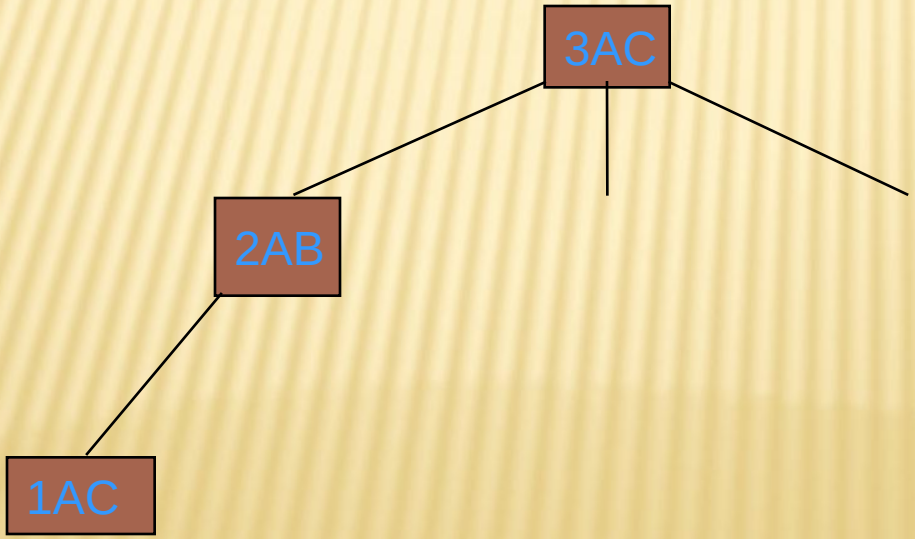
Problem Reduction Space



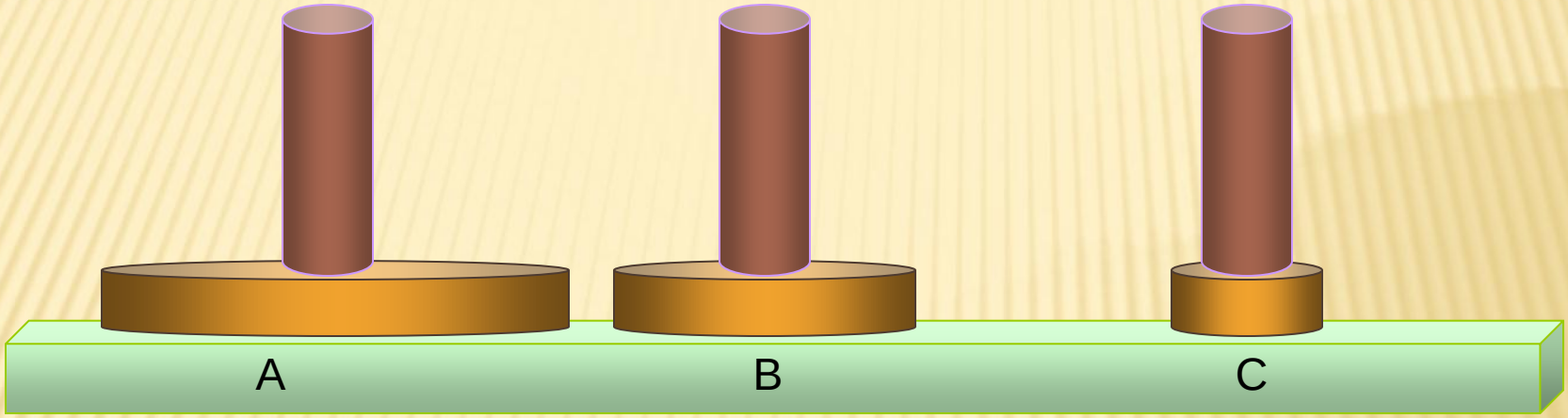


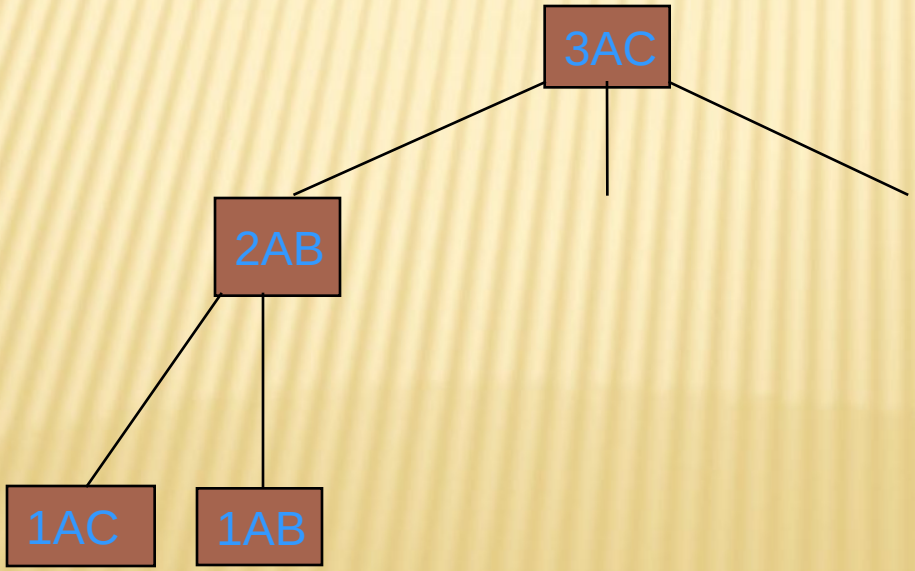
Problem Reduction Space



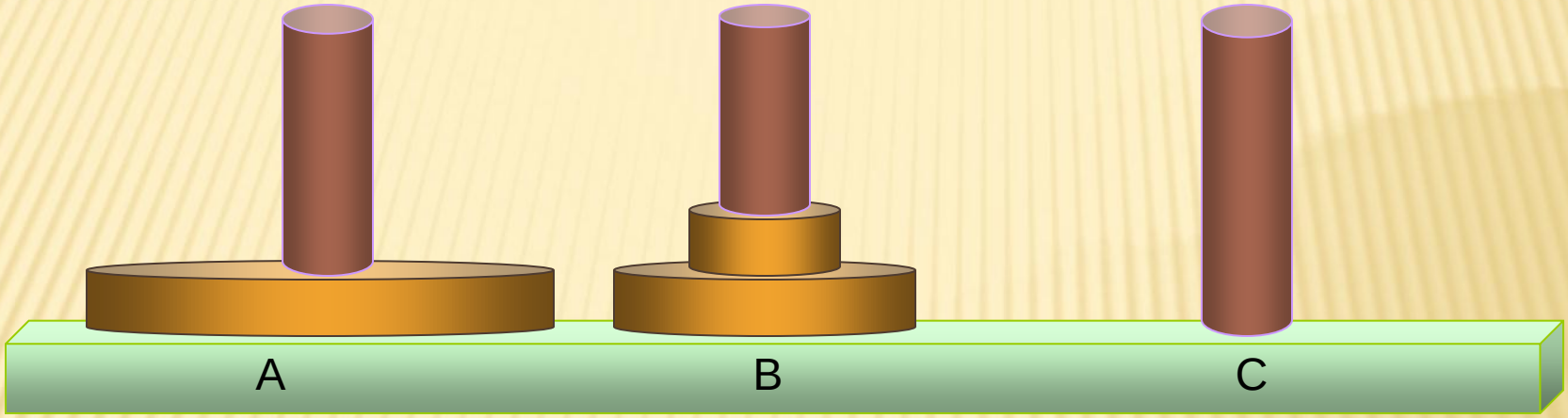


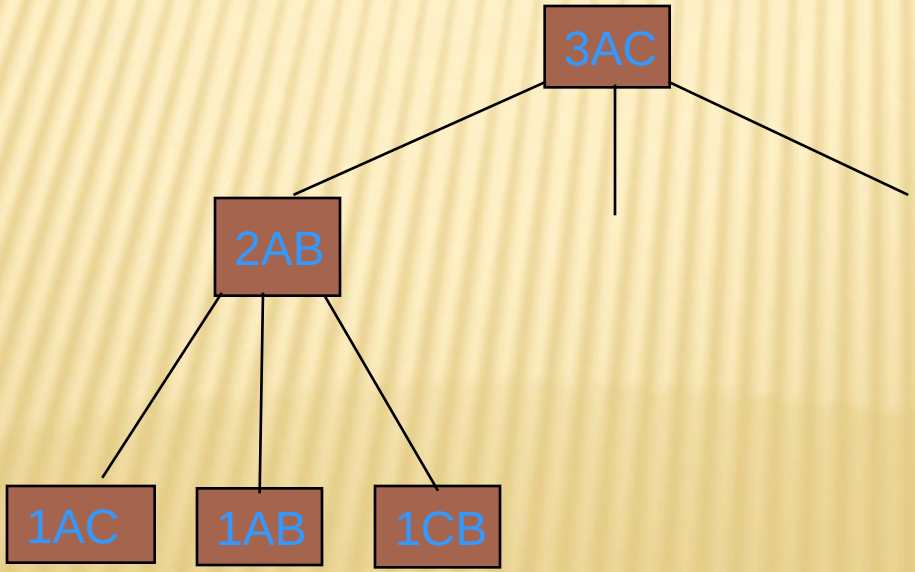
Problem Reduction Space



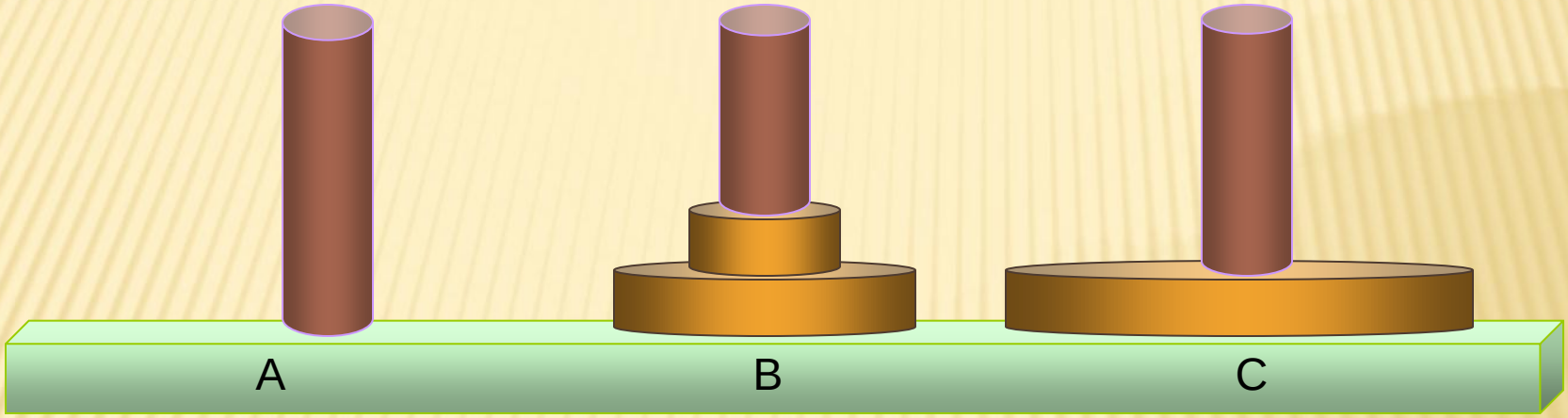


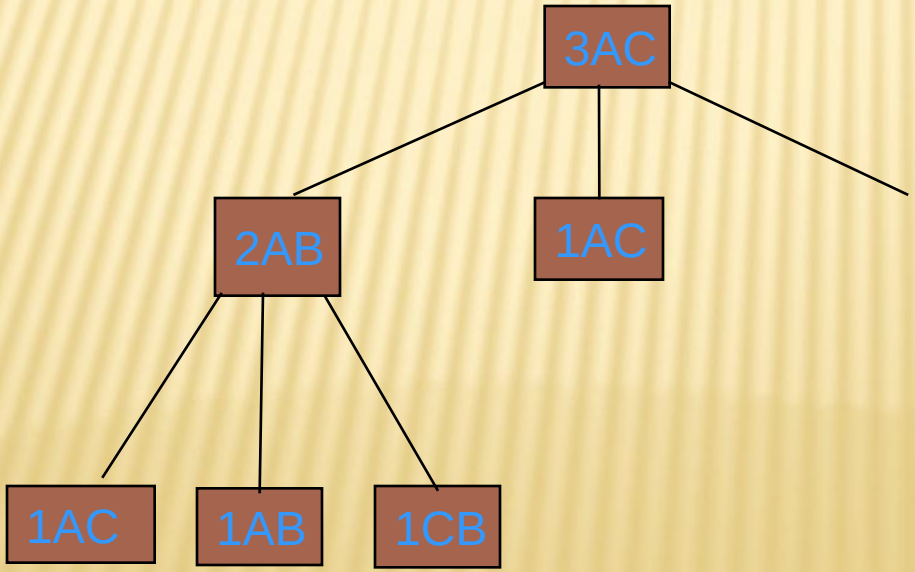
Problem Reduction Space



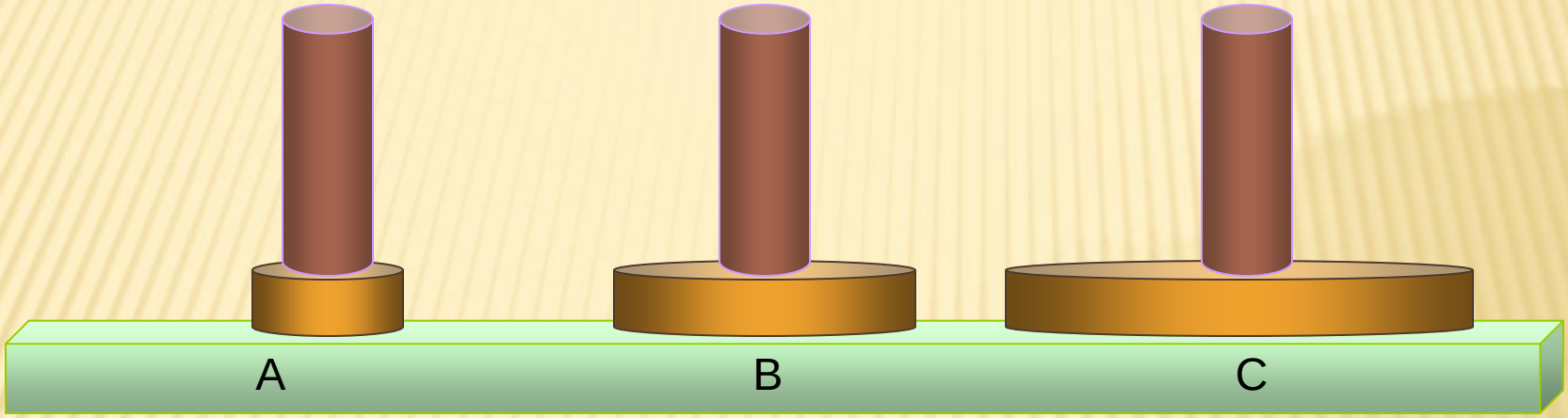


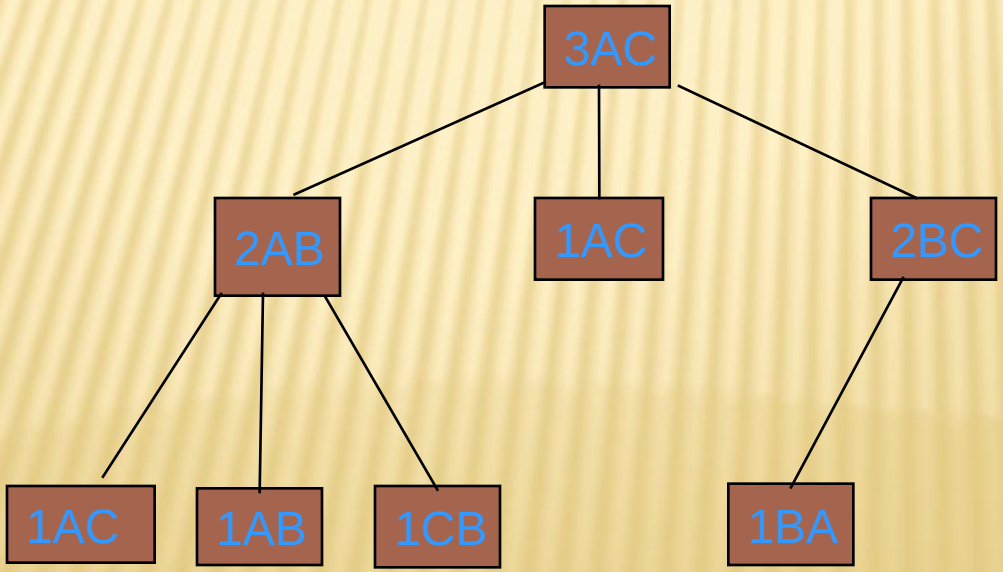
Problem Reduction Space



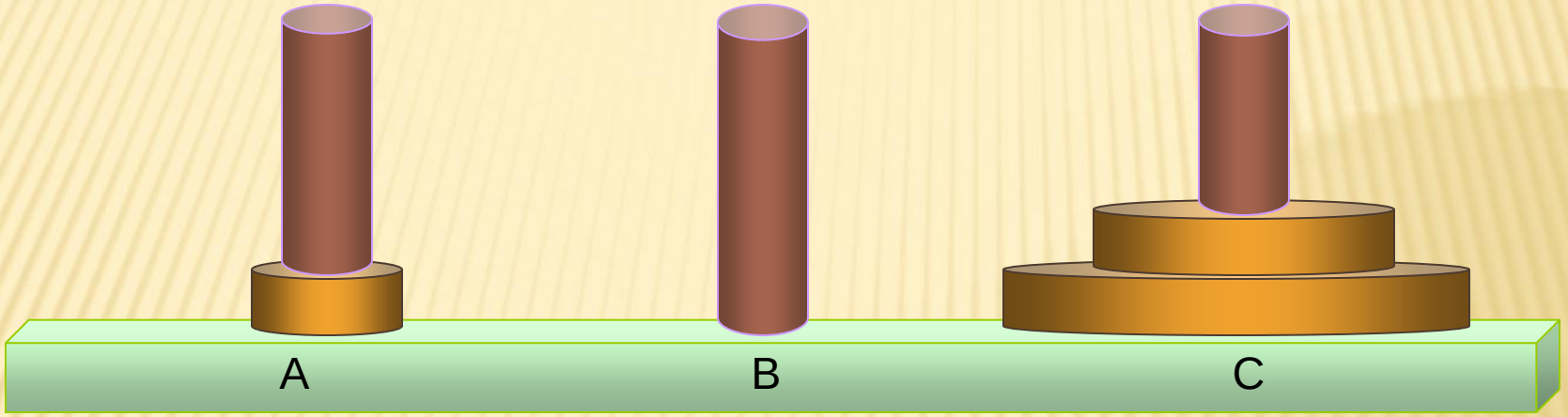


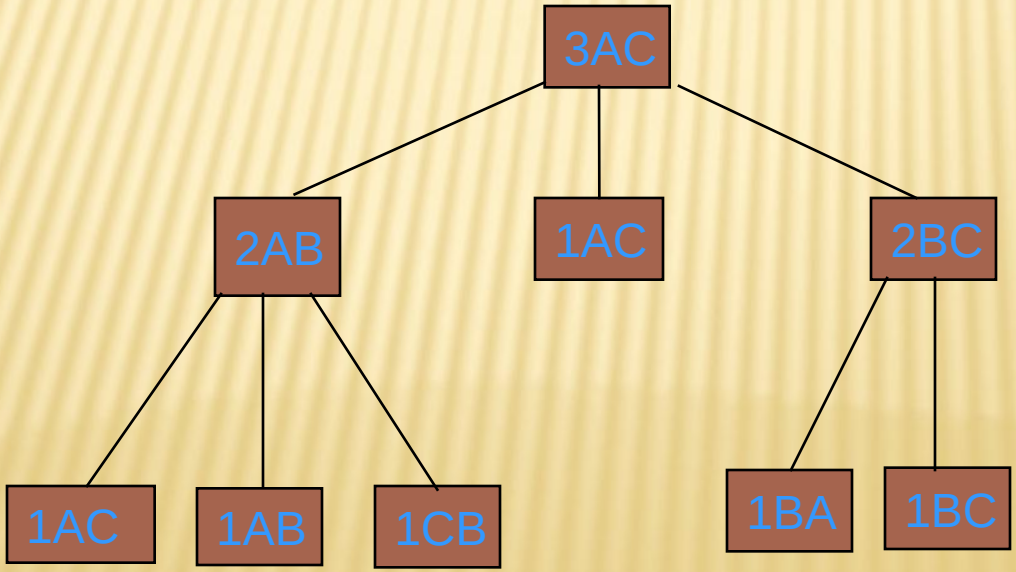
Problem Reduction Space



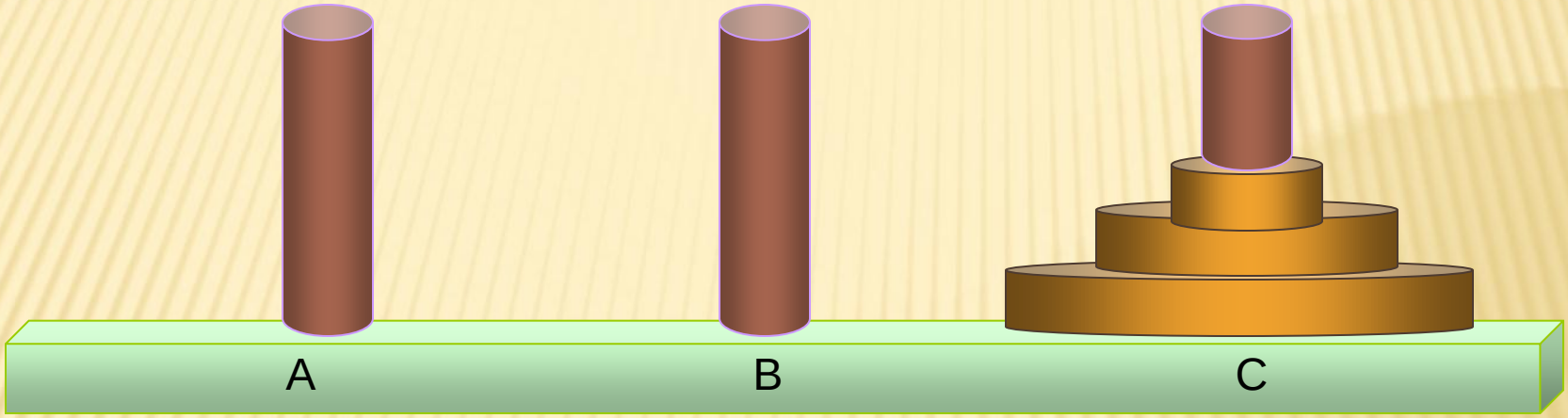


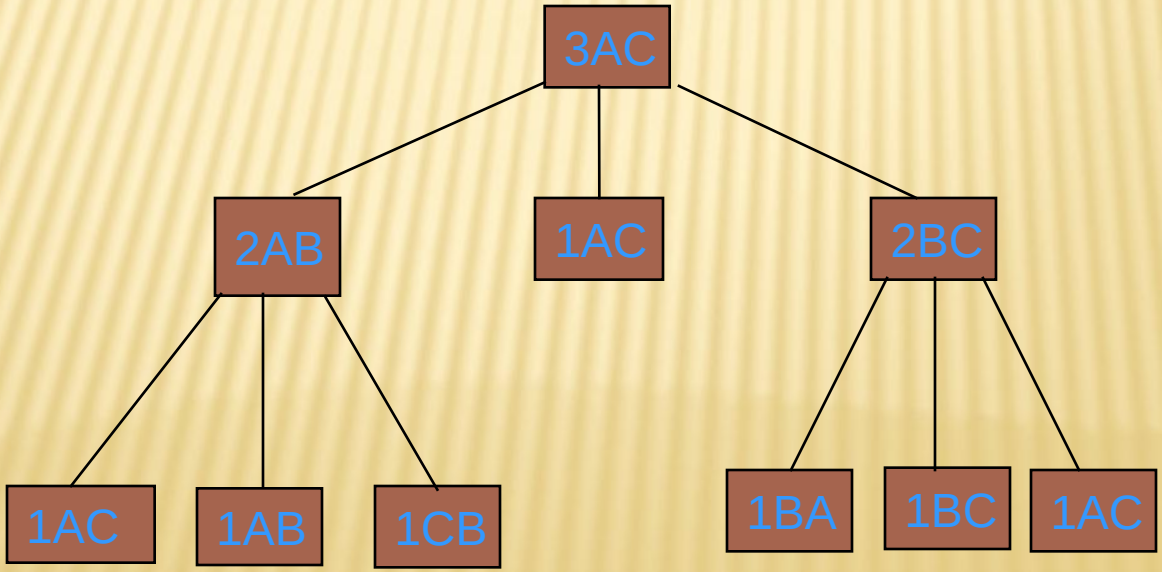
Problem Reduction Space





Problem Reduction Space





AND/OR GRAPHS

- An AND Graph Consists Entirely Of AND Nodes, And In Order To Solve A Problem Represented By It, You Need To Solve The Problems Represented By **All** Of His Children *(Hanoi Towers Example).*
- An Or Graph Consists Entirely Of Or Nodes, And In Order To Solve The Problem Represented By It, You Only Need To Solve The Problem Represented By **One** Of His Children *(Eight Puzzle Tree Example).*

AND/OR GRAPHS

- An AND/OR graph consists of both AND nodes and OR nodes.
- One source of AND/OR graphs is a problem where the effect of an action cannot be predicted in advanced, as in an interaction with the physical world.
- *Example*:
  - the counterfeit-coin problem.

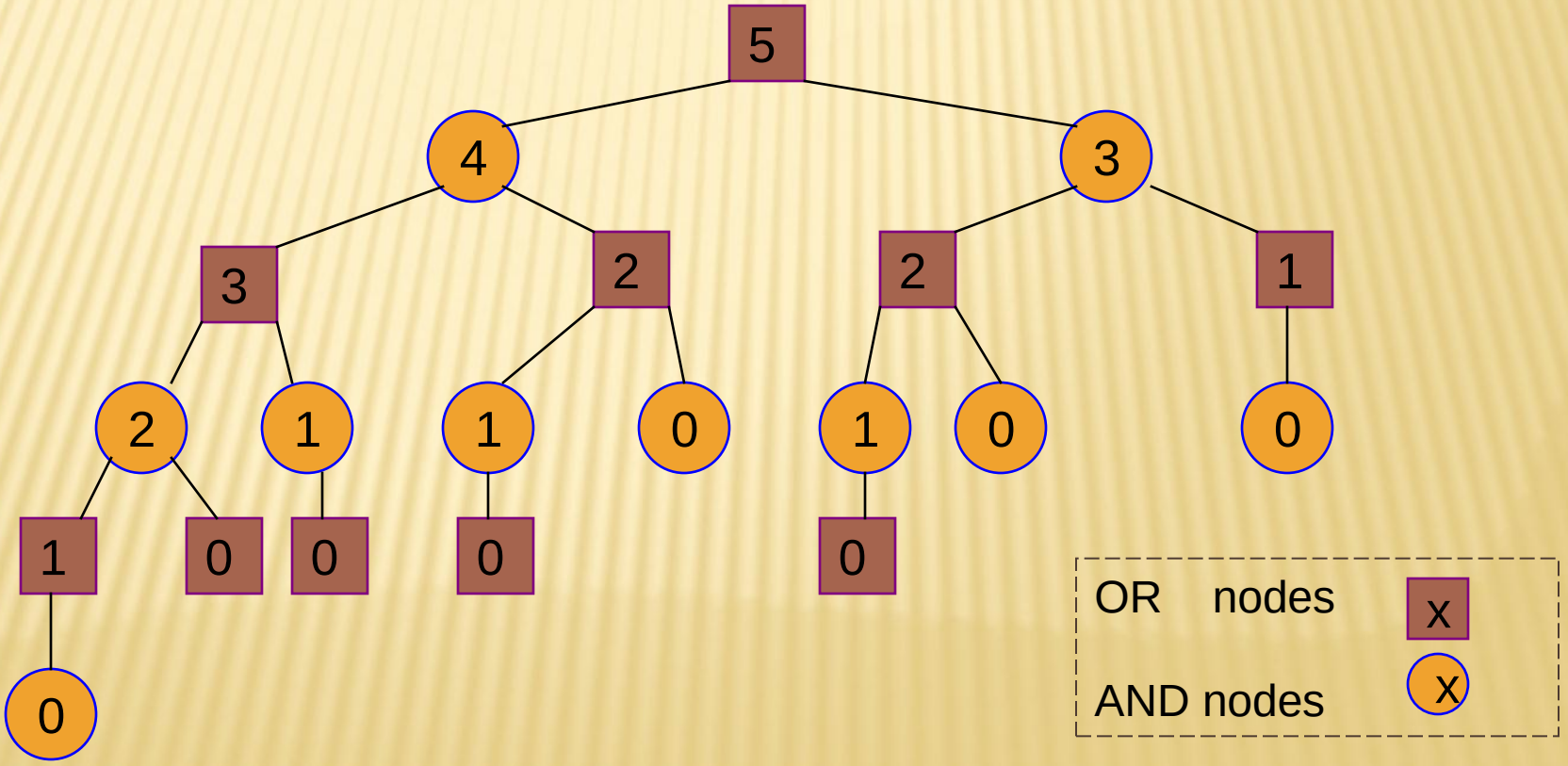
5-stone Nim Problem

- this game is played with two players and a pile of stones. Suppose we have 15 stones to start with. Each player removes either one or two stones from the pile and the player who removes the last stone wins the game.

- Initially let us assume that we start with 5 stones. Now, at this position the max player can move either one stone or two stones giving rise to either 4 or 3 stones left over. The min player can remove either 1 or 2 stones from 4 or from 3. Again the max player can remove 1 or 2 stones etc., and we stop expanding the tree until all the leaves have 0 that is no stone left. The box represents the max player and circle represents the min player. This is the max player and this is the min player.

Two-player Game Trees

- The most common source of AND/OR graphs is 2-player perfect-information games.
- *Example*: Game Tree for 5-Stone Nim:



MINIMAX THEOREM

- The Minimax theorem states that every two person zero sum game is a forced win for one player or a forced draw for either player, in principle these optimal MINIMAX strategies can be computed. So, if we have the pay offs at the leaves and we back this up finally at the root we will get a value. That value can be either 0 or positive or negative. So if the value is positive, if max plays judiciously max can force a win from this position. If the backed up pay off value at the root is negative no matter how well max plays if min plays well and extremely intelligently then min can force Max's loss. If the pay off is zero if both the players play at their best this game would end in a draw

Solution Subgraph For AND/OR Trees

- In general, a solution to an AND/OR graph is a subgraph with the following properties:
  - It contains the root node. For every OR node included in the solution subgraph, one child is included.
  - For every OR node included in the solution subgraph, all the children are included.
  - Every terminal node in the solution subgraph is a solved node.

SOLUTIONS

- The notion of a solution is different for the different problem types:
  - *For a path-finding problem*, an optimal solution is a solution of lowest cost.
  - *For a CSP(Constraint Satisfaction Problem)*, if there is a cost function associated with a state of the problem, an optimal solution would again be one of lowest cost.
  - *For a 2-player game*:
    - If the solution is simply a move to be made, an optimal solution would be the best possible move that can be made in a given situation.
    - If the solution is considered a complete strategy subgraph, then an optimal solution might be one that forces a win in the fewest number of moves in the worst case.

Combinatorial Explosion

- The number of different states of the problems above is enormous, and grows extremely fast as the problem size increases.
- *Examples* for the number of different possibilities:

Game	# of nodes
8-Puzzle	9!
15-Puzzle	16!
24-Puzzle	25!
Rubik's Cube- 2x2x2	3,265,920
Rubik's Cube- 3x3x3	4.32×10^{19}
N-city TSP	n!
Checkers	10^{20}
Chess	10^{40}
N-Queens	n!

Combinatorial Explosion

- The combinatorial explosion of the number of possible states as a function of problem size is a key characteristic that separates artificial intelligence search algorithms in other areas of computer science.
- Techniques that rely on storing all possibilities in memory, or even generating all possibilities, are out of the question except for the smallest of these problems. As a result, the problem-space graphs of AI problems are usually represented implicitly by specifying an initial state and a set of operators to generate new states.