CSD 3102 ARTIFICIAL INTELLIGENCE TECHNIQUES
MODULE IV

PLANNING

Topic: Goal Stack Planning

MODULE IV PLANNING 9

Planning Problem – Simple Planning agent –Blocks world - **Goal Stack Planning**-Means Ends Analysis- Planning as a Statespace Search - Partial Order Planning-Planning Graphs-Hierarchical Planning - Non- linear Planning -Conditional Planning-Reactive Planning - Knowledge based Planning-Using Temporal Logic – Execution Monitoring and Re-planning- Continuous Planning-Multi-agent Planning-Job shop Scheduling Problem.

Goal Stack Planning

- Goal Stack Planning is one of the earliest methods in artificial intelligence in which we work **backwards from the goal state to the initial state.**
- We start at the goal state and we try fulfilling the preconditions required to achieve the initial state.
- These preconditions in turn have their own set of preconditions, which are required to be satisfied first.
- We keep solving these “goals” and “sub-goals” until we finally arrive at the Initial State.
- **We make use of a stack to hold these goals that need to be fulfilled as well the actions that we need to perform for the same**.
- At the end of this algorithm we are left with an empty stack and a set of actions which helps us navigate from the Initial State to the Goal State.

Representing the configurations as a list of “predicates”

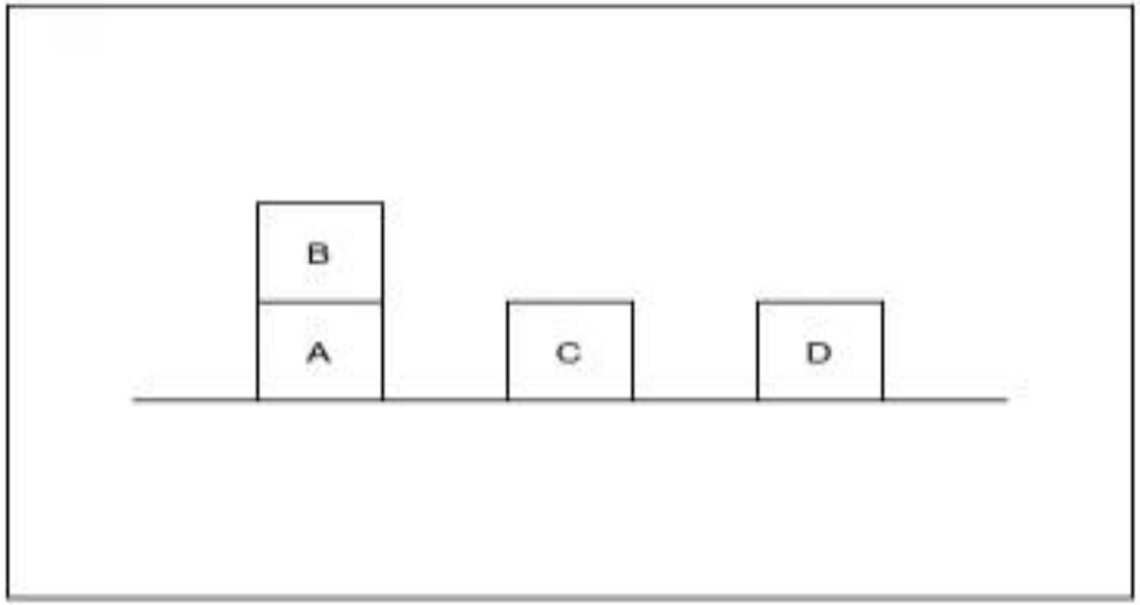
- Predicates can be thought of as a statement which helps us convey the information about a configuration in Blocks World.
- Given below are the list of predicates as well as their intended meaning
- ON(A,B) : Block A is on B
- ONTABLE(A) : A is on table
- CLEAR(A) : Nothing is on top of A
- HOLDING(A) : Arm is holding A.
- ARMEMPTY : Arm is holding nothing

Goal Stack Planning- BLOCK WORLD PROBLEM:

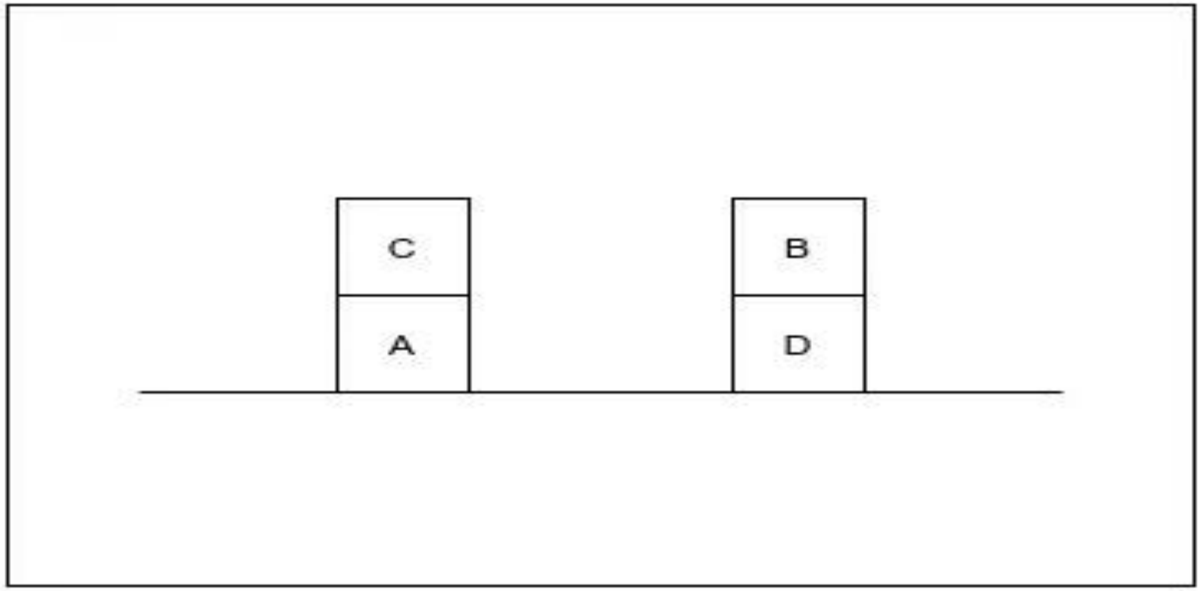
- Using these predicates, we represent the Initial State and the Goal State in our example like this:

Initial State — ON(B,A) ∧ ONTABLE(A) ∧ ONTABLE(C) ∧ ONTABLE(D) ∧ CLEAR(B) ∧ CLEAR(C) ∧ CLEAR(D) ∧ ARMEMPTY

Goal State — ON(C,A) ∧ ON(B,D) ∧ ONTABLE(A) ∧ ONTABLE(D) ∧ CLEAR(B) ∧ CLEAR(C) ∧ ARMEMPTY



Initial State



Goal State

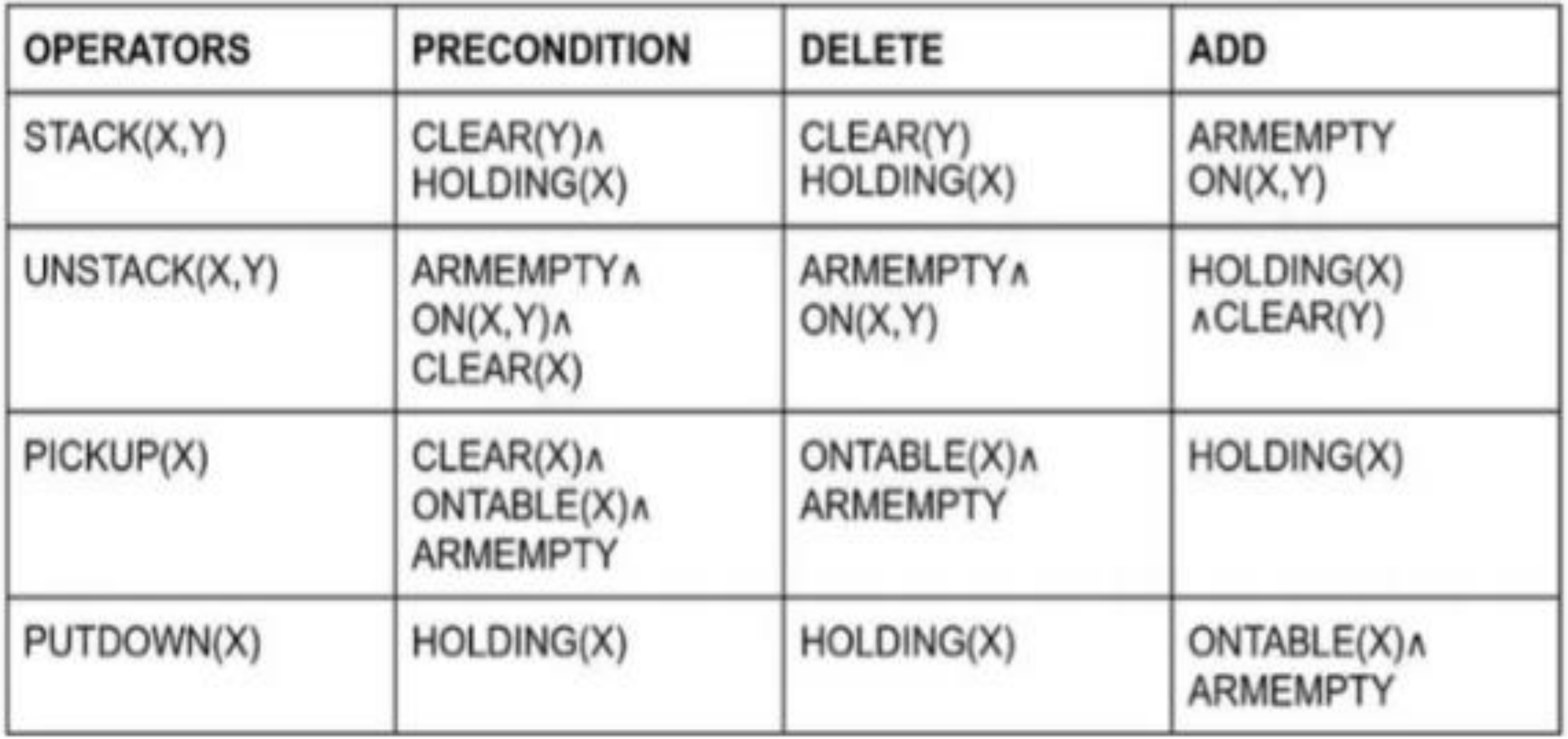
“Operations” performed by the robot arm

The Robot Arm can perform 4 operations:

- STACK(X,Y) : Stacking Block X on Block Y
- UNSTACK(X,Y) : Picking up Block X which is on top of Block Y
- PICKUP(X) : Picking up Block X which is on top of the table
- PUTDOWN(X) : Put Block X on the table

- All the four operations have certain preconditions which need to be satisfied to perform the same. These preconditions are represented in the form of predicates.
- The effect of these operations is represented using two lists **ADD** and **DELETE**. DELETE List contains the predicates which will cease to be true once the operation is performed. ADD List on the other hand contains the predicates which will become true once the operation is performed.

- The Precondition, Add and Delete List for each operation is rather intuitive and have been listed below.



OPERATORS	PRECONDITION	DELETE	ADD

STACK(X,Y)	CLEAR(Y)∧ HOLDING(X)	CLEAR(Y)
		HOLDING(X)
			ARMEMPTY
			ON(X,Y)

UNSTACK(X,Y)	ARMEMPTY∧ ON(X,Y)∧ CLEAR(X)	ARMEMPTY∧ ON(X,Y)	HOLDING(X) ∧CLEAR(Y)
PICKUP(X)	CLEAR(X)∧ ONTABLE(X)∧ ARMEMPTY	ONTABLE(X)∧ ARMEMPTY	HOLDING(X)
PUTDOWN(X)	HOLDING(X)	HOLDING(X)	ONTABLE(X)∧ ARMEMPTY

Operations performed by the Robot Arm

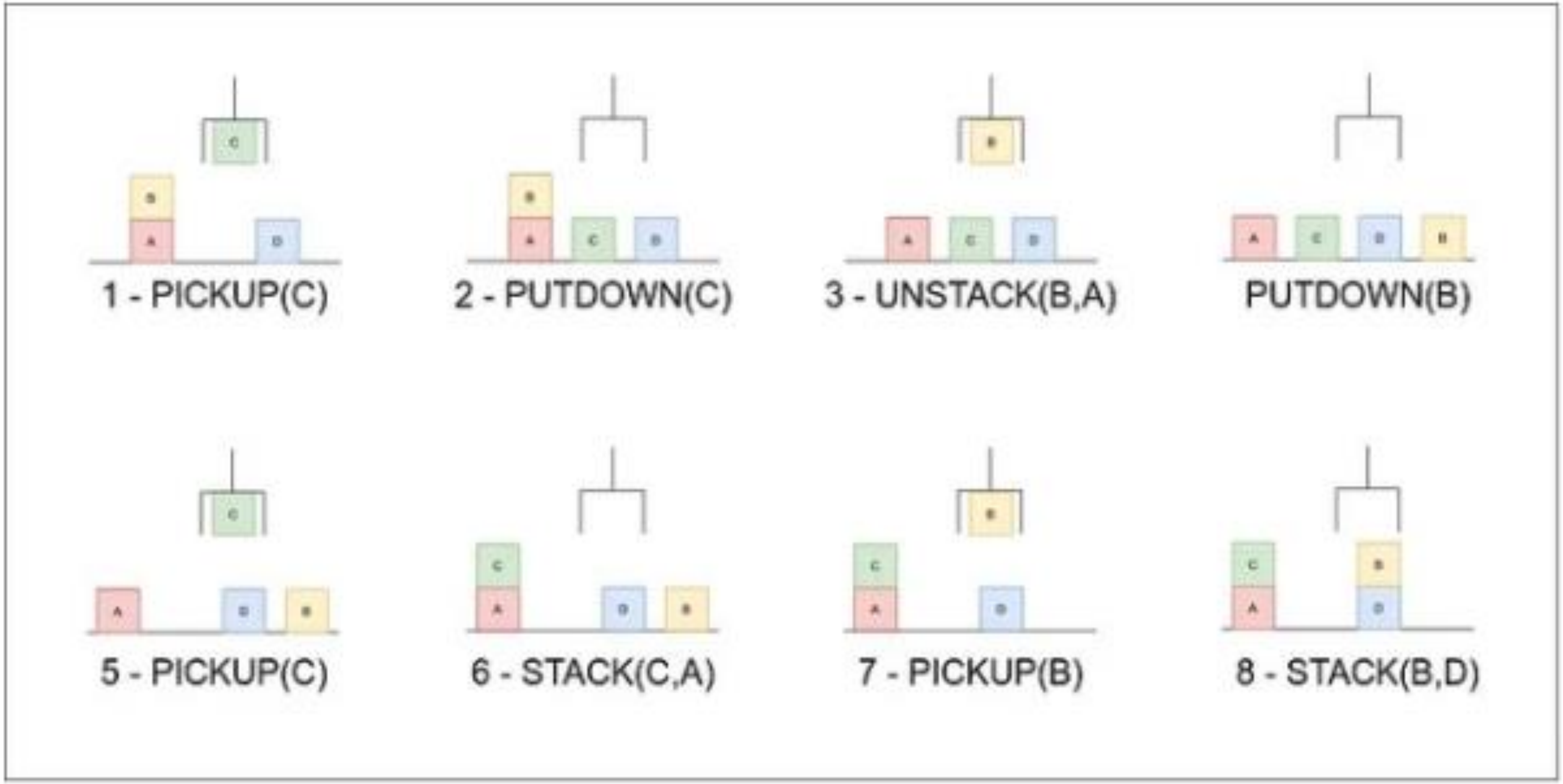
- For example, to perform the **STACK(X,Y)** operation i.e. to Stack Block X on top of Block Y, No other block should be on top of Y **(CLEAR(Y))** and the Robot Arm should be holding the Block X (**HOLDING(X)**).

- Once the operation is performed, these predicates will cease to be true, thus they are included in **DELETE List** as well. (Note : It is not necessary for the Precondition and DELETE List to be the exact same).

- On the other hand, once the operation is performed, The robot arm will be free (**ARMEMPTY**) and the block X will be on top of Y (**ON(X,Y)**).

- The other 3 Operators follow similar logic, and this part is the cornerstone of Goal Stack Planning.

Solution is



1 - PICKUP(C)

2 - PUTDOWN(C)

3 - UNSTACK(B,A)

PUTDOWN(B)

5 - PICKUP(C)

6 - STACK(C,A)

7 - PICKUP(B)

8 - STACK(B,D)