

# Planning

---

# Introduction

---

Planning is the task of coming up with a sequence of actions that will achieve the goal.

A plan is a sequence of actions and each action has its own set of preconditions to be satisfied before performing the action and also some effects which can be positive or negative.

Goal : To attend a class

1. Start from home.
2. Go to the college.
3. Attend the class.

# Introduction

---

The *planning* is about the decision making performed by intelligent agents like robots, humans, or computer programs when trying to achieve some goal.

It involves choosing a sequence of actions that will (with a high likelihood) transform the initial state to the goal state.

The world is usually regarded as being made up of atomic facts (state variables), and action(s) make some facts real and some facts false.

# The Language of Planning Problems - STRIPS

---

STanford Research Institute Problem Solver(STRIPS)

A restricted language for planning that describes actions and objects in a system.

# STRIPS

---

This abstracts away many important details!

Restricted language  $\rightarrow$  efficient algorithm

- Precondition: conjunction of positive literals

$$p \wedge Q$$

- Effect: conjunction of literals

$$P \wedge \neg Q$$

A complete set of STRIPS operators can be translated into a set of successor-state axioms

# Specification of actions

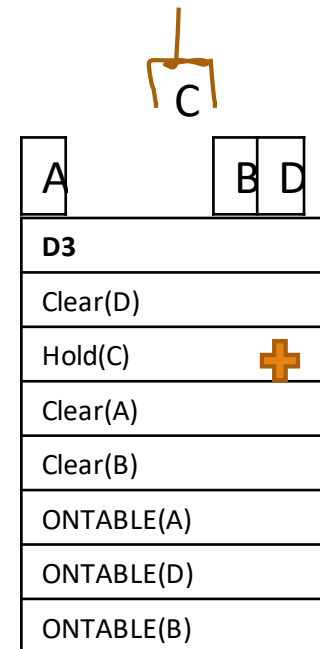
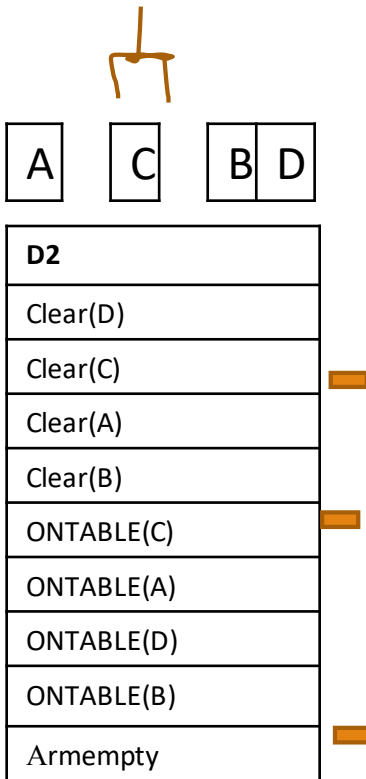
---

- PRECONDITION: list of predicates that **must be true** for an operator to be applied.

- Effect {
- ADD: list of new predicates that an operator causes to **become true**.
  - DELETE: list of old predicates that an operator causes to **become false**.
  - Predicates not in ADD nor DELETE are **unaffected**.

# For Action Pickup(C)

Precondition:  $\text{armempty} \wedge \text{clear}(C)$



# Example:

---

Initial State: *have\_Not(bread)*

Goal State: *Have(bread)*

- Action: *Goto(Shop), Buy(bread), Comeback(home)*
- Precondition: *At(shop), Sells(shop,bread)*
- Effect: *Have(bread)*

*P: At(shop), Sells(shop,bread)*

*Buy(bread)*

*Have( bread)*



# Example of Planning Problems

## Air Cargo Transport:

---

- Action( Fly( p, from, to ))
  - Precond:  $\text{At}(p, \text{from}) \wedge \text{Plane}(p) \wedge \text{Airport}(\text{from}) \wedge \text{Airport}(\text{to})$
  - Effect:  $\neg \text{At}(p, \text{from}) \wedge \text{At}(p, \text{to})$
- This is also known as an **action schema**

# Air Cargo Transport Cont..

---

$\text{Init}(\text{At}(C_1, \text{CLE}) \wedge \text{At}(C_2, \text{LAS}) \wedge \text{At}(P_1, \text{CLE}) \wedge \text{At}(P_2, \text{LAS}) \wedge \text{Cargo}(C_1) \wedge \text{Cargo}(C_2) \wedge \text{Plane}(P_1) \wedge \text{Plane}(P_2) \wedge \text{Airport}(\text{CLE}) \wedge \text{Airport}(\text{LAS}))$

$\text{Goal}(\text{At}(C_1, \text{LAS}) \wedge \text{At}(C_2, \text{CLE}))$

# Air Cargo Transport Cont..

---

*Action( Load( $c, p, a$ ),*

*Precond:  $At(c, a) \wedge At(p, a) \wedge Cargo(c) \wedge Plane(p) \wedge Airport(a)$*

*Effect:  $\neg At(c, a) \wedge In(c, p)$* )

*Action( Unload( $c, p, a$ ),*

*Precond:  $In(c, p) \wedge At(p, a) \wedge Cargo(c) \wedge Plane(p) \wedge Airport(a)$*

*Effect:  $At(c, a) \wedge \neg In(c, p)$* )

*Action( Fly( $p, from, to$ ),*

*Precond:  $At(p, from) \wedge Plane(p) \wedge Airport(from) \wedge Airport(to)$*

*Effect:  $\neg At(p, from) \wedge At(p, to)$* )

# Plan for Air Cargo Transport

---

[ Load( $C_1$ ,  $P_1$ , CLE),  
Fly( $P_1$ , CLE, LAS),  
Unload(  $C_1$ ,  $P_1$ , LAS),  
Load( $C_2$ ,  $P_2$ , LAS),  
Fly( $P_2$ , LAS, CLE),  
Unload(  $C_2$ ,  $P_2$ , CLE)]

# Partially Ordered Plans (POP)

---

Partial order planning find the actions corresponding to the current state which can be taken independently.

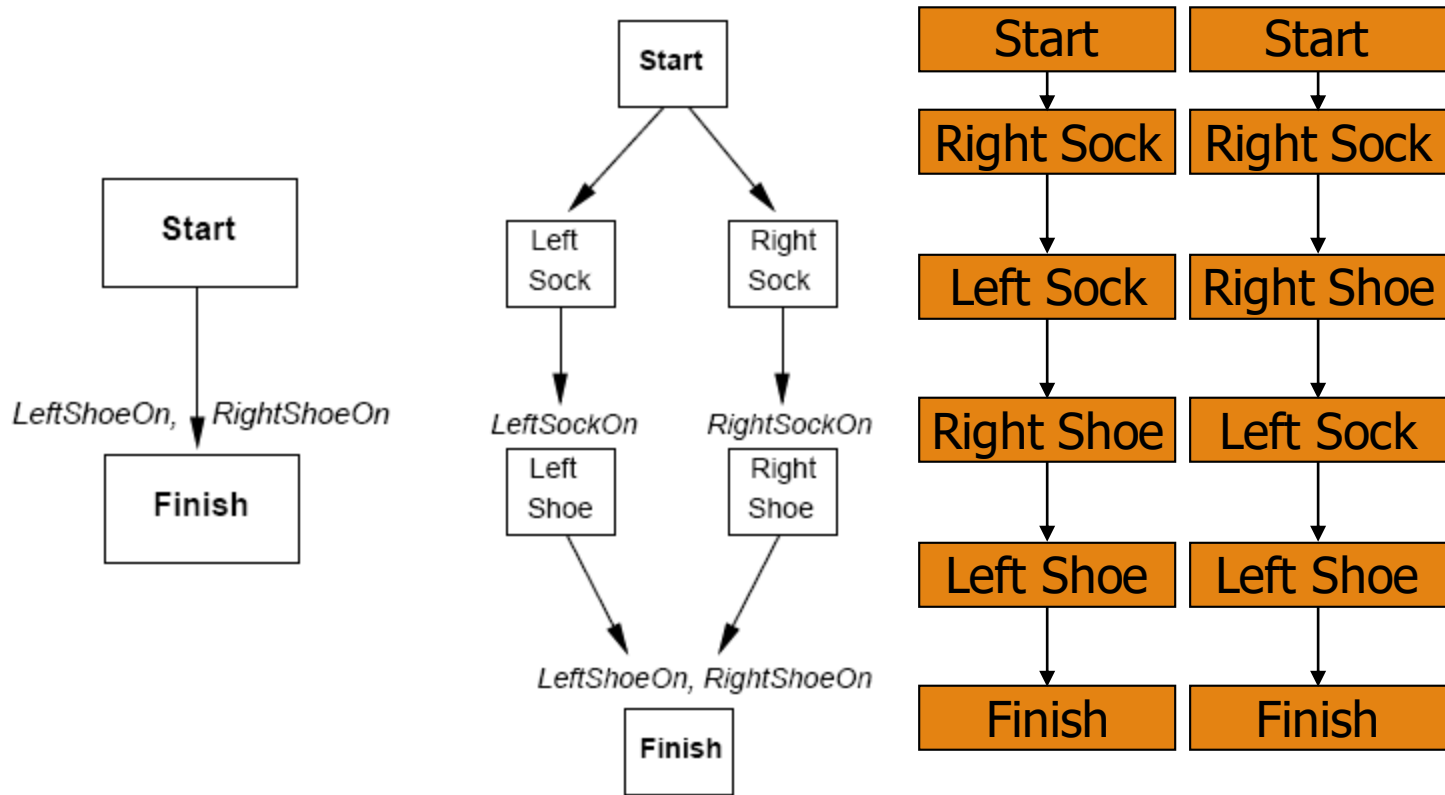
The ordering is not concerned in partial order planning .

E.g. if X and Y are the independent actions , precondition for both X (p1) and Y(p2) can be same, none or both p1 and p2 exist.

Plan of goal is : X:Y or Y:X

If any such actions exist , then the agent decompose the goal into sub goals which later on combined for the completion of a task.

# Partially Ordered Plans



# Goal Stack Planning

---

It is an important algorithm in Planning. STRIPS Language is used to complete the specified task.

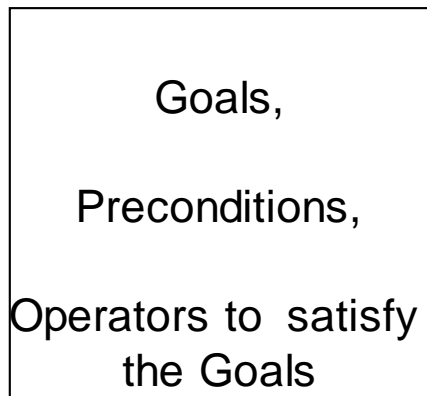
In the algorithm, a **Stack** is maintained with actions.

**A database** is used to hold the current state and actions. A sequence of actions is gathered as a separated list.

# Goal Stack Planning

---

Stack



+

Database





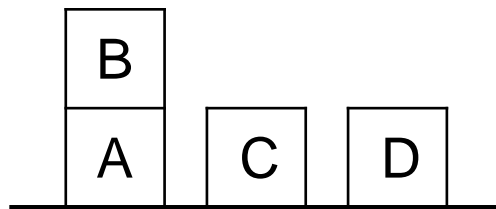
# Algorithm

---

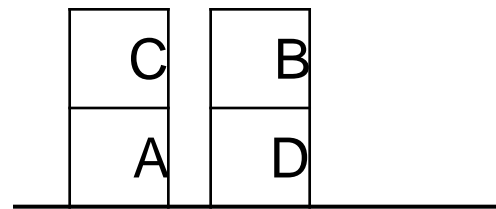
1. Start by pushing the original goal on the stack.
2. If stack top is a compound goal, then push its unsatisfied subgoals on the stack.
3. If stack top is a single unsatisfied goal then, replace it by an action and push the action's precondition on the stack to satisfy the condition.
4. If stack top is an action, pop it from the stack, execute it and change the knowledge base by the effects of the action.
5. If stack top is a satisfied goal, pop it from the stack.

# Plan for The Blocks World

---



start



goal

Planning = generating a **sequence of actions** to achieve the goal from the start

# Actions can be taken place in Block World Problem

---

- **pickup (X)**: pick up block X from its current location and hold it
- **putdown (X)**: place block X on the table.
- **stack (X, Y)**: place block X on top of block Y
- **unstack (X, Y)**: remove block X from the top of block Y and hold it
- All assume that the robot arm can precisely reach the block.

# Conditions and Results

---

- ON(X, Y)
- ONTABLE(X)
- CLEAR(X)
- HOLDING(X)
- ARMEMPTY

# Action Schema for Block World Problem

---

STACK(x, y):

P: CLEAR(y)  $\wedge$  HOLDING(x)

D: CLEAR(y)  $\wedge$  HOLDING(x)

A: ARMEMPTY  $\wedge$  ON(x, y)

UNSTACK(x, y):

P: ON(x, y)  $\wedge$  CLEAR(x)  $\wedge$  ARMEMPTY

D: ON(x, y)  $\wedge$  ARMEMPTY

A: HOLDING(x)  $\wedge$  CLEAR(y)

# Action Schema for Block World Problem

---

PICKUP(x):

P:  $\text{CLEAR}(x) \wedge \text{ONTABLE}(x) \wedge \text{ARMEMPTY} \quad ??$

D:  $\text{ONTABLE}(x) \wedge \text{ARMEMPTY}$

A:  $\text{HOLDING}(x)$

PUTDOWN(x):

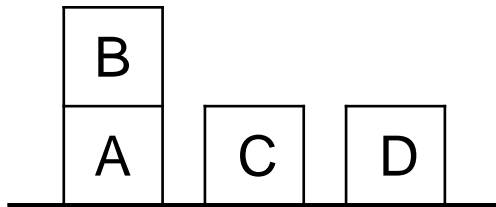
P:  $\text{HOLDING}(x)$

D:  $\text{HOLDING}(x)$

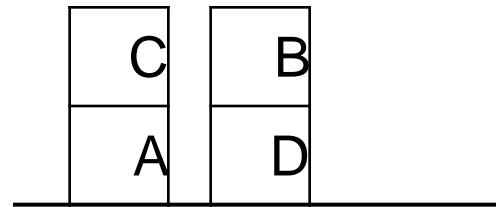
A:  $\text{ONTABLE}(x) \wedge \text{ARMEMPTY}$

# The Blocks World

---



**start:**  $\text{ON}(\text{B}, \text{A}) \wedge$   
 $\text{ONTABLE}(\text{A}) \wedge$   
 $\text{ONTABLE}(\text{C}) \wedge$   
 $\text{ONTABLE}(\text{D}) \wedge$   
 $\text{ARMEMPTY}$



**goal:**  $\text{ON}(\text{C}, \text{A}) \wedge$   
 $\text{ON}(\text{B}, \text{D}) \wedge$   
 $\text{ONTABLE}(\text{A}) \wedge$   
 $\text{ONTABLE}(\text{D}) \wedge$

# Goal Stack Planning

---

## Goal

$\text{ON}(\text{C}, \text{A}) \wedge \text{ON}(\text{B}, \text{D}) \wedge \text{ONTABLE}(\text{A}) \wedge \text{ONTABLE}(\text{D})$

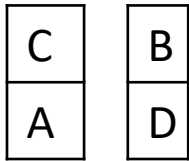
## Database

$\text{ON}(\text{B}, \text{A}) \quad \text{ONTABLE}(\text{C}) \quad \text{ONTABLE}(\text{D}) \quad \text{ARMEMPTY}$



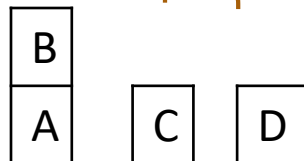
# Using Goal Stack Planning

Goal State



Initial Stack( $S_0$ )
ON(C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

Database



Initial Database ( $D_0$ )
ON(B, A)
ONTABLE(D)
ONTABLE(A)
ONTABLE(C)
ArmEmpty

Plan

Empty

# Using Goal Stack Planning

Goal State

C	B
A	D

Initial Stack (S <sub>0</sub> )
ON(C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

S1
P: Clear(A)
P: Hold(C)
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

Database

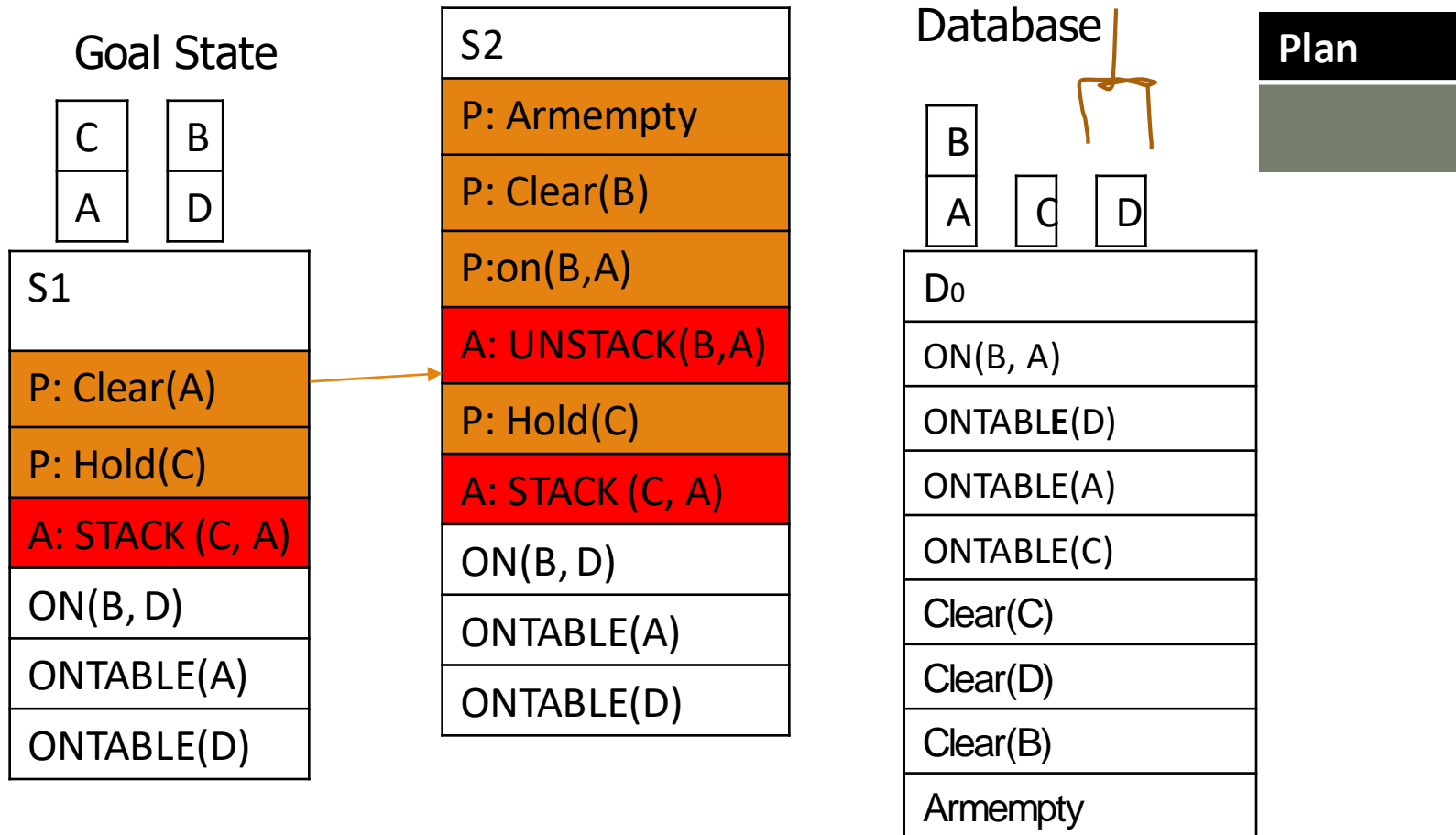
B		
A	C	D

D <sub>0</sub>
ON(B, A)
ONTABLE(D)
ONTABLE(A)
ONTABLE(C)
Clear(C)
Clear(D)
Clear(C)
Armempty

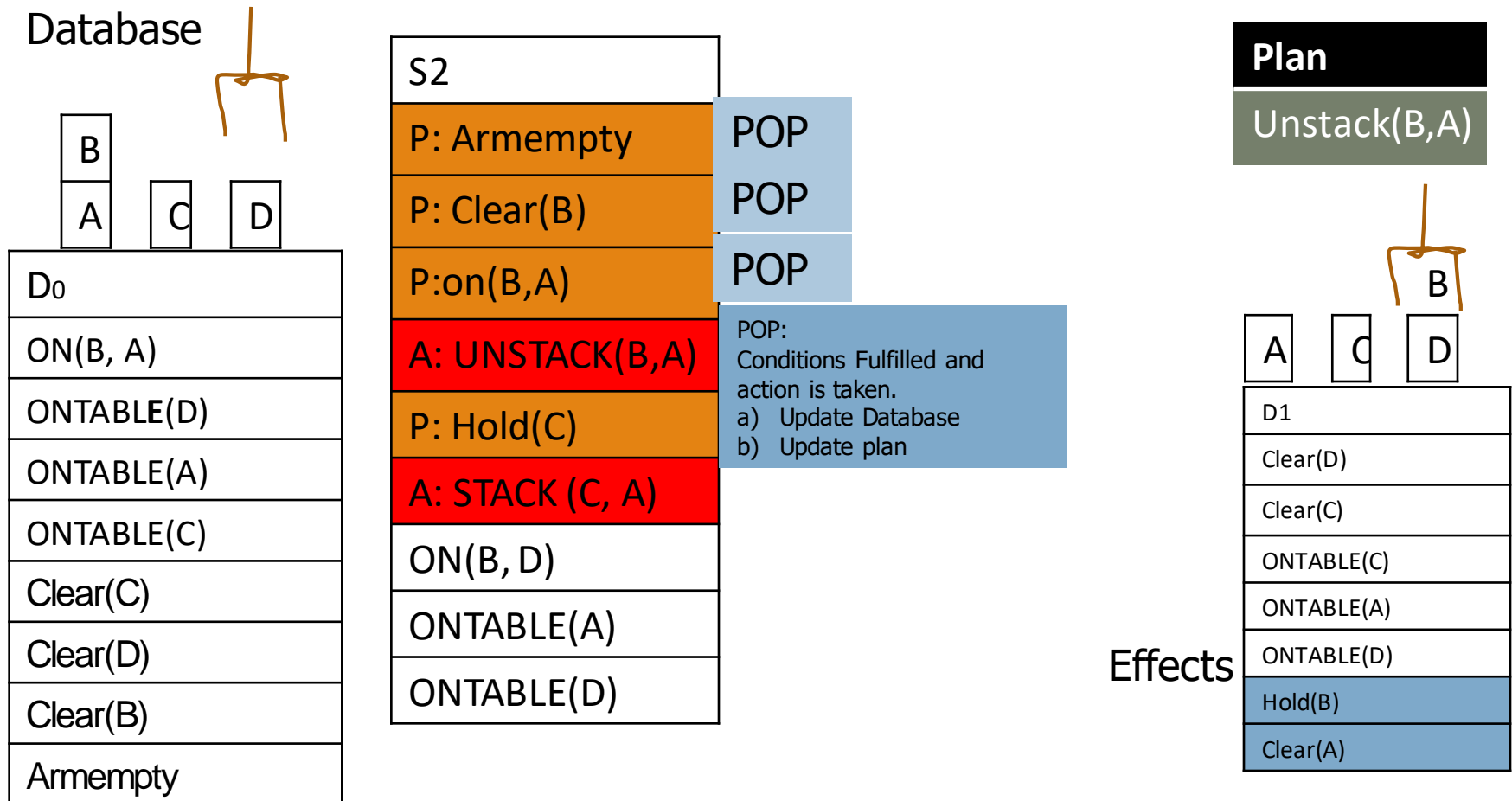
Plan

--

# Using Goal Stack Planning

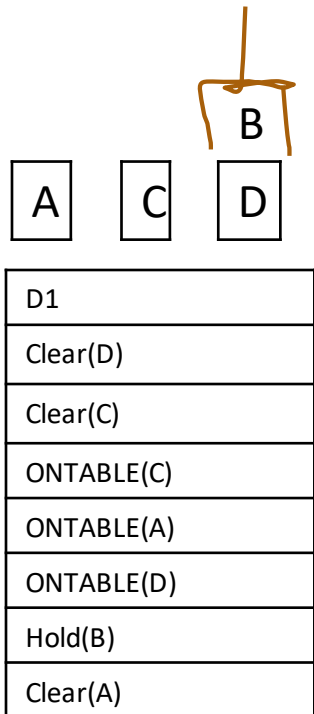


# Using Goal Stack Planning



# Using Goal Stack Planning

Database



S3
P: Hold(C)
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

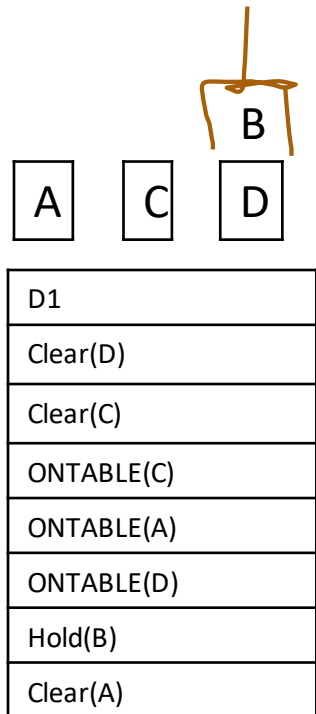
S4
P: ARMEMPTY
P: ONTABLE(C)
P: Clear(C)
A: PICKUP(C)
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

Plan

UNSTACK(B,A)

# Using Goal Stack Planning

## Database



S4
P: ARMEMPTY
P: ONTABLE(C)
P: Clear(C)
A: PICKUP(C)
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

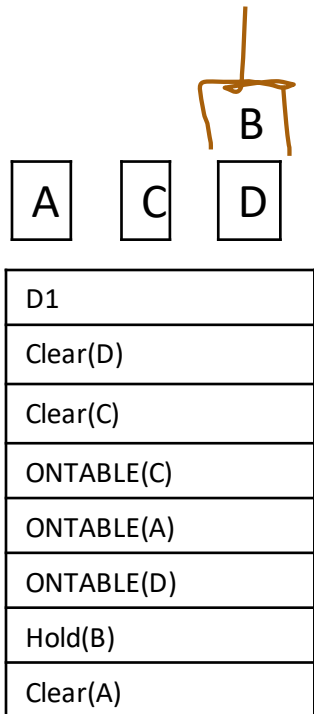
S5
P: Hold(B)
A: PUTDOWN(B)
P: ONTABLE(C)
P: Clear(C)
A: PICKUP(C)
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

## Plan

UNSTACK(B,A)

# Using Goal Stack Planning

## Database



S5
P: Hold(B)
A: PUTDOWN(B)
P: ONTABLE(C)
P: Clear(C)
A: PICKUP(C)
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

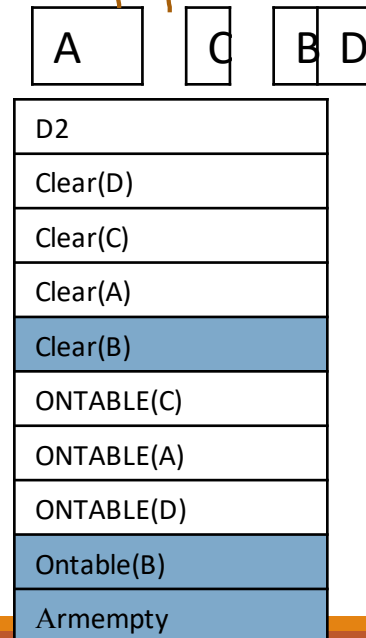
POP

Action  
Taken

S6
P: ONTABLE(C)
P: Clear(C)
A: PICKUP(C)
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)


## Plan

UNSTACK(B,A)  
PUTDOWN(B)



# Using Goal Stack Planning

## Database




A	C	B	D
<b>D2</b>			
Clear(D)			
Clear(C)			
Clear(A)			
Clear(B)			
ONTABLE(C)			
ONTABLE(A)			
ONTABLE(D)			
ONTABLE(B)			
Armempty			

<b>S6</b>		
P:ONTABLE(C)	POP	
P: Clear(C)	POP	
A: PICKUP(C)	POP: Action taken	
A: STACK (C, A)		
ON(B, D)		
ONTABLE(A)		
ONTABLE(D)		

<b>S7</b>	
A: STACK (C, A)	
ON(B, D)	
ONTABLE(A)	
ONTABLE(D)	

## Plan

UNSTACK(B,A)  
 PUTDOWN(B)  
**Pickup(C)**

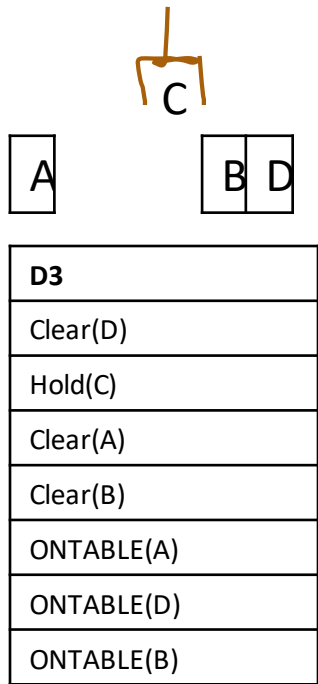


A	B	D
<b>D3</b>		
Clear(D)		
Hold(C)		
Clear(A)		
Clear(B)		
ONTABLE(A)		
ONTABLE(D)		
ONTABLE(B)		



# Using Goal Stack Planning

## Database



S7
A: STACK (C, A)
ON(B, D)
ONTABLE(A)
ONTABLE(D)

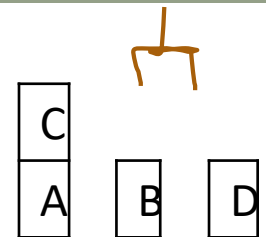
POP: Action taken

S8
ON(B, D)
ONTABLE(A)
ONTABLE(D)

<b>D4</b>
Clear(D)
Clear(B)
ONTABLE(A)
ONTABLE(D)
ONTABLE(B)
ON(C,A)
Clear(C)
Armempty

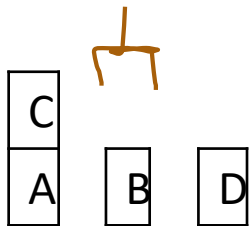
## Plan

UNSTACK(B,A)  
 PUTDOWN(B)  
 Pickup(C)  
 STACK(C,A)



# Using Goal Stack Planning

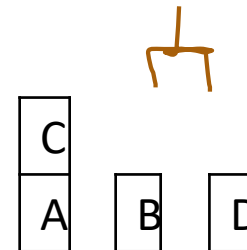
## Database



D4
Clear(D)
Clear(B)
ONTABLE(A)
ONTABLE(D)
ONTABLE(B)
ON(C,A)
Clear(C)
Armempty

S8
ON(B, D)
ONTABLE(A)
ONTABLE(D)

S9
P:Hold(B)
P:Clear(D)
A: STACK(B, D)
ONTABLE(A)



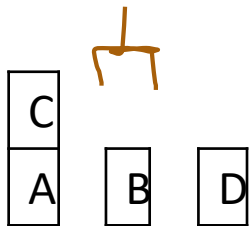
D4
Clear(D)
Clear(B)
ONTABLE(A)
ONTABLE(D)
ONTABLE(B)
ON(C,A)
Clear(C)
Armempty

## Plan

UNSTACK(B,A)  
 PUTDOWN(B)  
 Pickup(C)  
 STACK(C,A)

# Using Goal Stack Planning

## Database



D4
Clear(D)
Clear(B)
ONTABLE(A)
ONTABLE(D)
ONTABLE(B)
ON(C,A)
Clear(C)
Armempty

S9
P:Hold(B)
P:Clear(D)
A: STACK(B, D)
ONTABLE(A)

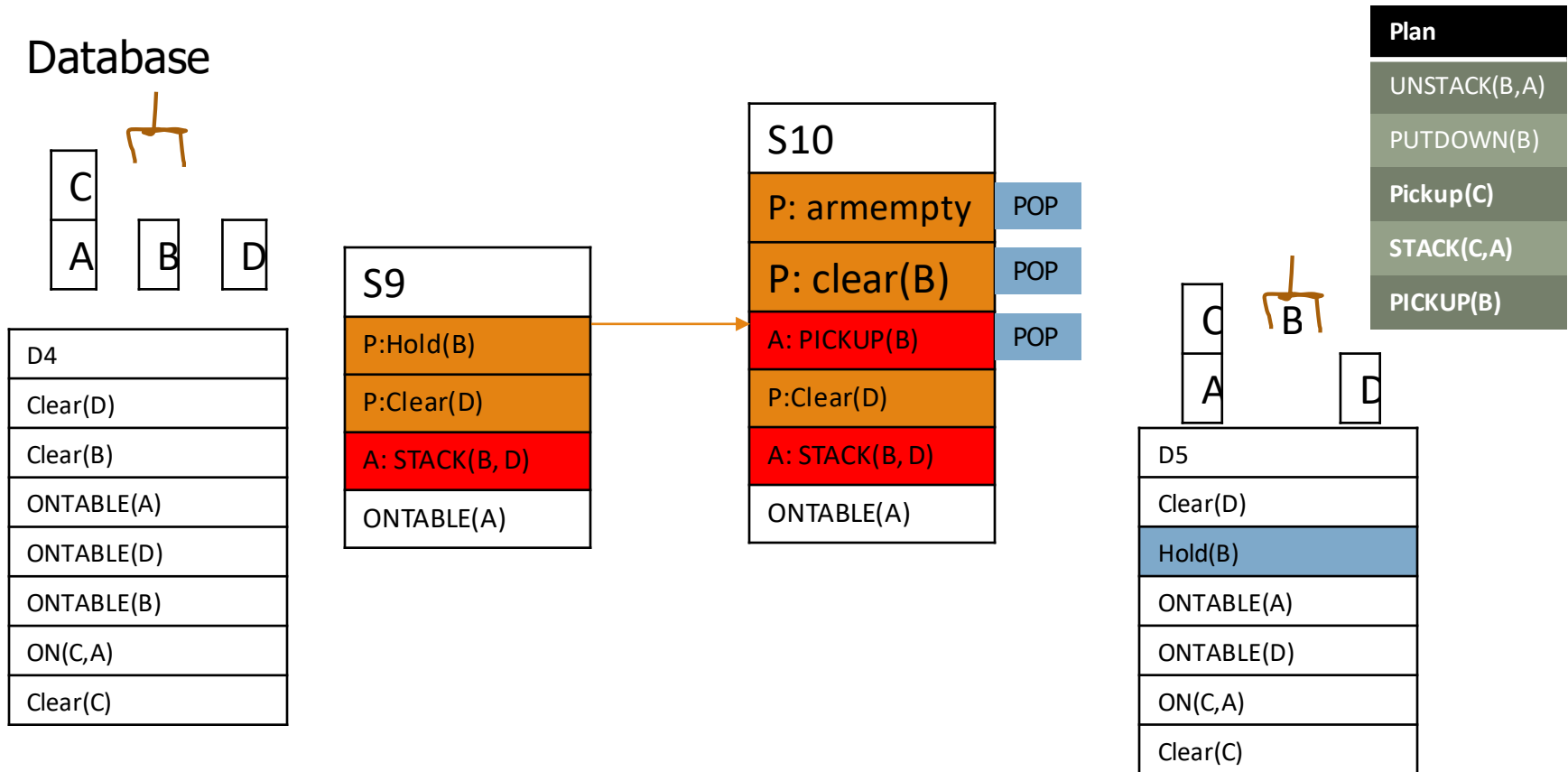
S10
P: armempty
P: clear(B)
A: PICKUP(B)
P:Clear(D)
A: STACK(B, D)
ONTABLE(A)

POP  
POP  
POP

D5
Clear(D)
Hold(B)
ONTABLE(A)
ONTABLE(D)
ON(C,A)
Clear(C)

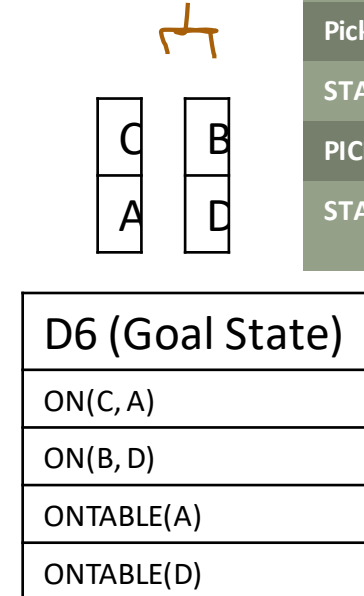
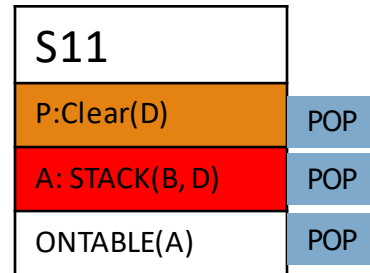
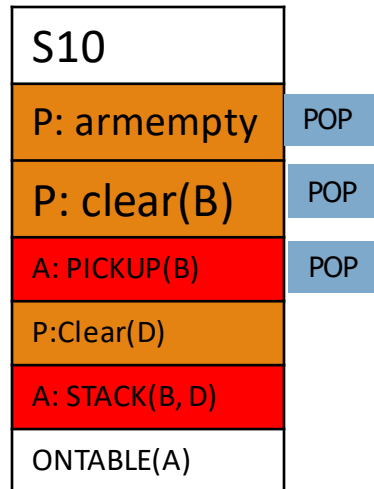
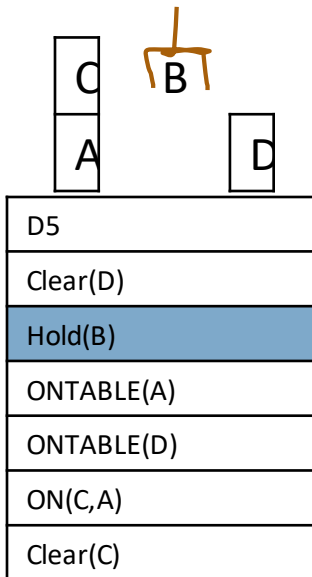
Plan
UNSTACK(B,A)
PUTDOWN(B)
Pickup(C)
STACK(C,A)
PICKUP(B)

# Using Goal Stack Planning



# Using Goal Stack Planning

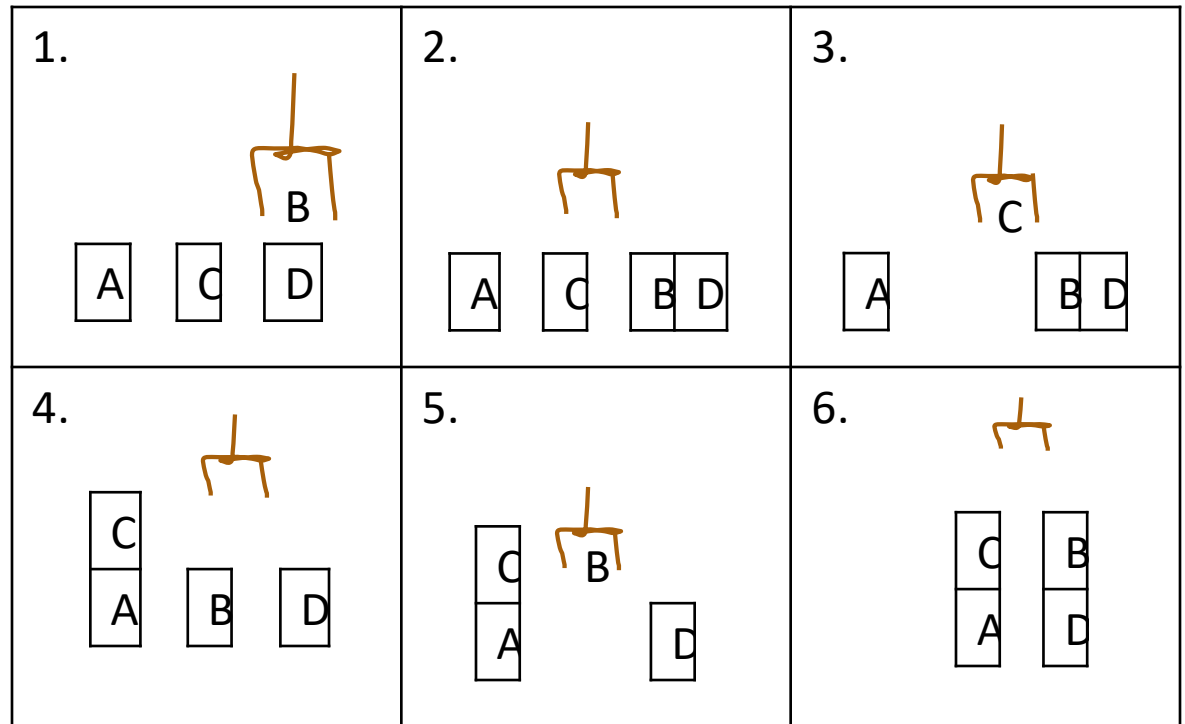
## Database



Plan
UNSTACK(B,A)
PUTDOWN(B)
Pickup(C)
STACK(C,A)
PICKUP(B)
STACK(B,D)

# Final Plan

Plan
1.UNSTACK(B,A)
2.PUTDOWN(B)
3.Pickup(C)
4.STACK(C,A)
5.PICKUP(B)
6.STACK(B,D)



# Summary

---

Goal state is like a node in a search tree, if there is a choice of action, we create branches

Planning helps the agent for fast and efficient execution of the actions to achieve specified goals.