Université de M'Sila
Faculté des mathématiques et de l'informatique
Département d'informatique

PLANNING

Dr. Abdelbasset Barkat

Planning: general

- Planning is the task of finding a sequence of actions that will achieve a goal.
- The representation of planning problems concerns states, actions and objectives

STRIPS

STRIPS (*STanford Research Institute Problem Solver*) is an automated planner developed by **Richard Fikes** and **Nils Nilsson** in 1971. The same name was later used to refer to the formal language of the inputs to this planner. This language is the base for most of the languages for expressing automated planning problem instances in use today; such languages are commonly known as action languages.

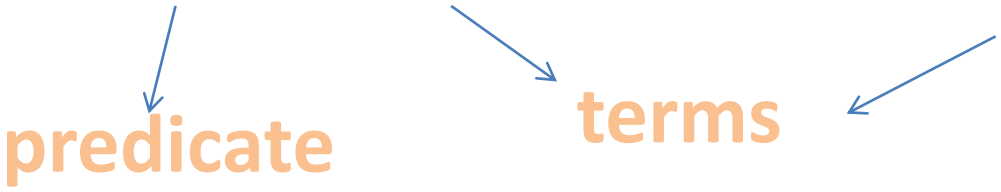
STRIPS language components

- States are represented by a set of facts or proposition. (see predicate logic or first-order logic)

the facts or proposition:

- Ex. We want to represent the fact that someone wrote something: Simple atomic formula:

HAS-WRITTEN (Mohamed, Artificial Intelligence)



- The predicate often corresponds to the verb of the sentence: “The room is beige”

BEIGE(ROOM)

COLOR (ROOM 1, BEIGE)

VALUE(COLOR, ROOM1, BEIGE)

The connectors

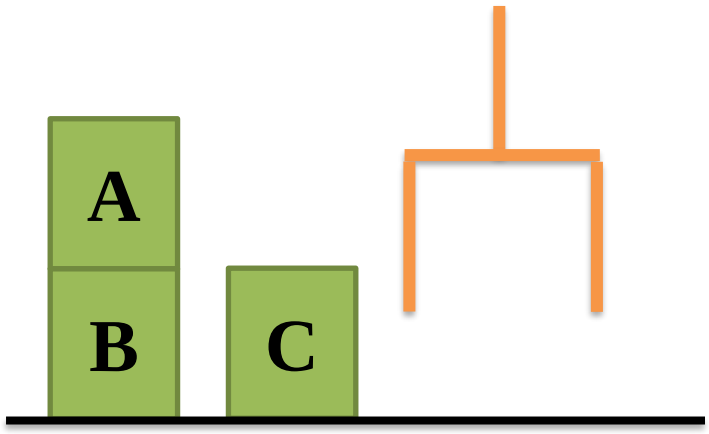
We can combine facts to form more complex facts using connectors or operators

$\wedge$ (and), V (or), => (implication) $\Leftrightarrow$ (equivalence)

Ex. TEACH (Med, ROOM1) $\wedge$ COLOR (ROOM1, BEIGE)

A state of the world...

State representation: Planners decompose the world into logical conditions and represent a state as a conjunction of predicates (first-order logical facts).

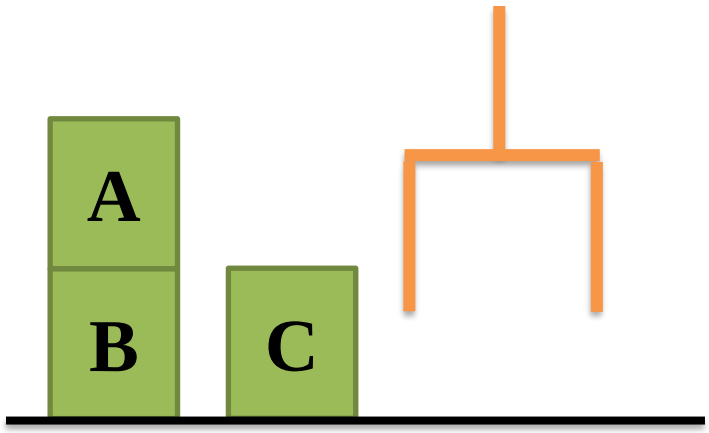


- OnTable(C)
- OnTable(B)
- On(A,B)
- Free(C)
- Free(A)
- EmptyHand

An action on the states of the world…..

Representation of actions: an action is specified in terms of **preconditions** that must hold before it can be executed, and in terms of **effects** that follow when it is executed

An action is applicable to each state that satisfies the precondition; otherwise the action has no effect.

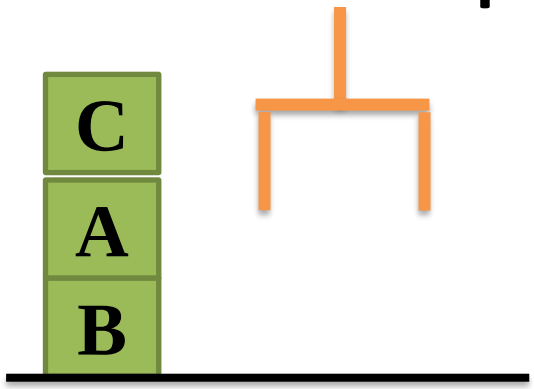


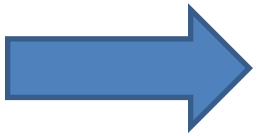
Action *Take(C)*

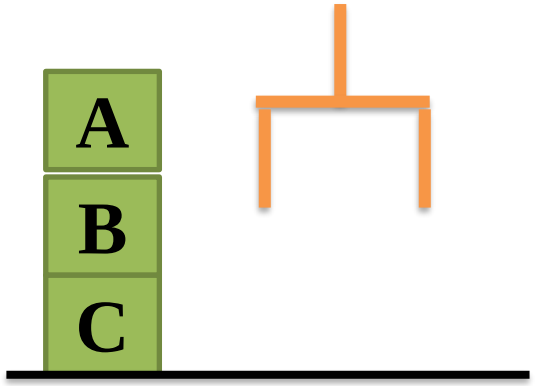
preconditions OnTable(C)
Free(C)
EmptyHand

effects Hold(C)

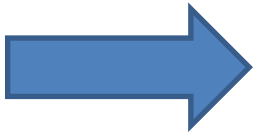
Example: the world of blocks







- EmptyHand
- On(A,B)
- On(C,A)
- OnTable(B)
- Free(C)



- EmptyHand
- On(A,B)
- On(B,C)
- OnTable(C)
- Free(A)

The actions

Action	preconditions	effects
• Stack(X,Y)	• Free(Y) • Hold(X)	• On(X,Y) • Free(X) • EmptyHand

The actions

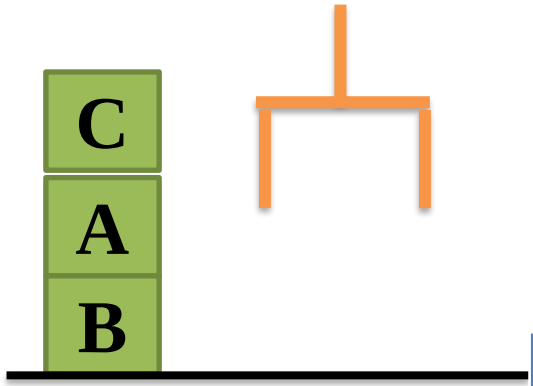
Action	preconditions	effects
• Stack(X,Y)	• Free(Y) • Hold(X)	• On(X,Y) • Free(X) • EmptyHand
• UnStack(X,Y)	• On(X,Y) • Free(X) • EmptyHand	• Hold(X) • Free(Y)

The actions

Action	preconditions	effects
• Stack(X,Y)	• Free(Y) • Hold(X)	• On(X,Y) • Free(X) • EmptyHand
• UnStack(X,Y)	• On(X,Y) • Free(X) • EmptyHand	• Hold(X) • Free(Y)
• Take(X)	• OnTable(X) • Free(X) • EmptyHand	• Hold(X)

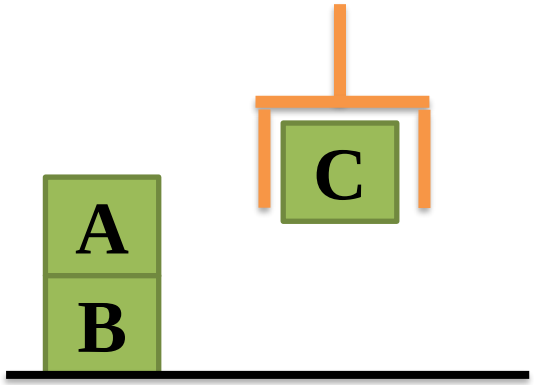
The actions

Action	preconditions	effects
• Stack(X,Y)	• Free(Y) • Hold(X)	• On(X,Y) • Free(X) • EmptyHand
• UnStack(X,Y)	• On(X,Y) • Free(X) • EmptyHand	• Hold(X) • Free(Y)
• Take(X)	• OnTable(X) • Free(X) • EmptyHand	• Hold(X)
• Put(X)	• Hold(X)	• OnTable(X) • Free(X) • EmptyHand

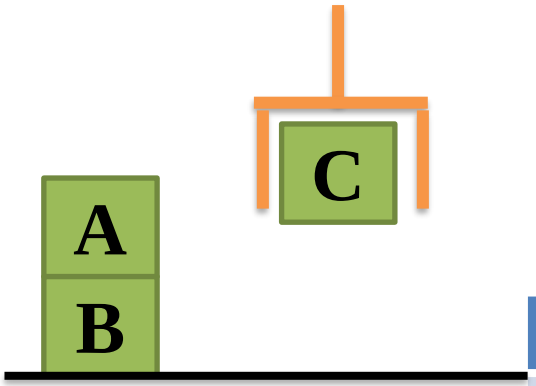


- EmptyHand
- On(A,B)
- On(C,A)
- OnTable(B)
- Free(C)

Action	UnStack(C,A)
preconditions	On(C,A) Free(C) EmptyHand
effects	Hold(C) Free(A)

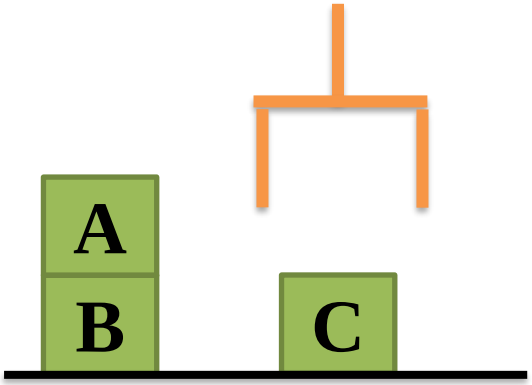


- On(A,B)
- OnTable(B)
- Hold(C)
- Free(A)

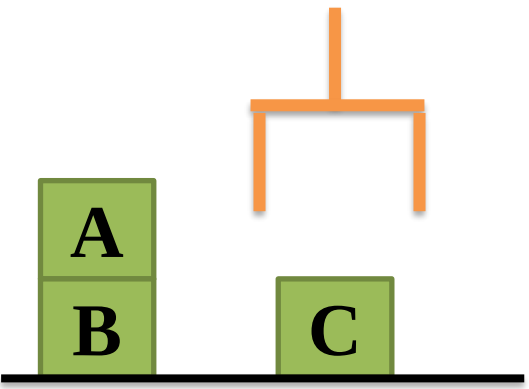


- On(A,B)
- OnTable(B)
- Hold(C)
- Free(A)

Action	Put(C)
preconditions	Hold(C)
effects	OnTable(C) Free(C) EmptyHand

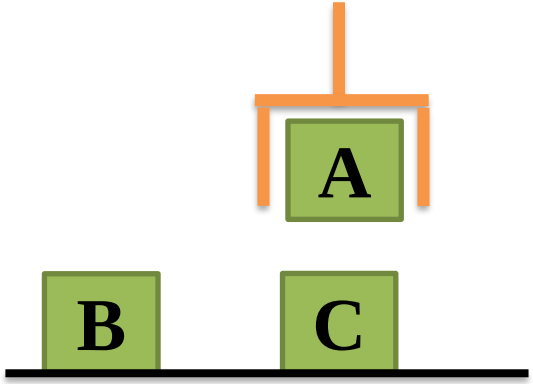


- On(A,B)
- OnTable(B)
- Free(A)
- OnTable(C)
- Free(C)
- EmptyHand

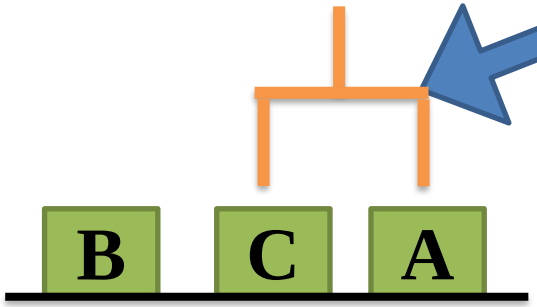


- On(A,B)
- OnTable(B)
- Free(A)
- OnTable(C)
- Free(C)
- EmptyHand

Action	UnStack(A,B)
preconditions	Free(A) On(A,B) EmptyHand
effects	Hold(A) Free(B)

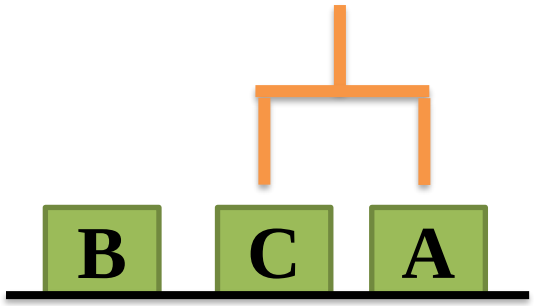


- OnTable(B)
- OnTable(C)
- Free(C)
- Hold(A)
- Free(B)



- OnTable(B)
- OnTable(C)
- OnTable(A)
- Free(A)
- Free(C)
- Free(B)
- EmptyHand

Action	Put(A)
preconditions	Hold(A)
effects	OnTable(A) Free(A) EmptyHand



- OnTable(B)
- OnTable(C)
- OnTable(A)
- Free(A)
- Free(C)
- Free(B)
- EmptyHand

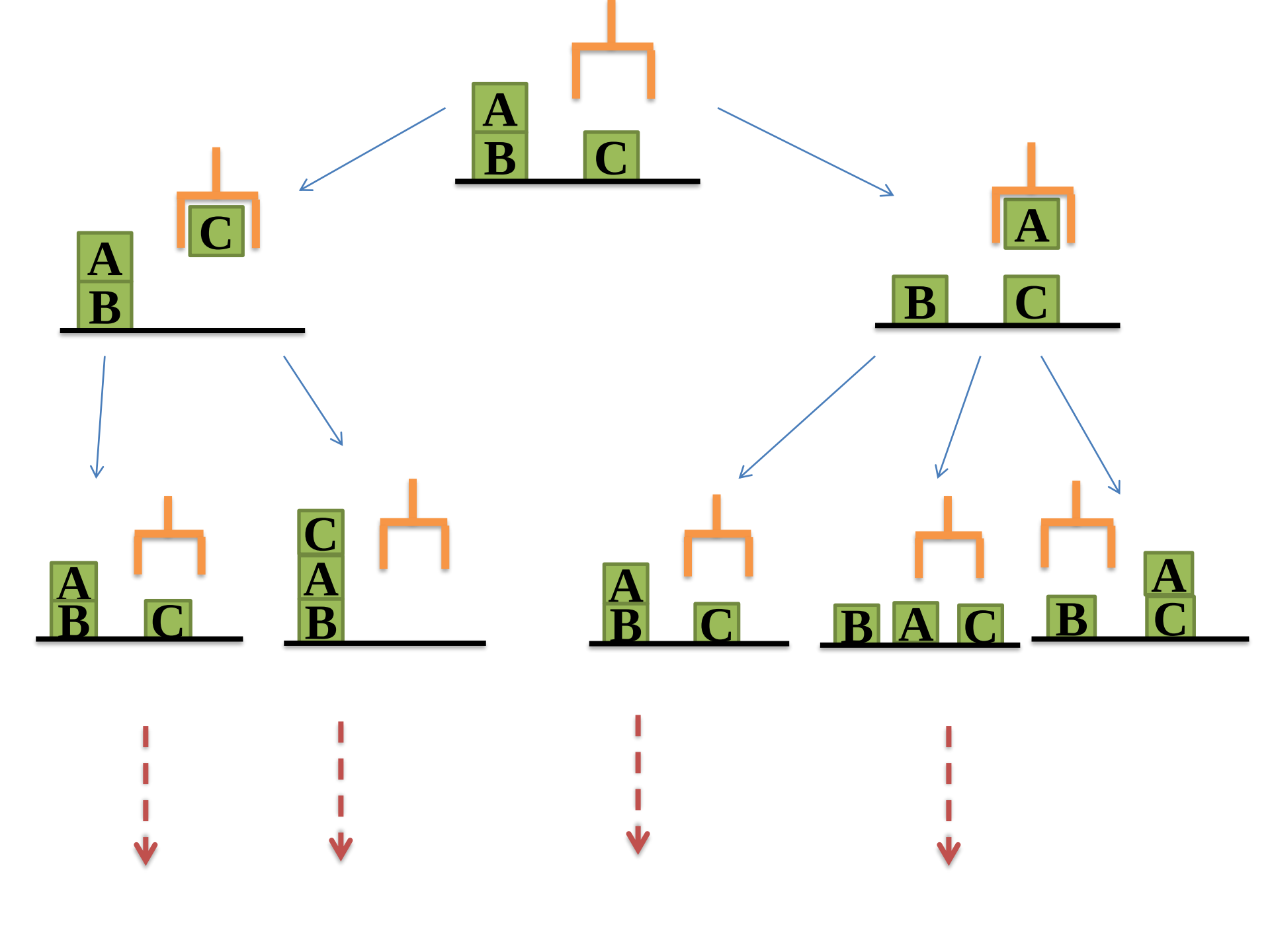
Action	Take(A)

Action	Take(B)

Action	Take(C)

Planning as Search in State-Space

The most direct approach is the use of state-space search Since the descriptions of actions in a planning problem specify both, namely preconditions and effects, it is possible to search in both directions: either forward chaining (forward) from the initial state, or backward chaining (backward) from the objective


A
B
C
C
A
B
A
B
C
A
B
C
C
A
B
A
B
C
B
A
C
B
A
C

Pseudo Code

A = the set of all actions
S = Initial state
G = Goal state

STRIPS(*A*, *S G*)
p = empty plan
loop…
if *S* satisfies *G* then return *p*
a = [an applicable action in *A*, relevant for *G*]
if *a* = null, then return failure
p' = STRIPS(*A*, *S*, preconditions(*a*))
if *p'* = failure, then return failure
s = apply *p'* to *s*
s = apply *a* to *s*
p = *p* + *p'* + *a*

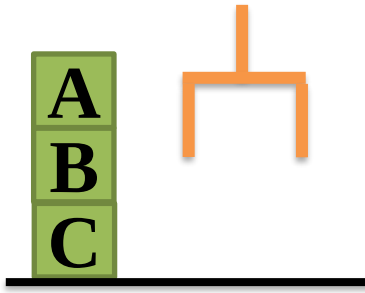
Goal stack planning pseudocode

Push the original goal on the stack. Repeat until the stack is empty:

- if stack top is a compound goal, push its unsatisfied subgoals on the stack.
- If stack top is a single unsatisfied goal, replace it by an action that makes it satisfied and push the action's precondition on the stack.
- If stack top is an action, pop it from the stack, execute it and change the knowledge base by the action's effects.
- If stack top is a satisfied goal, pop it from the stack.

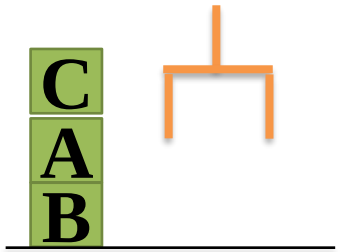
EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

Final state



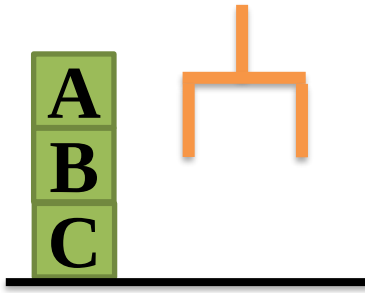
EmptyHand ,On(A,B), On(C,A), OnTable(B) , Free(C)

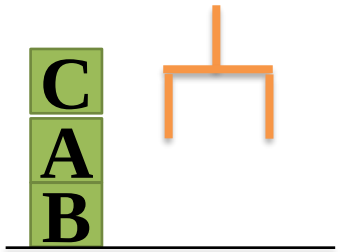
Initial state



EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

EmptyHand ,On(A,B), On(C,A), OnTable(B) , Free(C)





Free(A)

OnTable(C)

On(B,C)

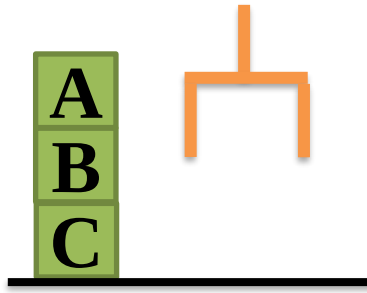
On(A,B)

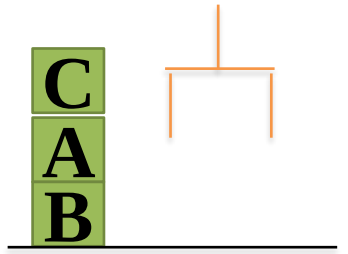
EmptyHand

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

EmptyHand ,On(A,B), On(C,A), OnTable(B) , Free(C)





On(C,A)
Free(C)

EmptyHand

EmptyHand, Free(C), On(C,A)

True

UnStack(C,A)

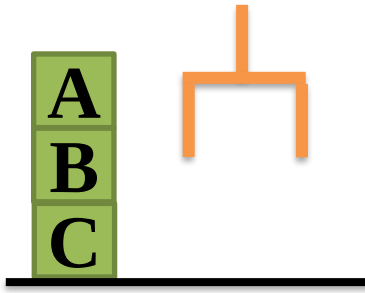
Execute the Unstack(C,A) action and change of the knowledge base

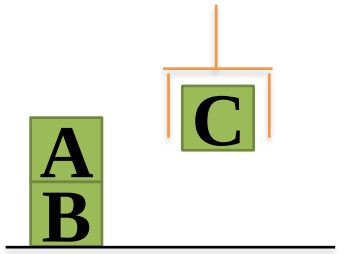
OnTable(C)
On(B,C)
On(A,B)
EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , Hold(C), Free(A)





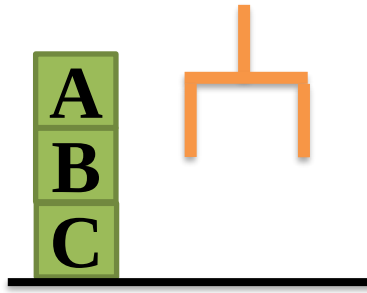
OnTable(C)
On(B,C)
On(A,B)
EmptyHand
On(A,B), On(B,C), OnTable(C), Free(A)

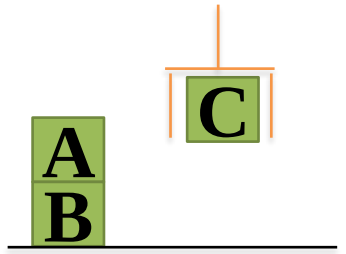
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , Hold(C), Free(A)





Hold(C)

Put(C)

On(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

True

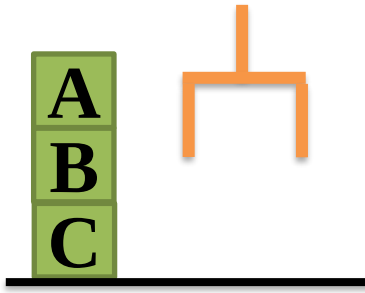
Execute the Put(C) action and change of the knowledge base

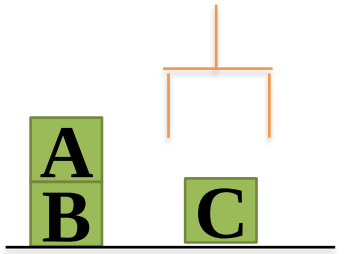
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





On(B,C)
On(A,B)
EmptyHand
On(A,B), On(B,C), OnTable(C), Free(A)

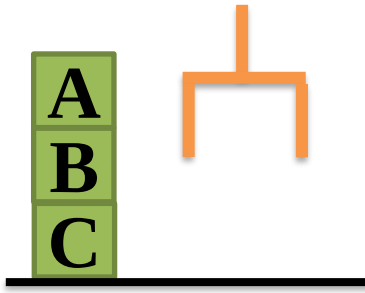
Put(C)

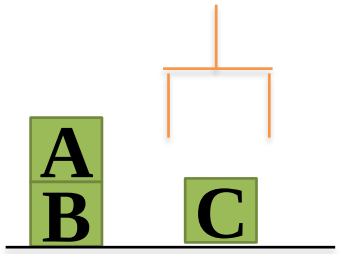
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

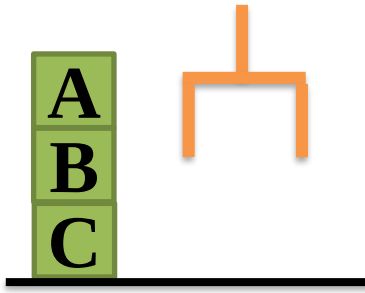
Put(C)

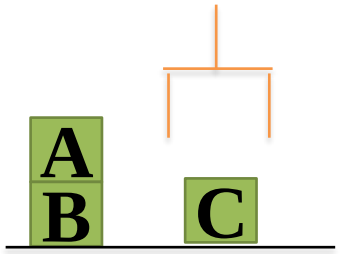
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Hold(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

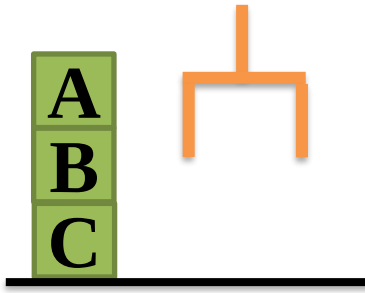
Put(C)

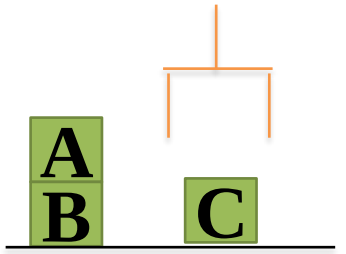
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

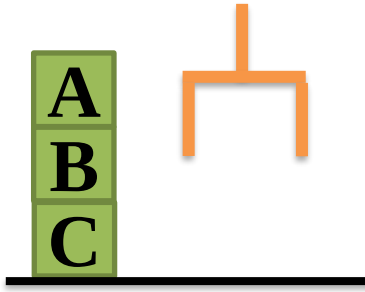
Put(C)

UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)



OnTable(B)

EmptyHand

Satisfait

Free(B)

OnTable(B), EmptyHand, Free(B)

Take(B)

Free(C)

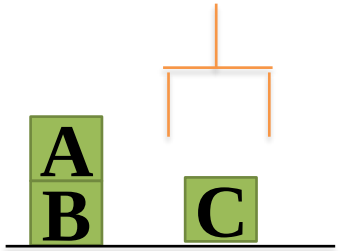
Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)



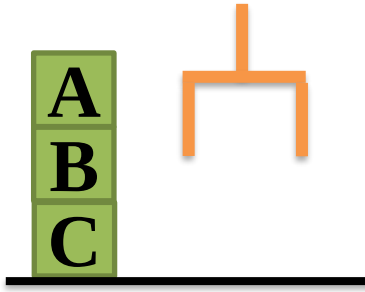
Put(C)

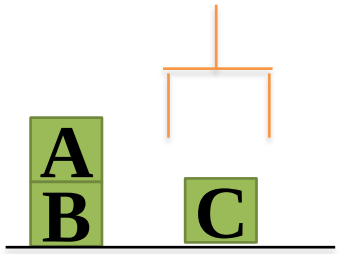
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Unstack(A,B)

OnTable(B), EmptyHand, Free(B)

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

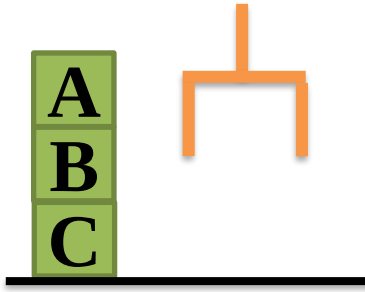
Put(C)

UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)



On(A,B), EmptyHand, Free(A)

Unstack(A,B)

OnTable(B), EmptyHand, Free(B)

Take(B)

Free(C)

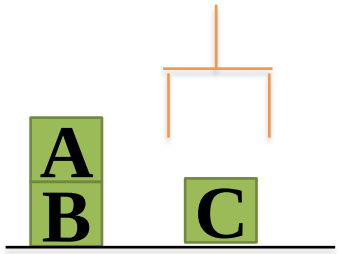
Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)



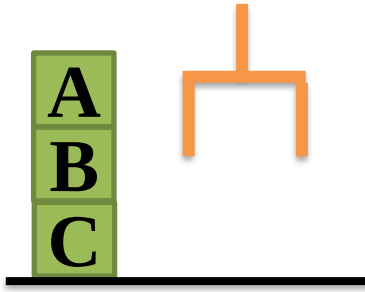
Put(C)

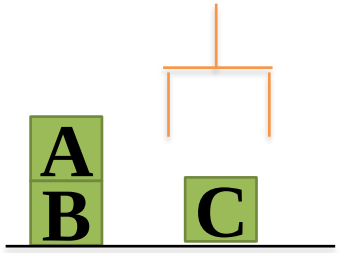
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Unstack(A,B)

OnTable(B), EmptyHand, [illegible](B)

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Execute the Unstack(A,B) action and change of the knowledge base

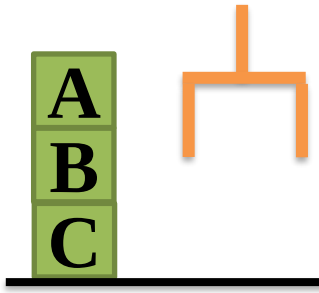
Put(C)

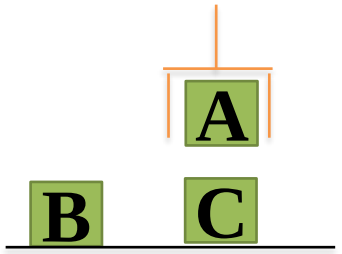
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





OnTable(B), EmptyHand, Free(B)

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

UnStack(A,B)

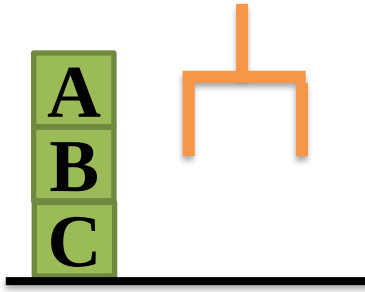
Put(C)

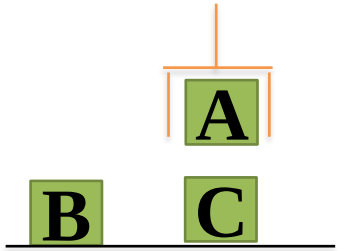
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





OnTable(B), Free(B), EmptyHand

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

UnStack(A,B)

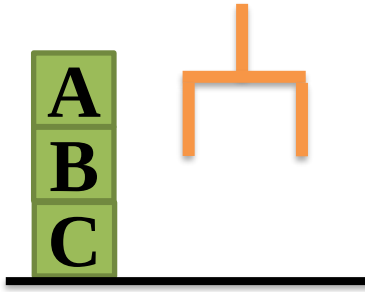
Put(C)

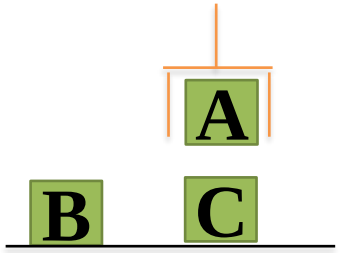
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





EmptyHand

OnTable(B), Free(B), EmptyHand

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

UnStack(A,B)

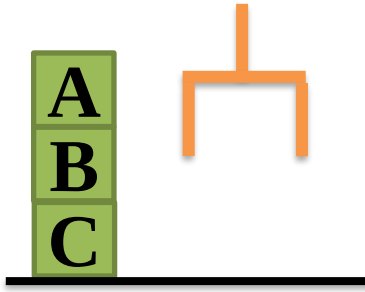
Put(C)

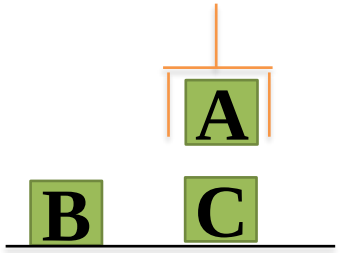
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Put(A)

OnTable(B), Free(B), EmptyHand

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

UnStack(A,B)

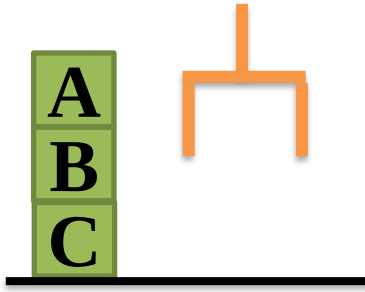
Put(C)

UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)



Satisfied

Hold(A)

Put(A)

OnTable(B), Free(B), EmptyHand

Take(B)

Free(C)

Hold(B), Free(C)

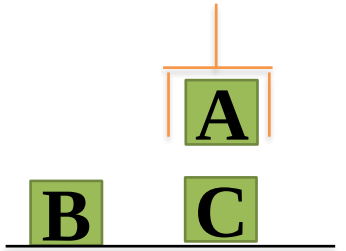
Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Execute the Put(C) action and change of the knowledge base



UnStack(A,B)

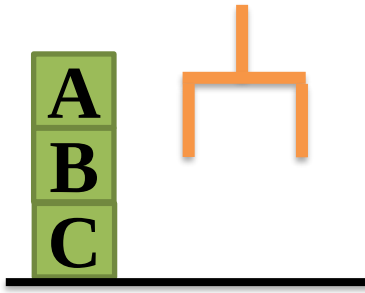
Put(C)

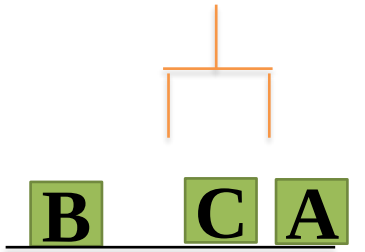
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





OnTable(B), Free(B), EmptyHand

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Put(A)

UnStack(A,B)

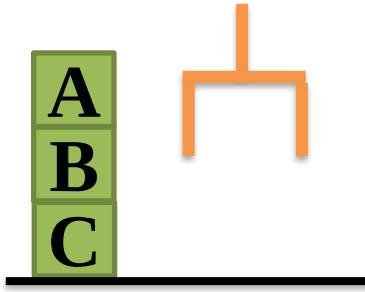
Put(C)

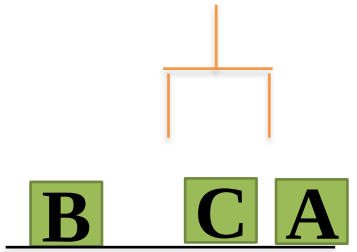
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





OnTable(B), Free(B), EmptyHand

Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Put(A)

UnStack(A,B)

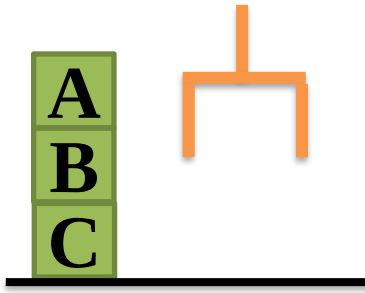
Put(C)

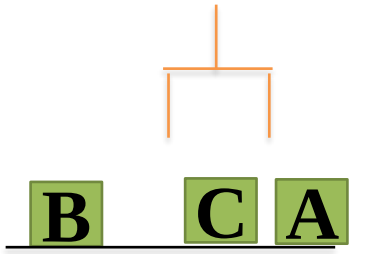
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Take(B)

Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Put(A)

UnStack(A,B)

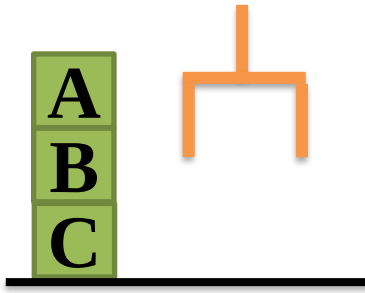
Put(C)

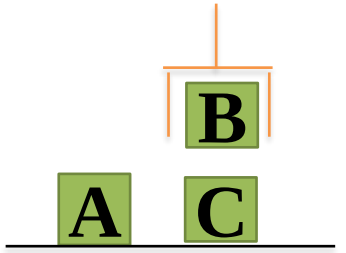
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Free(C)

Hold(B), Free(C)

Stack(B,C)

On(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Take(B)

Put(A)

UnStack(A,B)

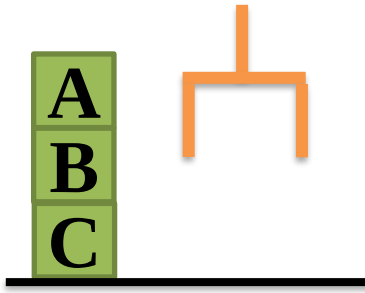
Put(C)

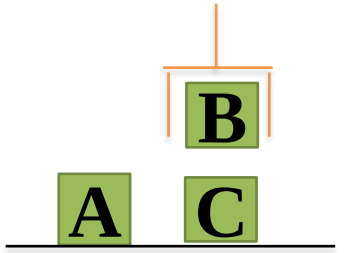
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Take(B)

Put(A)

UnStack(A,B)

Put(C)

UnStack(C,A)

P

Stack(B,C)

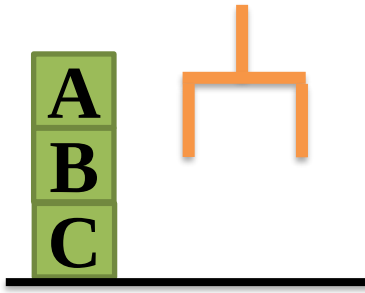
On(A,B)

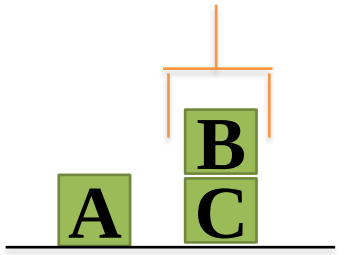
EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

Put(C)

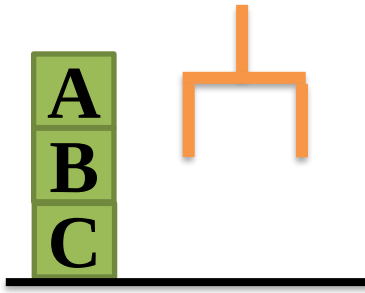
UnStack(C,A)

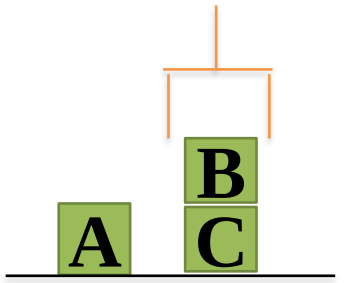
P

On(A,B)
EmptyHand
On(A,B), On(B,C), OnTable(C), Free(A)

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Hold(A)

Free(B)

Hold(A), Free(B)

Stack(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

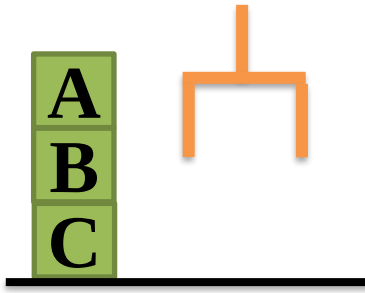
Put(C)

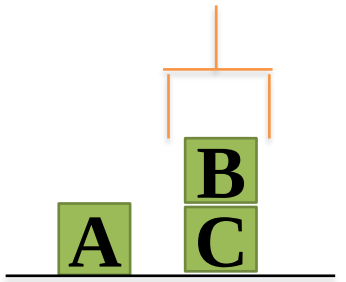
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





OnTable(A), Free(A), EmptyHand

Take(A)

Free(B)

Hold(A), Free(B)

Stack(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

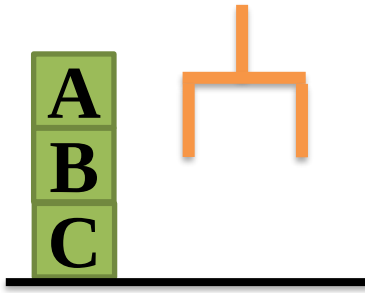
Put(C)

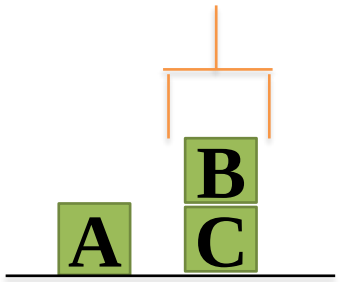
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





OnTable(A), Free(A), EmptyHand

Take(A)

Free(B)

Hold(A), Free(B)

Stack(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Satisfait

Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

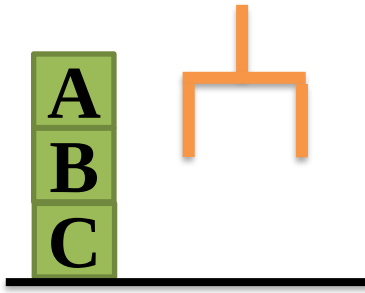
Put(C)

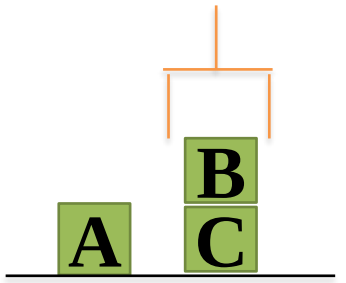
UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)





Take(A)

Free(B)

Hold(A), Free(B)

Stack(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

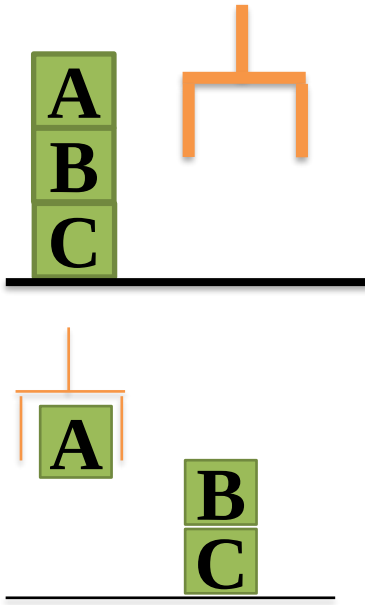
Put(C)

UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)



Free(B)

Hold(A), Free(B)

Stack(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Take(A)

Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

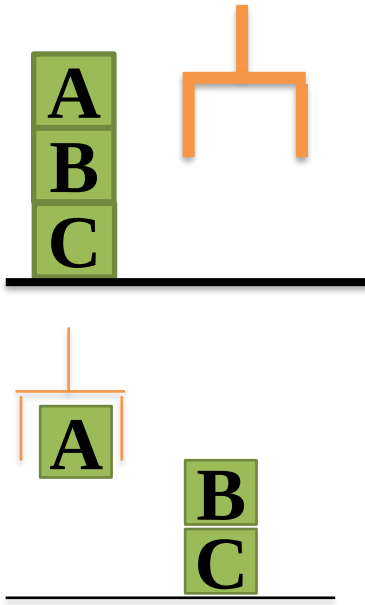
Put(C)

UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)



Free(B)

Hold(A), Free(B)

Stack(A,B)

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

Satisfait

Take(A)

Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

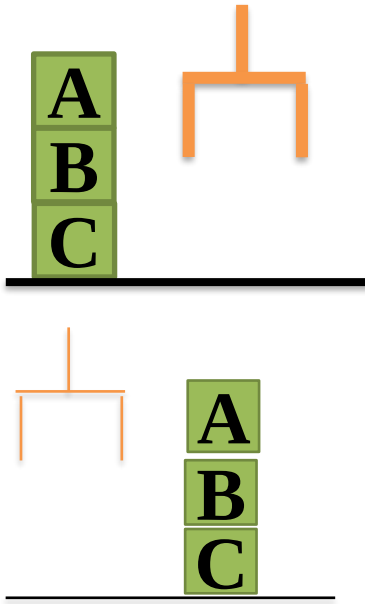
Put(C)

UnStack(C,A)

P

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)



Stack(A,B)

Take(A)

Stack(B,C)

Take(B)

Put(A)

UnStack(A,B)

Put(C)

UnStack(C,A)

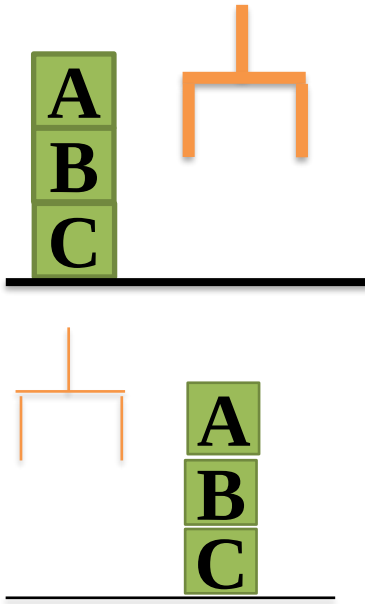
P

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

On(A,B), OnTable(B) , OnTable(C), Free(A), Free(C)



Stack(A,B)

Take(A)

Stack(B,C)

Take(B)

Put(C)

UnStack(A,B)

Put(C)

UnStack(C,A)

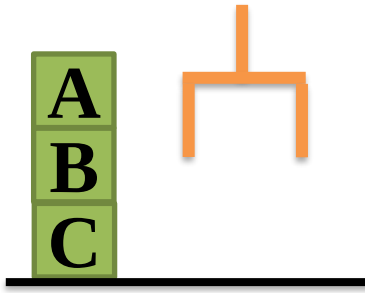
P

EmptyHand

On(A,B), On(B,C), OnTable(C), Free(A)

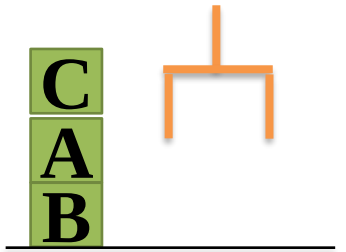
EmptyHand, On(A,B), On(B,C), OnTable(C), Free(A)

Final state



EmptyHand ,On(A,B), On(C,A), OnTable(B) , Free(C)

initial state



Stack(A,B)

Take(A)

Stack(B,C)

Take(B)

Put(C)

UnStack(A,B)

Put(C)

UnStack(C,A)

P