DEDUCTIVE PLANNING

Let us consider the following initial state expressed with the Kovalsky formulation

holds(on(a,d),s0).

holds(on(b,e),s0).

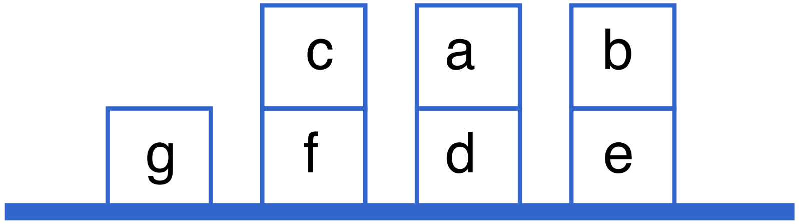
holds(on(c,f),s0).

holds(clear(a),s0).

holds(clear(b),s0).

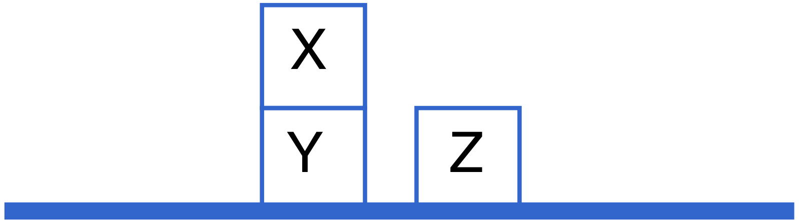
holds(clear(c),s0).

holds(clear(g),s0).

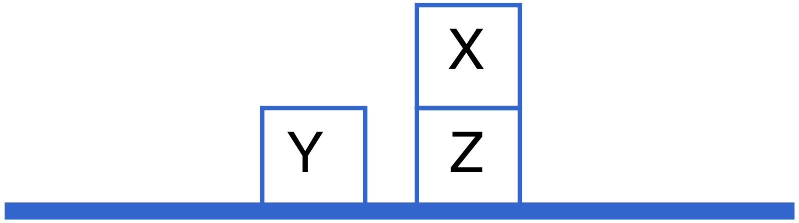


DEDUCTIVE PLANNING

Consider the action move(X,Y,Z)



on(X,Y)
clear(X)
clear(Z)



clear(Y)
on(X,Z)

DEDUCTIVE PLANNING

```
%Effects move(X,Y,Z):

holds(clear(Y),do(move(X,Y,Z),S)).
holds(on(X,Z),do(move(X,Y,Z),S)).

%Frame condition for move(X,Y,Z):

holds(V,do(move(X,Y,Z),S)):-
    holds(V,S),
    V\=clear(Z),
    V\=on(X,Y).
```

DEDUCTIVE PLANNING

```
% Clause for the preconditions of move(X,Y,Z):
pact(move(X,Y,Z),S):-
    holds(clear(X),S), holds(clear(Z),S),
    holds(on(X,Y),S), X\=Z.

%Clause for the reachability of a state:
poss(s0).
poss(do(A,S)):-
    poss(S),
    pact(A,S).
```

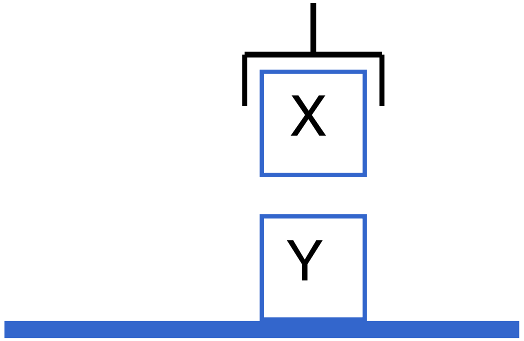
DEDUCTIVE PLANNING

- Goal:

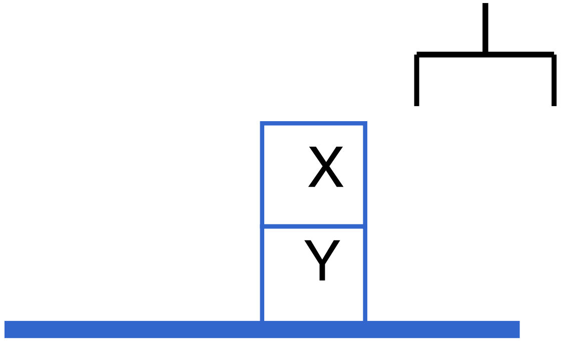
  :- poss(S), holds(on(a,b),S),holds(on(b,g),S).

DEDUCTIVE PLANNING

- Exercise: use the block world but change actiong

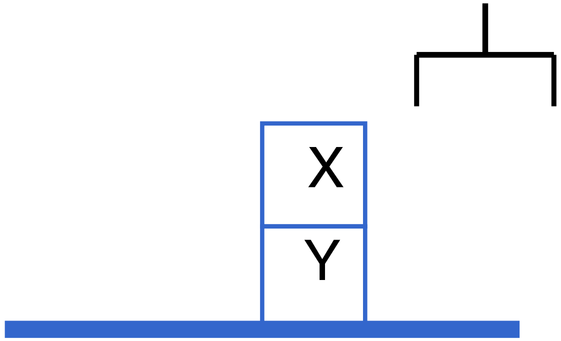


clear(Y), holding(X)

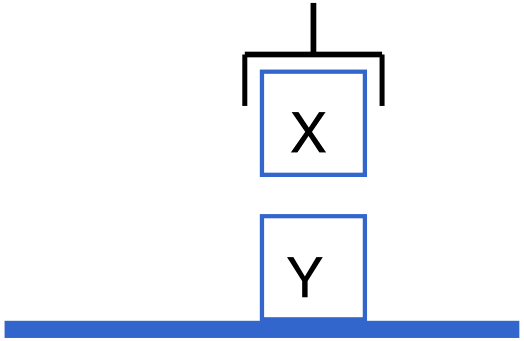


on(X,Y), handempty, clear(X)

STACK(X,Y)



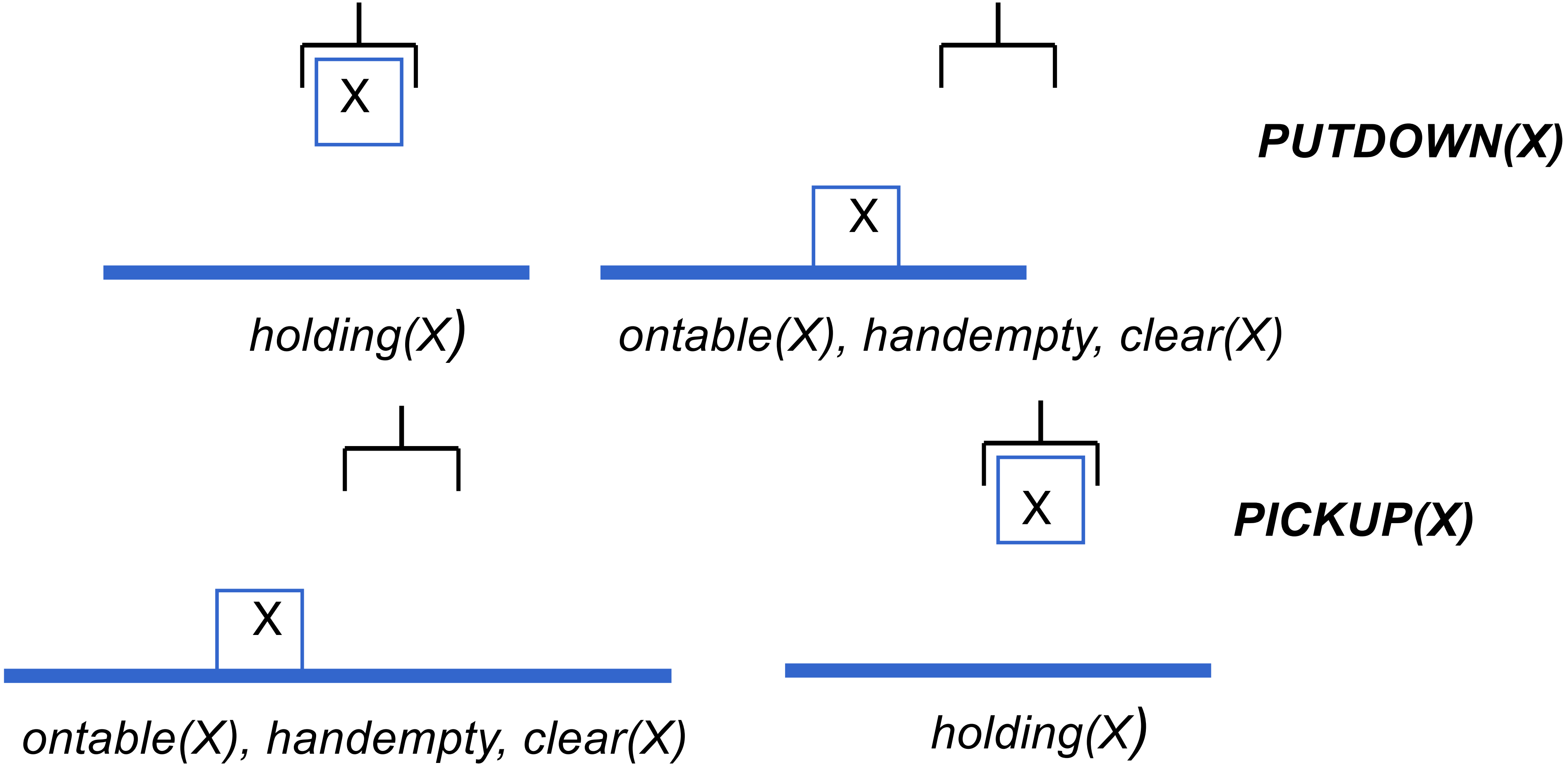
on(X,Y), handempty, clear(X)



clear(Y), holding(X)

UNSTACK(X,Y)

DEDUCTIVE PLANNING



holding(X)

ontable(X), handempty, clear(X)

PUTDOWN(X)

ontable(X), handempty, clear(X)

holding(X)

PICKUP(X)

DEDUCTIVE PLANNING

```
%Effects stack(X,Y):

holds(clear(X), do(stack(X,Y),S)).
holds(on(X,Y), do(stack(X,Y),S)).
holds(handempty, do(stack(X,Y),S)).

%Frame condition for stack(X,Y):

holds(V,do(stack(X,Y),S)):-
   holds(V,S),
   V\=clear(Y),
   V\=holding(X).
```

DEDUCTIVE PLANNING

```
% Clause for the preconditions of stack(X,Y):
pact(stack(X,Y),S):-
    holds(holding(X),S), holds(clear(Y),S).

%Clause for the reachability of a state:
poss(s0).
poss(do(A,S)):-
    poss(S),
    pact(A,S).
```

DEDUCTIVE PLANNING

```
%Effects unstack(X,Y):

holds(holding(X), do(unstack(X,Y),S)).
holds(clear(Y), do(unstack(X,Y),S)).
holds(handempty, do(stack(X,Y),S)).

%Frame condition for unstack(X,Y):
holds(V,do(stack(X,Y),S)):-
    holds(V,S),
    V\=clear(X),
    V\=handempty,
    V\= on(X,Y).
```

DEDUCTIVE PLANNING

```
% Clause for the preconditions of unstack(X,Y):
pact(unstack(X,Y),S):-
    holds(clear(X),S), holds(handempty,S),
    holds(on(X,Y),S).
```

DEDUCTIVE PLANNING

```
%Effects pickup(X):

holds(holding(X), do(pickup(X),S)).

%Frame condition for pickup(X):

holds(V,do(pickup(X),S)):-
    holds(V,S),
    V\=clear(X),
    V\=ontable(X),
    V\= handemtpy.
```

DEDUCTIVE PLANNING

% Clause for the preconditions of pickup(X):

```
pact(pickup(X),S):-
    holds(ontable(X),S), holds(clear(X),S),
    holds(handempty, S).
```

DEDUCTIVE PLANNING

%Effects putdown(X):

holds(holding(X), do(putdown(X),S)).

%Frame condition for putdown(X):

```
holds(V,do(putdown(X),S)):-
    holds(V,S),
    V\=holding(X).
```

DEDUCTIVE PLANNING

```
% Clause for the preconditions of putdown(X):
pact(putdown(X),S):-
    holds(holding(X),S).
```

DEDUCTIVE PLANNING

Model the following actions

Load of an object

```
load(Object,Trolley,Location)
PREC:   at(Object,Location), at(Trolley,Location)
ADD LIST: in(Object, Trolley)
DELETE LIST: at(Object,Location)
```

Trasporto

```
drive(Trolley,Location1,Location2)
PREC:at(Trolley,Location1), connected(Location1,Location2)
ADD LIST: at(Trolley,Location2)
DELETE LIST: at(Trolley,Location1)
```

Scaricamento di un oggetto

```
unload(Object, Trolley,Location)
PREC:at(Trolley,Location), in(Object, Trolley)
ADD LIST: at(Object,Location)
DELETE LIST: in(Object, Trolley)
```

DEDUCTIVE PLANNING

With the following initial state and goal

Initial State:
`in(carico1,carrello1), at(carrello1,milano)`
`connected(milano,bologna), connected(bologna,roma)`

Goal: **`at(carico1,roma)`**

DEDUCTIVE PLANNING

Initial State:

```
holds(in(carico1,carrello1),s0).
holds(at(carrello1,milano),s0).
connected(milano,bologna).
connected(bologna,roma).
```

Note: the connected property does not depend on the state.

Goal:

```
:- holds(at(carico1,roma),S).
```

Reachability

```
poss(s0).
poss(do(A,S)):-
    poss(S),
    pact(A,S).
```

DEDUCTIVE PLANNING

```
holds(in(Object, Trolley), do(load(Object,Trolley,Location),S)).
pact(load (Object,Trolley,Location),S):-
   holds(at(Object,Location), S),
   holds(at(Trolley,Location), S).
holds(V, do(load(Object,Trolley,Location),S):- holds(V,S),
 V\= at(Object,Location).


holds(at(Trolley, Location), do(drive(Trolley,Location1,Location2),S)).
pact(drive(Trolley,Location1,Location2),S):-
   holds(at(Trolley,Location1), S),
   conenected(Location1, Location2).
holds(V, do(drive(Trolley,Location1,Location2),S):- holds(V,S),
 V\= at(Trolley,Location1).


holds(at(Object,Location), do(unload(Object,Trolley,Location),S)).
pact(unload(Object,Trolley,Location),S):-
   holds(at(Trolley, Location), S), holds(in(Object,Trolley),S).
holds(V, do(unload(Object,Trolley,Location),S):- holds(V,S),
 V\= in(Object, Trolley).
```