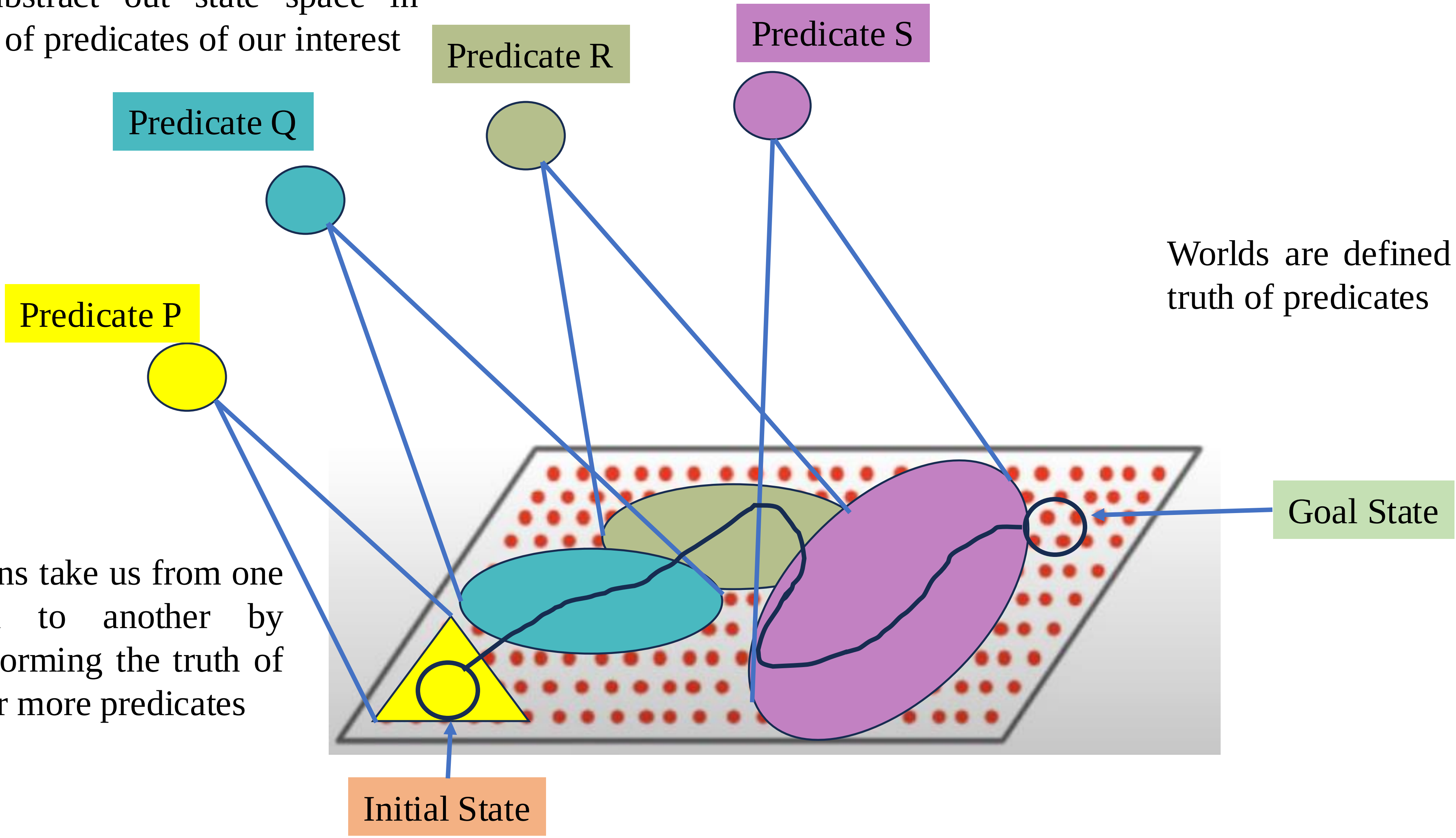
AIFA: PLANNING

24/03/2025

Koustav Rudra

State Spaces ➡ Predicate Worlds

We abstract out state space in terms of predicates of our interest

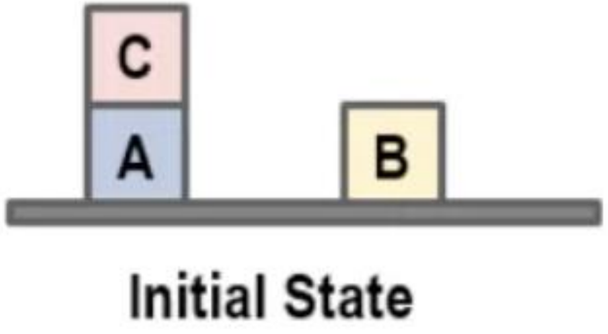


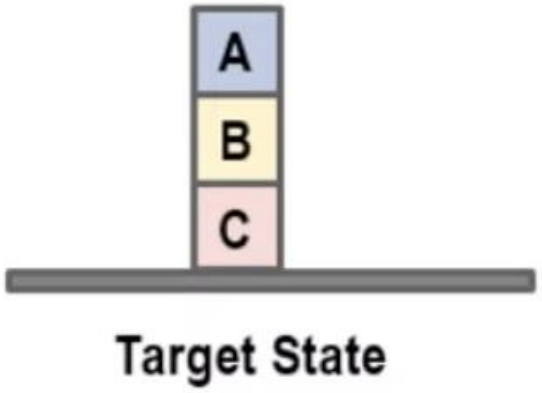
Worlds are defined in terms of truth of predicates

Actions take us from one world to another by transforming the truth of one or more predicates

Blocks World

- Classical test bed for planning algorithms





Predicates describing initial state:
- On(C,A)
- On(A, Table)
- On(B, Table)
- Clear(B)
- Clear(C)

Predicates describing initial state:
- On(A,B)
- On(B,C)
- On(C, Table)

Actions:

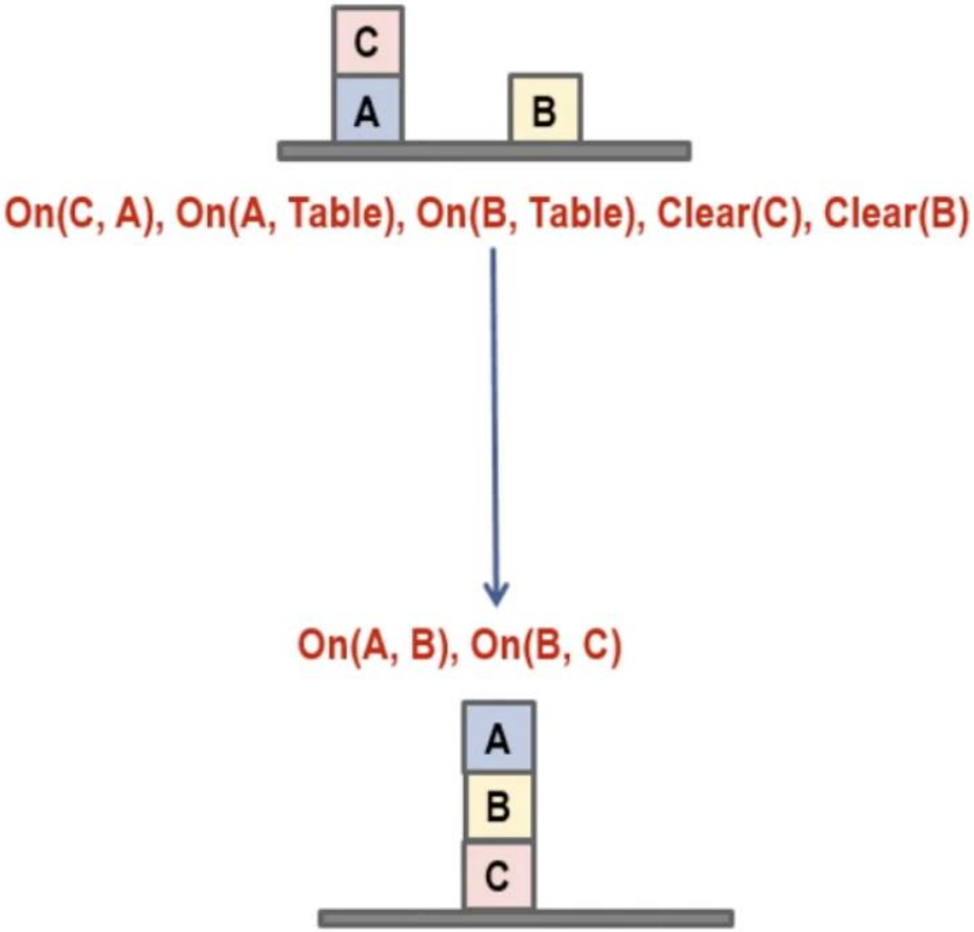
Move(X,Y): Move X on top of Y
Precond: Clear(X), Clear(Y)
Effect: On(X,Y)

Move(X,Table): Move X to Table
Precond: Clear(X)
Effect: On(X,Table)

The planning task is to determine the actions for reaching the target state from the initial state

Choosing Actions

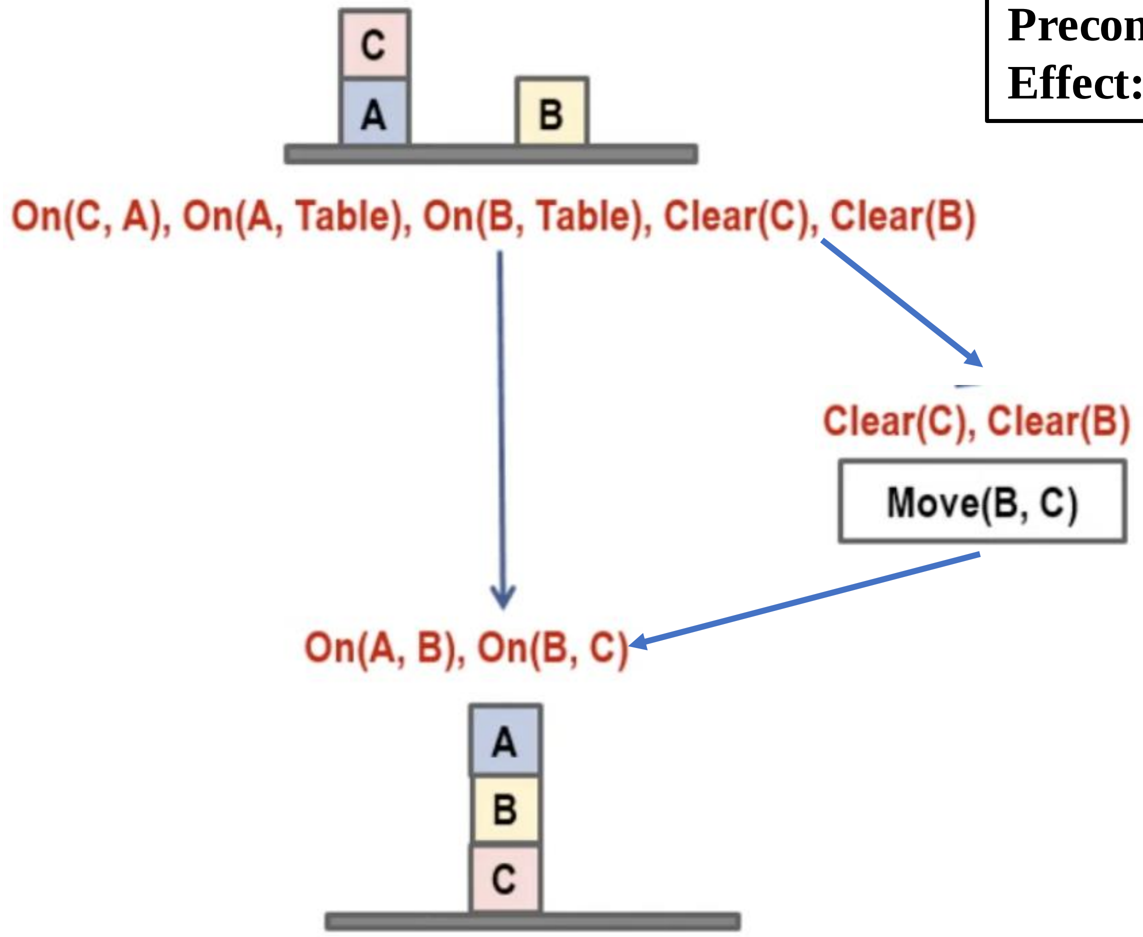
Move(X,Y): Move X on top of Y **Precond:** Clear(X), Clear(Y) **Effect:** On(X,Y)	Move(X,Table): Move X to Table **Precond:** Clear(X) **Effect:** On(X,Table)



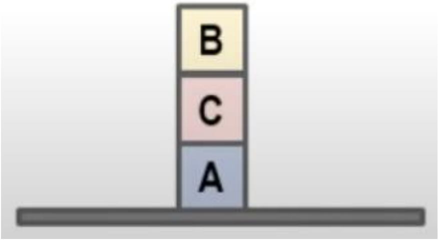
- We can move C to the Table
  - This achieves none of the goal predicates
- We can move C to the top of B
  - This achieves none of the goal predicates
- We can move B to the top of C
  - This achieves On(B,C)

Partial Solutions

Move(X,Y): Move X on top of Y	Move(X,Table): Move X to Table
Precond: Clear(X), Clear(Y)	**Precond:** Clear(X)
Effect: On(X,Y)	**Effect:** On(X,Table)

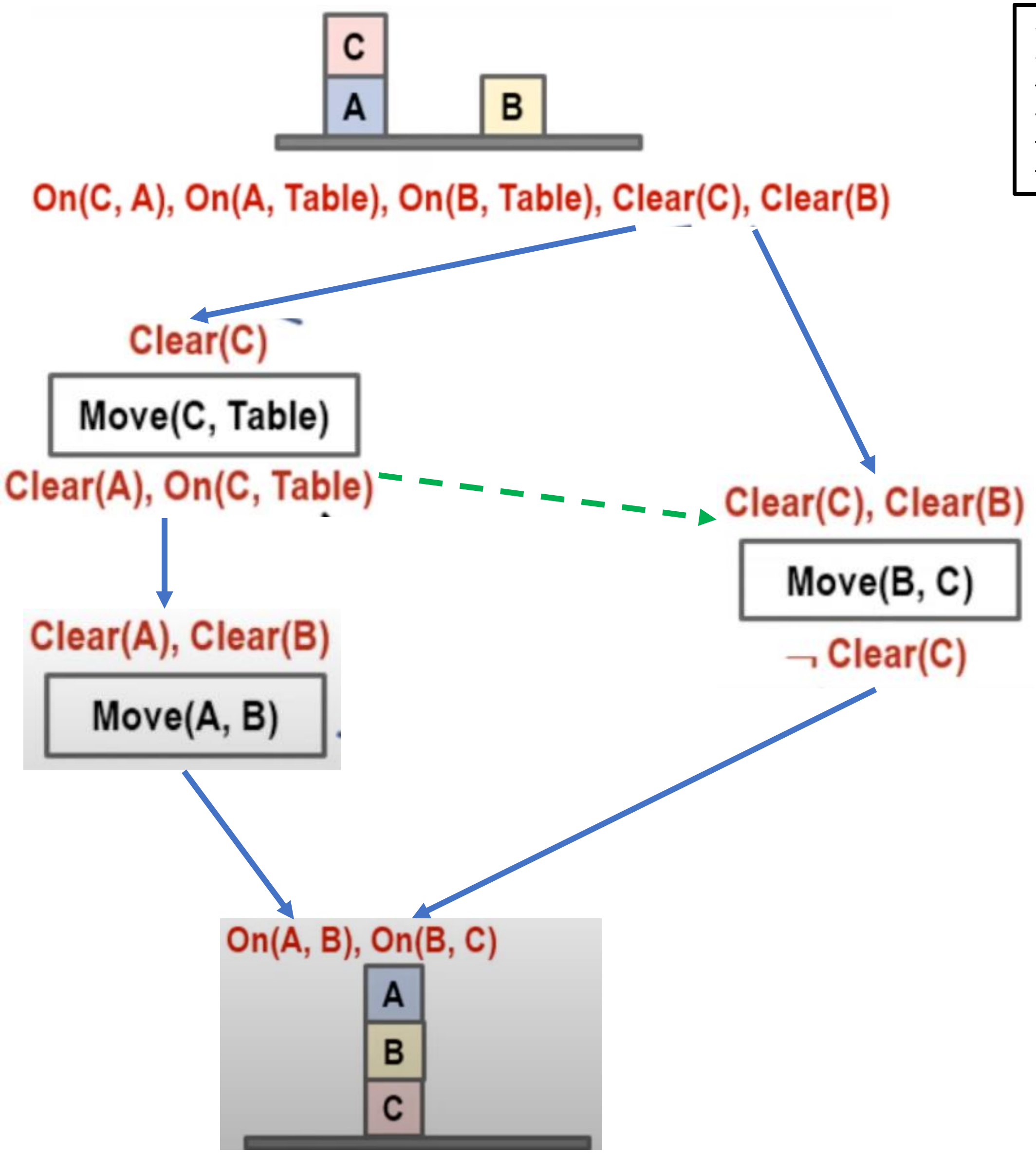


- We use Move(B,C) to achieve the subgoal On(B,C)
- But if we apply this move at the beginning, we get:



- We do not want

Partial Solutions

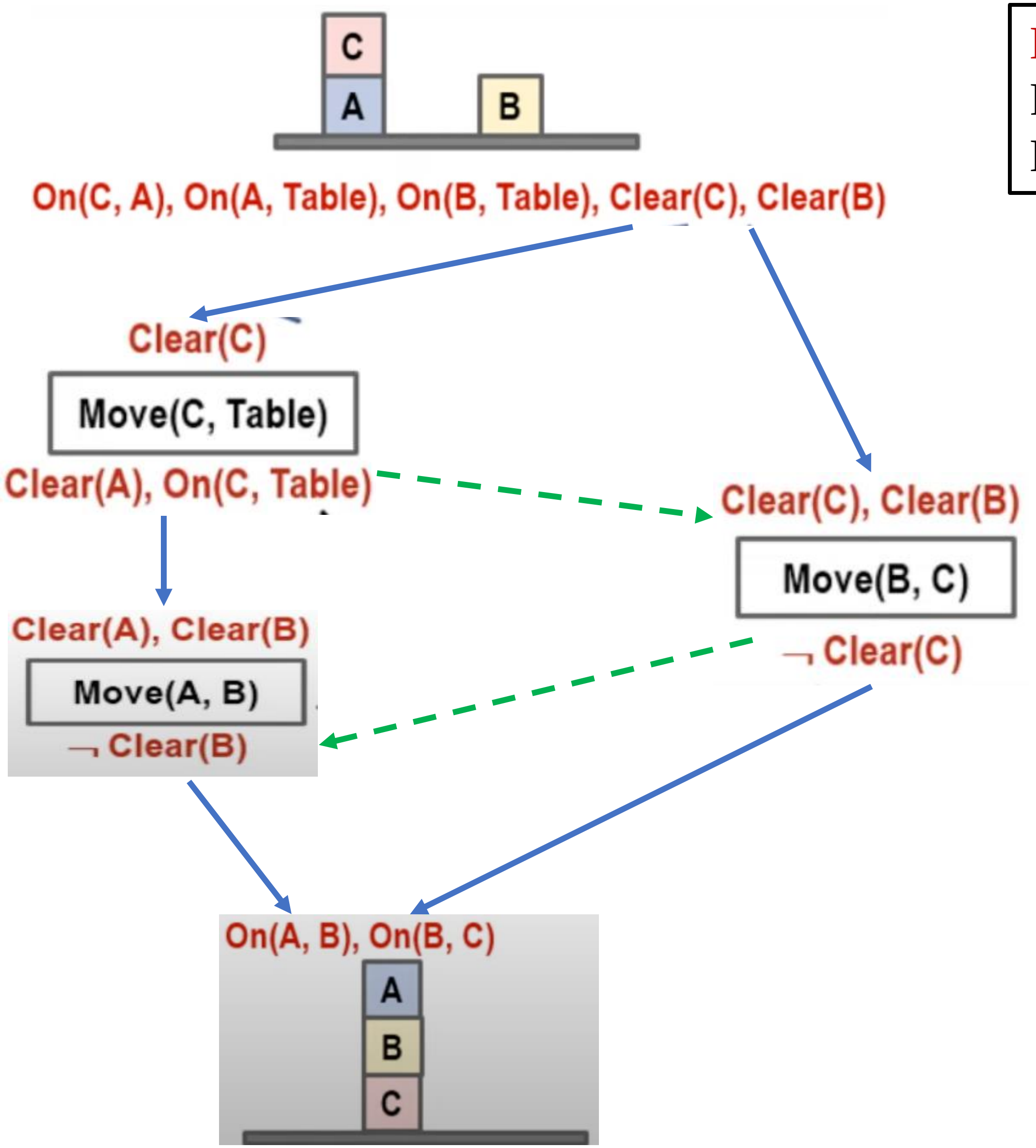


Move(X,Y): Move X on top of Y **Precond:** Clear(X), Clear(Y) **Effect:** On(X,Y)	Move(X,Table): Move X to Table **Precond:** Clear(X) **Effect:** On(X,Table)

- Move(B,C) removes the Clear(C) predicate which is essential for Move(C, Table)
- Hence Move(C, Table) must precede Move(B,C)

- Can Move(B,C) and Move(A,B) be executed in any order?

Partial Solutions



Move(X,Y): Move X on top of Y Precond: Clear(X), Clear(Y) Effect: On(X,Y)	Move(X,Table): Move X to Table Precond: Clear(X) Effect: On(X,Table)

- Move(B,C) removes the Clear(C) predicate which is essential for Move(C, Table)
- Hence Move(C, Table) must precede Move(B,C)

How to achieve each sub-goals?
Which actions to choose?
How to serialize the actions so that precedence constraints get satisfied?

The only total order is:

- Move(C, Table)
- Move(B, C)
- Move(A, B)

Do we always need total ordering?

Some partial orders may stay

- Actions

Op(ACTION: RightShoe,
PRECOND: RightSockOn,
EFFECT: RightShoeOn)

Op(ACTION: RightSock,
EFFECT: RightSockOn)

Op(ACTION: LeftShoe,
PRECOND: LeftSockOn,
EFFECT: LeftShoeOn)

Op(ACTION: LeftSock,
EFFECT: LeftSockOn)

Which of these situations are allowed by these actions?

Some partial orders may stay

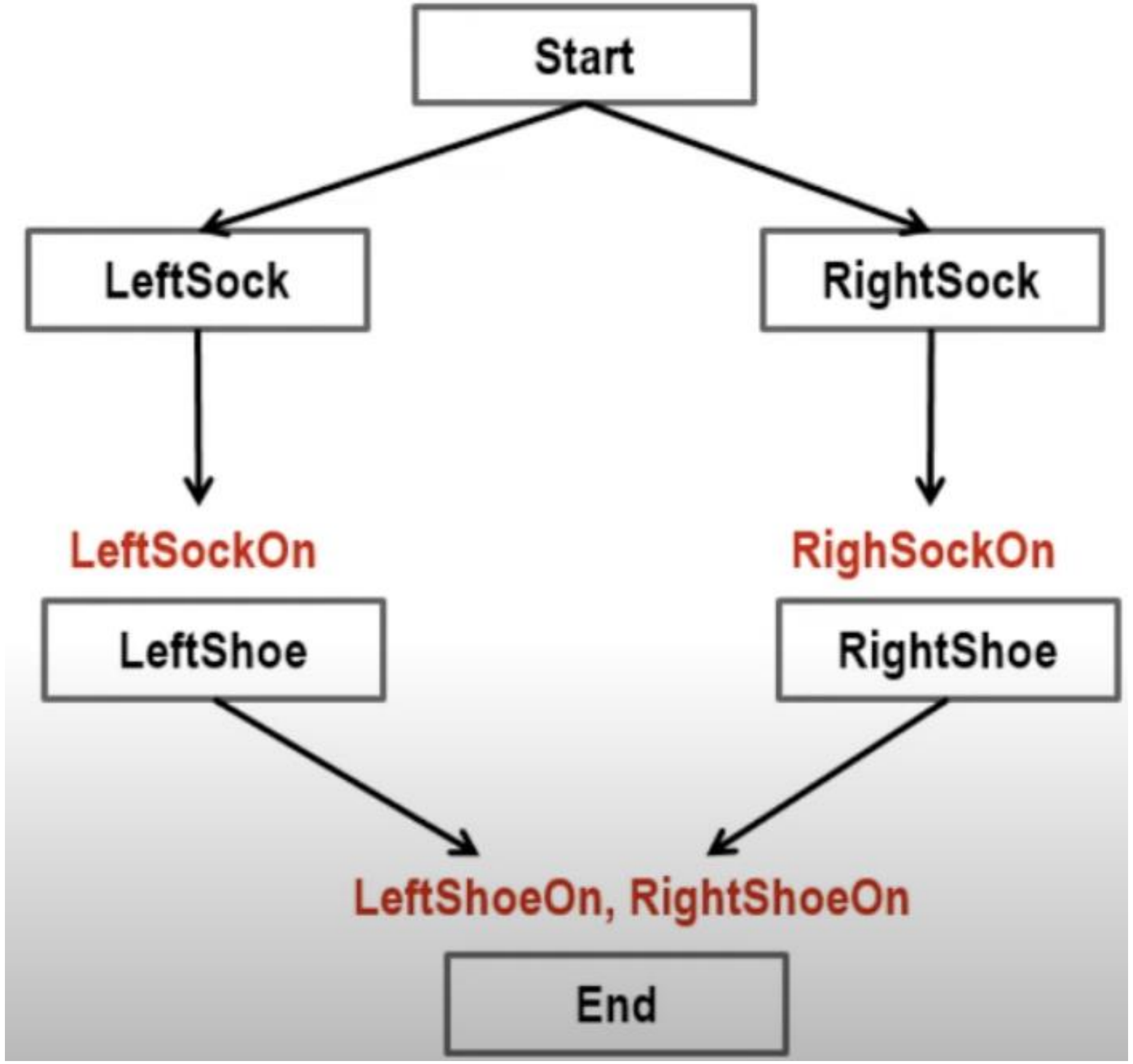
- Actions

Op(ACTION: RightShoe,
PRECOND: RightSockOn,
EFFECT: RightShoeOn)

Op(ACTION: RightSock,
EFFECT: RightSockOn)

Op(ACTION: LeftShoe,
PRECOND: LeftSockOn,
EFFECT: LeftShoeOn)

Op(ACTION: LeftSock,
EFFECT: LeftSockOn)



Planning: Automation

- Partial order planning
- GraphPlan
- SATPlan
- Stochastic Planning

Partial Order Planning

- Basic Idea: Make choices only that are relevant to solving the current part of the problem

- Least Commitment Choices:
  - Orderings: Leave actions unordered, unless they must be sequential
  - Bindings: Leave variables unbound, unless needed to unify with conditions being achieved
  - Actions: usually not subject to “least commitment”

Terminology

- Totally Ordered Plan
  - There exists sufficient orderings O such that all actions in A are ordered with respect to each other
- Fully Instantiated Plan
  - There exist sufficient constraints in B such that all variables are constrained to be equal to some constant
- Consistent Plan
  - There are no contradictions in O or B
- Complete Plan
  - Every precondition P of every action Ai in A is achieved:
    - There exists an effect of an action A_j that comes before A_i and unifies with P, and no action A_k that deletes P comes between A_j and A_i

STRIPS

- Stanford Research Institute Problem Solver
- Many planners today use specification languages that are variants of the one used in STRIPS

- Our running example:
  - Given:
    - Initial State: The agent is at home without tea, biscuits, book
    - Goal State: The agent is at home with tea, biscuits, book
    - A set of actions

State Representation

- States are represented by conjunctions of function-free ground literals
  - $At(Home) \wedge \sim Have(Tea) \wedge \sim Have(Biscuits) \wedge \sim Have(Book)$

- Goals are also described by conjunction of literals
  - $At(Home) \wedge Have(Tea) \wedge Have(Biscuits) \wedge Have(Book)$

- Goals can also contain variables
  - $At(x) \wedge Sells(x, Tea)$
  - The above goal is being at a shop that sells tea

Representing Actions

- Action description: serves as a name
- Precondition: a conjunction of positive literals
- Effect: a conjunction of literals (+ve or –ve)

- OP(
    - ACTION: $Go(there)$
    - PRECOND: $At(here) \wedge Path(here, there)$
    - EFFECT: $At(there) \wedge \sim At(here)$
    -)

Representing Plans

- A set of plan steps
  - Each step is one of the operators for the problem

- A set of step ordering constraints
  - Each ordering constraint is of the form $S_i \prec S_j$
  - indicating S_i must occur sometime before S_j

- A set of variable binding constraints of the form v=x
  - v is a variable in some step
  - x is either a constant or another variable

- A set of causal links written as $S \rightarrow c: S'$ indicating S satisfies the precondition c for S'

Example

- Initial Plan
- Plan(
  - STEPS: {
    - S1: Op(ACTION: start),
    - S2: Op(ACTION: finish, PRECOND: RightShoeOn ∧ LeftShoeOn)
    - },
  - ORDERINGS: $\{S_1 \prec S_2\}$,
  - BINDINGS: {},
  - LINKS: {}
  -)

POP Example: Get Tea, Biscuits, Book

Initial State:

Op(ACTION: Start,
  EFFECT: At(Home) ∧ Sells(BS, Book)
    ∧ Sells(TS, Tea)
    ∧ Sells(TS, Biscuits))

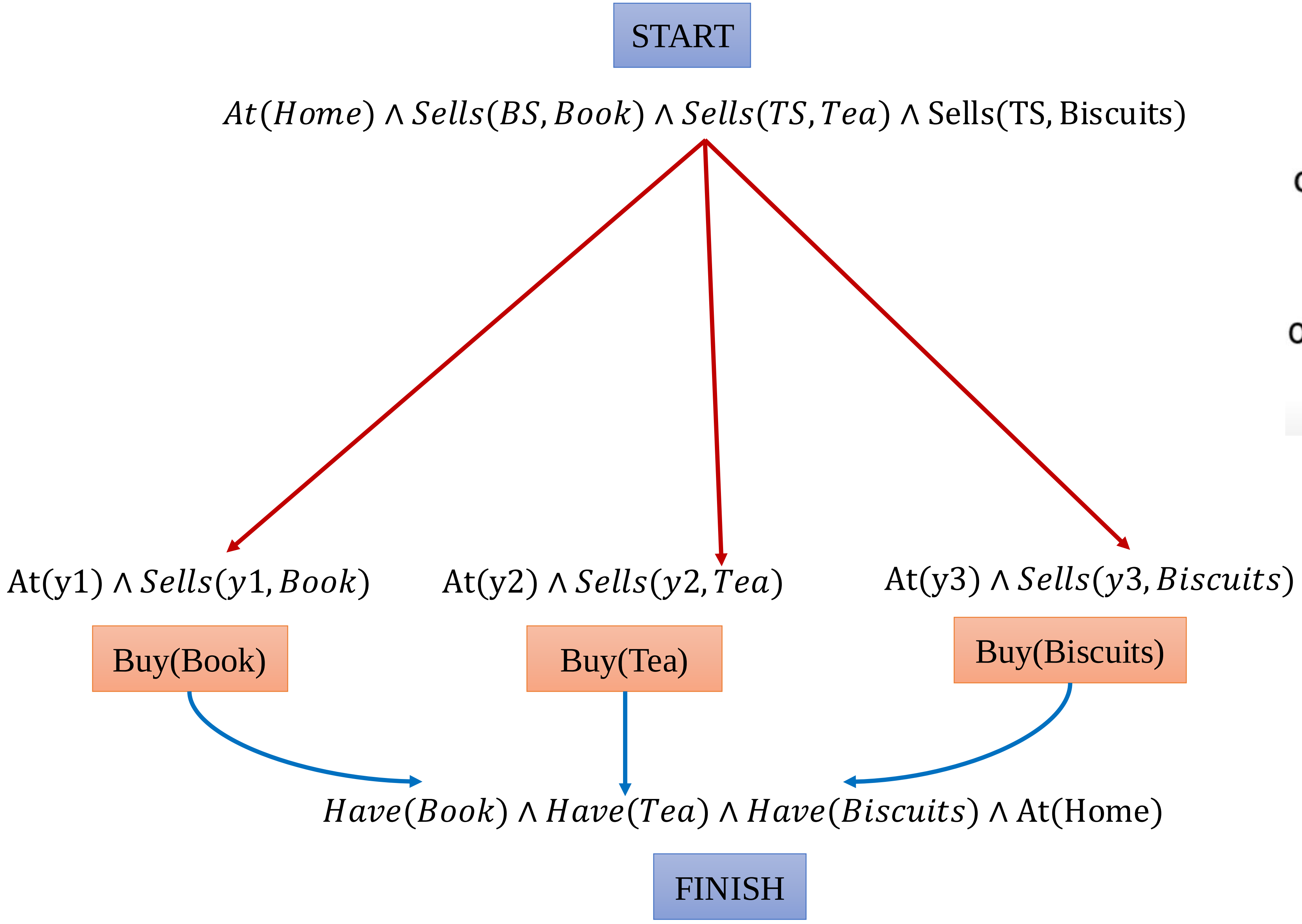
Goal State:

Op(ACTION: Finish,
  PRECOND: At(Home) ∧ Have(Tea)
    ∧ Have(Biscuits)
    ∧ Have(Book))

Actions:

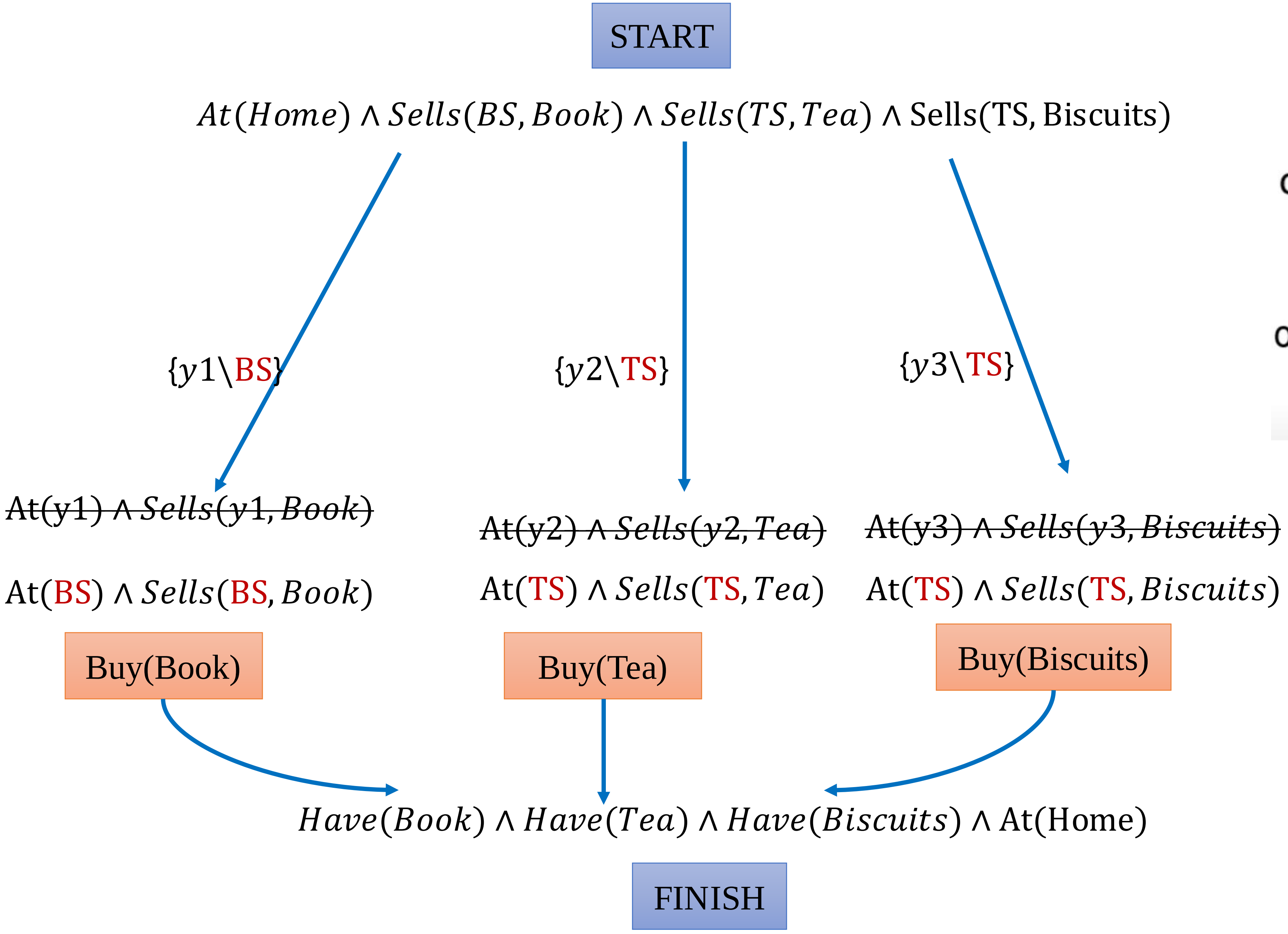
Op(ACTION: Go(y),
  PRECOND: At(x),
  EFFECT: At(y) ∧ ¬At(x))

Op(ACTION: Buy(x),
  PRECOND: At(y) ∧ Sells(y, x),
  EFFECT: Have(x))



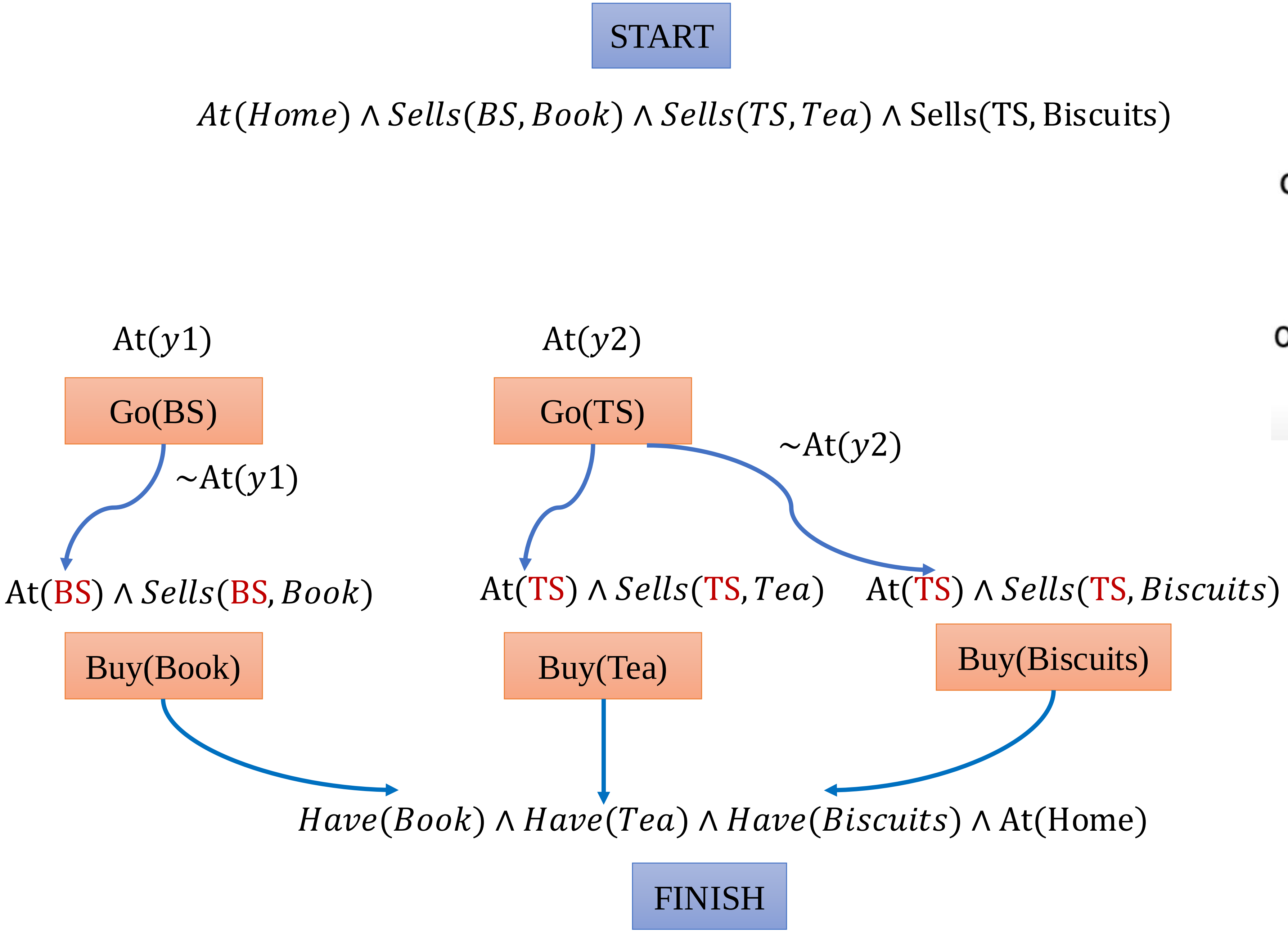
Op(ACTION: Go(y),
PRECOND: At(x),
EFFECT: At(y) $\wedge \neg$At(x))

Op(ACTION: Buy(x),
PRECOND: At(y) $\wedge$ Sells(y, x),
EFFECT: Have(x))



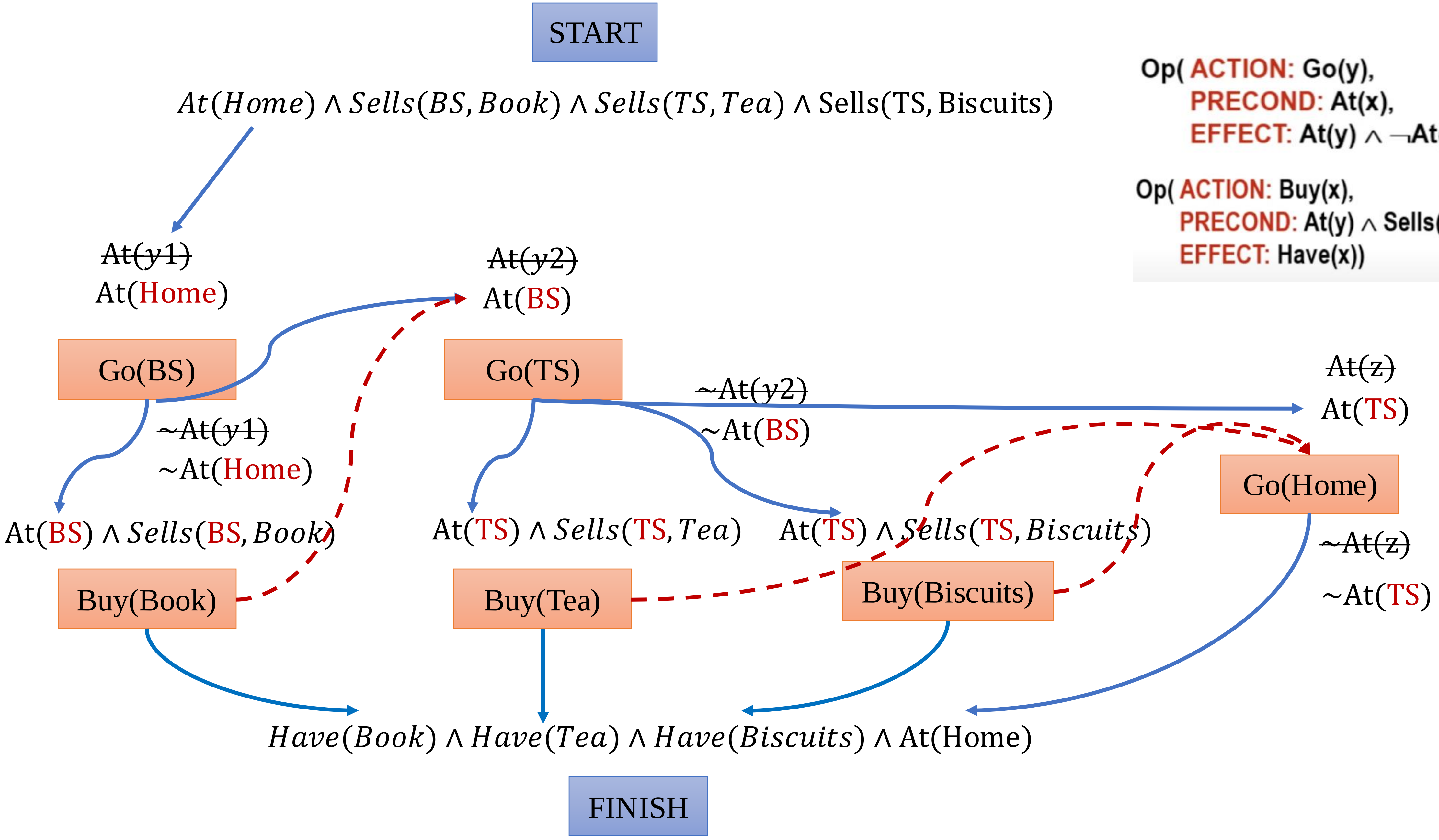
Op(ACTION: Go(y),
PRECOND: At(x),
EFFECT: At(y) ∧ ¬At(x))

Op(ACTION: Buy(x),
PRECOND: At(y) ∧ Sells(y, x),
EFFECT: Have(x))



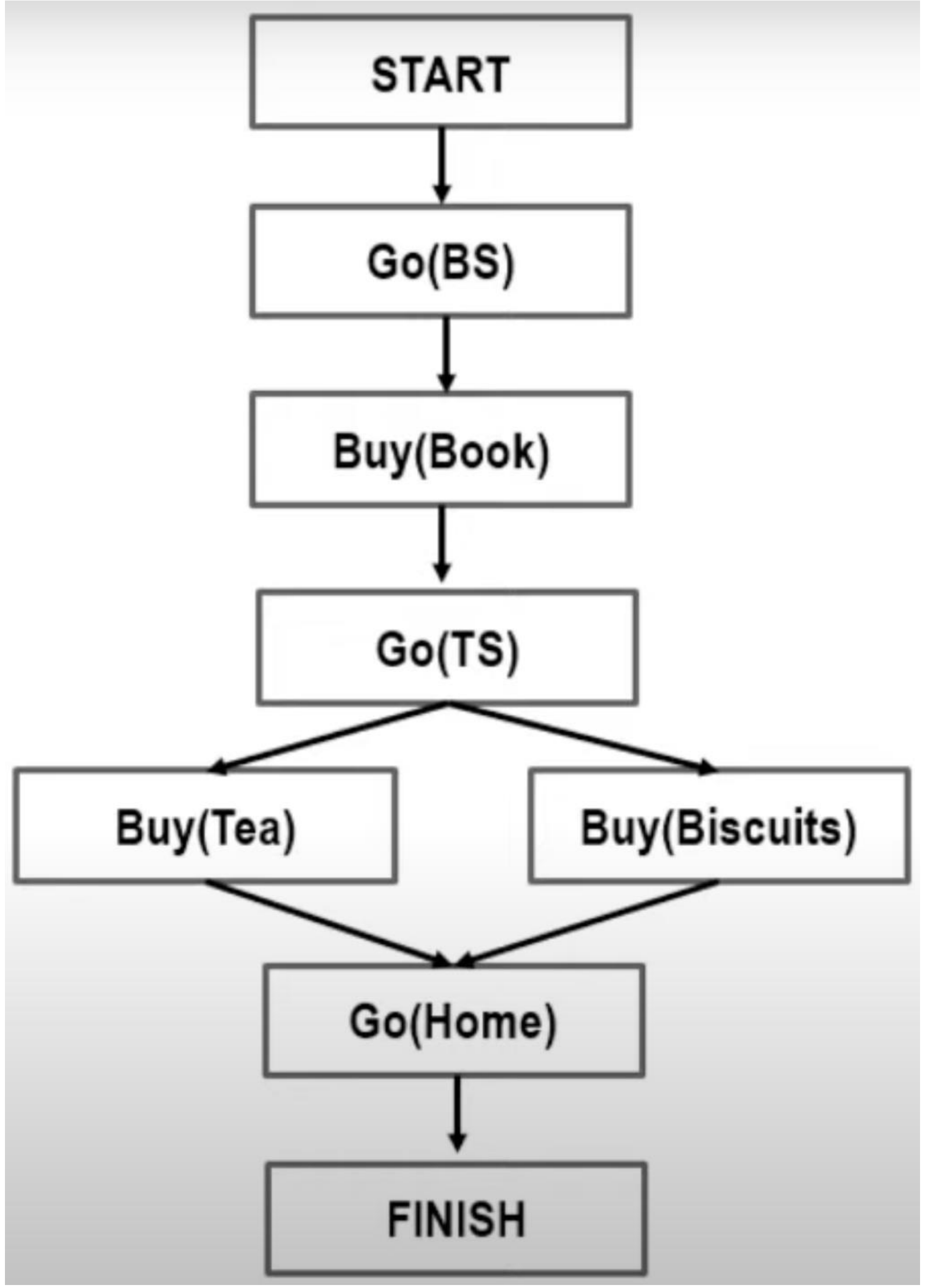
Op(ACTION: Go(y),
PRECOND: At(x),
EFFECT: $At(y) \wedge \neg At(x)$)

Op(ACTION: Buy(x),
PRECOND: $At(y) \wedge Sells(y, x)$,
EFFECT: Have(x))



Op(ACTION: Go(y),
PRECOND: At(x),
EFFECT: At(y) $\wedge$ $\neg$At(x))

Op(ACTION: Buy(x),
PRECOND: At(y) $\wedge$ Sells(y, x),
EFFECT: Have(x))



- Ordering
- Causal Constraints

Thank You