MODULE-III

SYLLABUS

Intelligent Agents and Planning: Structure of Intelligent Agents and Environments. Simple Planning Agent, From Problem Solving to Planning, Planning in Situation Calculus, Basic Representations for Planning, A Partial-Order Planning Algorithm and examples.

What is an Agent?

An agent can be anything that perceive its environment through sensors and act upon that environment through actuators. An Agent runs in the cycle of **perceiving**, **thinking**, and **acting**. An agent can be:

- **Human-Agent:** A human agent has eyes, ears, and other organs which work for sensors and hand, legs, vocal tract work for actuators.
- **Robotic Agent:** A robotic agent can have cameras, infrared range finder, NLP for sensors and various motors for actuators.
- **Software Agent:** Software agent can have keystrokes, file contents as sensory input and act on those inputs and display output on the screen.

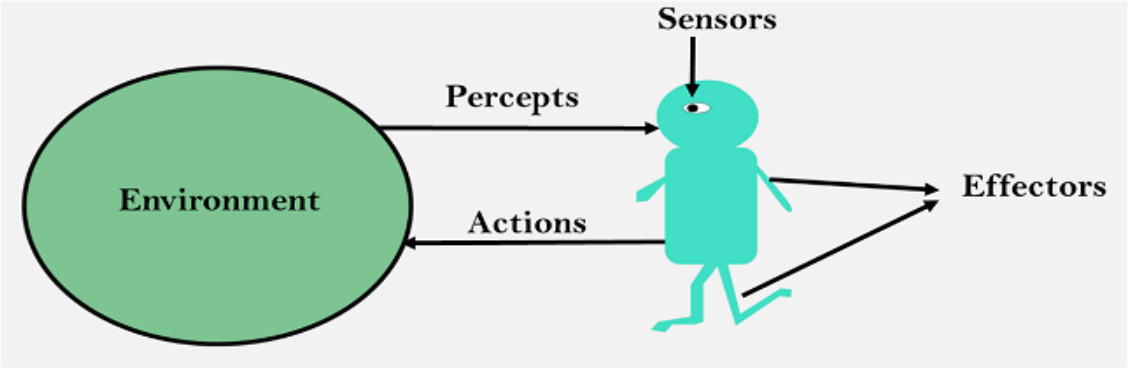
Hence the world around us is full of agents such as thermostat, cellphone, camera, and even we are also agents.

Before moving forward, we should first know about sensors, effectors, and actuators.

Sensor: Sensor is a device which detects the change in the environment and sends the information to other electronic devices. An agent observes its environment through sensors.

Actuators: Actuators are the component of machines that converts energy into motion. The actuators are only responsible for moving and controlling a system. An actuator can be an electric motor, gears, rails, etc.

Effectors: Effectors are the devices which affect the environment. Effectors can be legs, wheels, arms, fingers, wings, fins, and display screen.



Intelligent Agents:

An intelligent agent is an autonomous entity which acts upon an environment using sensors and actuators for achieving goals. An intelligent agent may learn from the environment to achieve their goals. A thermostat is an example of an intelligent agent.

Following are the main four rules for an AI agent:

- **Rule 1:** An AI agent must have the ability to perceive the environment.
- **Rule 2:** The observation must be used to make decisions.
- **Rule 3:** Decision should result in an action.
- **Rule 4:** The action taken by an AI agent must be a rational action.

Rational Agent:

A rational agent is an agent which has clear preference, models uncertainty, and acts in a way to maximize its performance measure with all possible actions.

A rational agent is said to perform the right things. AI is about creating rational agents to use for game theory and decision theory for various real-world scenarios.

For an AI agent, the rational action is most important because in AI reinforcement learning algorithm, for each best possible action, agent gets the positive reward and for each wrong action, an agent gets a negative reward.

Rationality:

The rationality of an agent is measured by its performance measure. Rationality can be judged on the basis of following points:

- Performance measure which defines the success criterion.
- Agent prior knowledge of its environment.
- Best possible actions that an agent can perform.
- The sequence of percepts.

Structure of an AI Agent

The task of AI is to design an agent program which implements the agent function. The structure of an intelligent agent is a combination of architecture and agent program. It can be viewed as:

Agent = Architecture + Agent program

Following are the main three terms involved in the structure of an AI agent:

1. **Architecture:** Architecture is machinery that an AI agent executes on.
2. **Agent Function:** Agent function is used to map a percept to an action.

$$\mathbf{f: P^* \rightarrow A}$$

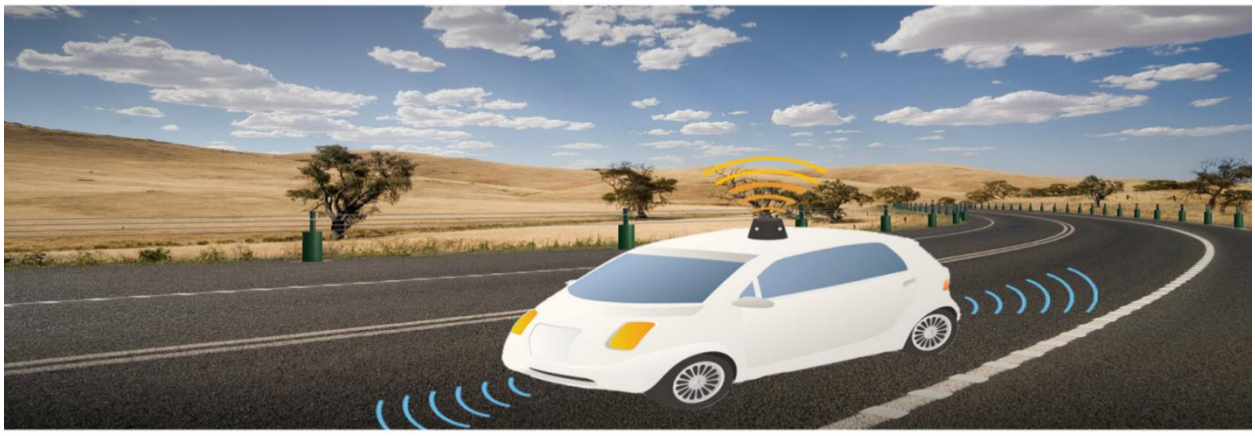
Agent program: Agent program is an implementation of agent function. An agent program executes on the physical architecture to produce function f.

<u>PEAS Representation</u>

PEAS is a type of model on which an AI agent works upon. When we define an AI agent or rational agent, then we can group its properties under PEAS representation model. It is made up of four words:

- **P:** Performance measure
- **E:** Environment
- **A:** Actuators
- **S:** Sensors

Here performance measure is the objective for the success of an agent's behavior.



Let's suppose a self-driving car then PEAS representation will be:

Performance: Safety, time, legal drive, comfort

Environment: Roads, other vehicles, road signs, pedestrian

Actuators: Steering, accelerator, brake, signal, horn

Sensors: Camera, GPS, speedometer, odometer, accelerometer, sonar.

Example of Agents with their PEAS representation

Agent	Performance measure	Environment	Actuators	Sensors
1. Medical Diagnose	o Healthy patient o Minimized cost	o Patient o Hospital o Staff	o Tests o Treatments	Keyboard (Entry of symptoms)
2. Vacuum Cleaner	o Cleanness o Efficiency o Battery life o Security	o Room o Table o Wood floor o Carpet o Various obstacles	o Wheels o Brushes o Vacuum Extractor	o Camera o Dirt detection sensor o Cliff sensor o Bump Sensor o Infrared Wall Sensor
3. Part - picking Robot	o Percentage of parts in correct bins.	o Conveyor belt with parts, o Bins	o Jointed Arms o Hand	o Camera o Joint angle sensors.

Agent Environment in AI

An environment is everything in the world which surrounds the agent, but it is not a part of an agent itself. An environment can be described as a situation in which an agent is present.

The environment is where agent lives, operate and provide the agent with something to sense and act upon it. An environment is mostly said to be non-feministic.

Features of Environment

As per Russell and Norvig, an environment can have various features from the point of view of an agent:

1. Fully observable vs Partially Observable
2. Static vs Dynamic
3. Discrete vs Continuous
4. Deterministic vs Stochastic
5. Single-agent vs Multi-agent
6. Episodic vs sequential
7. Known vs Unknown
8. Accessible vs Inaccessible

1. Fully observable vs Partially Observable:

- If an agent sensor can sense or access the complete state of an environment at each point of time then it is **a fully observable** environment, else it is **partially observable**.
- A fully observable environment is easy as there is no need to maintain the internal state to keep track history of the world.
- An agent with no sensors in all environments then such an environment is called as **unobservable**.

2. Deterministic vs Stochastic:

- If an agent's current state and selected action can completely determine the next state of the environment, then such environment is called a deterministic environment.
- A stochastic environment is random in nature and cannot be determined completely by an agent.
- In a deterministic, fully observable environment, agent does not need to worry about uncertainty.

3. Episodic vs Sequential:

- In an episodic environment, there is a series of one-shot actions, and only the current percept is required for the action.
- However, in Sequential environment, an agent requires memory of past actions to determine the next best actions.

4. Single-agent vs Multi-agent

- If only one agent is involved in an environment, and operating by itself then such an environment is called single agent environment.
- However, if multiple agents are operating in an environment, then such an environment is called a multi-agent environment.
- The agent design problems in the multi-agent environment are different from single agent environment.

5. Static vs Dynamic:
   - If the environment can change itself while an agent is deliberating then such environment is called a dynamic environment else it is called a static environment.
   - Static environments are easy to deal because an agent does not need to continue looking at the world while deciding for an action.
   - However for dynamic environment, agents need to keep looking at the world at each action.
   - Taxi driving is an example of a dynamic environment whereas Crossword puzzles are an example of a static environment.
6. Discrete vs Continuous:
   - If in an environment there are a finite number of percepts and actions that can be performed within it, then such an environment is called a discrete environment else it is called continuous environment.
   - A chess gamecomes under discrete environment as there is a finite number of moves that can be performed.
   - A self-driving car is an example of a continuous environment.
7. Known vs Unknown
   - Known and unknown are not actually a feature of an environment, but it is an agent's state of knowledge to perform an action.
   - In a known environment, the results for all actions are known to the agent. While in unknown environment, agent needs to learn how it works in order to perform an action.
   - It is quite possible that a known environment to be partially observable and an Unknown environment to be fully observable.
8. Accessible vs Inaccessible
   - If an agent can obtain complete and accurate information about the state's environment, then such an environment is called an Accessible environment else it is called inaccessible.
   - An empty room whose state can be defined by its temperature is an example of an accessible environment.
   - Information about an event on earth is an example of Inaccessible environment.

Planning:

Artificial intelligence is an important technology in the future. Whether it is intelligent robots, self-driving cars, or smart cities, they will all use different aspects of artificial intelligence!!! But Planning is very important to make any such AI project.

Even Planning is an important part of Artificial Intelligence which deals with the tasks and domains of a particular problem. Planning is considered the logical side of acting.

Everything we humans do is with a definite goal in mind, and all our actions are oriented towards achieving our goal. Similarly, Planning is also done for Artificial Intelligence.

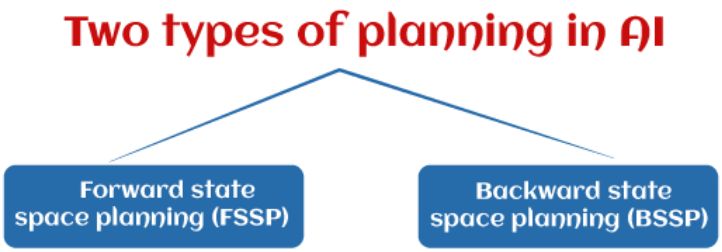
For example, Planning is required to reach a particular destination. It is necessary to find the best route in Planning, but the tasks to be done at a particular time and why they are done are also very important.

That is why Planning is considered the logical side of acting. In other words, Planning is about deciding the tasks to be performed by the artificial intelligence system and the system's functioning under domain-independent conditions.

What is a Plan?

We require domain description, task specification, and goal description for any planning system. A plan is considered a sequence of actions, and each action has its preconditions that must be satisfied before it can act and some effects that can be positive or negative.

So, we have **Forward State Space Planning (FSSP)** and **Backward State Space Planning (BSSP)** at the basic level.



1. Forward State Space Planning (FSSP)

FSSP behaves in the same way as forwarding state-space search. It says that given an initial state S in any domain, we perform some necessary actions and obtain a new state S' (which also contains some new terms), called a progression. It continues until we reach the target position. Action should be taken in this matter.

- **Disadvantage**: Large branching factor
- **Advantage**: The algorithm is Sound

2. Backward State Space Planning (BSSP)

BSSP behaves similarly to backward state-space search. In this, we move from the target state g to the sub-goal g, tracing the previous action to achieve that goal. This process is called regression (going back to the previous goal or sub-goal). These sub-goals should also be checked for consistency. The action should be relevant in this case.

- **Disadvantages**: not sound algorithm (sometimes inconsistency can be found)
- **Advantage**: Small branching factor (much smaller than FSSP)

So for an efficient planning system, we need to combine the features of FSSP and BSSP, which gives rise to target stack planning which will be discussed in the next article.

What is planning in AI?

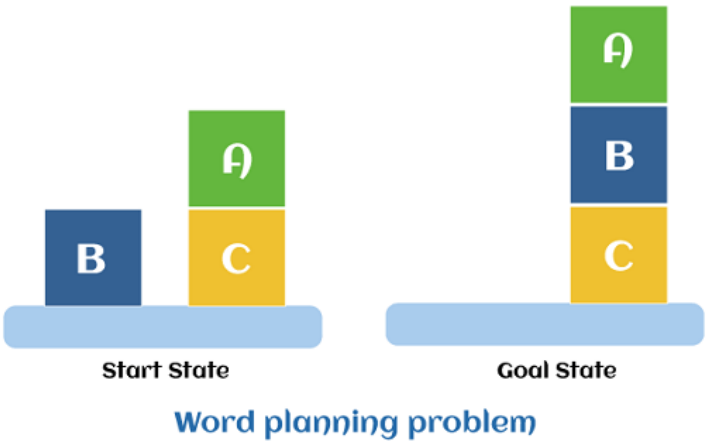
Planning in artificial intelligence is about decision-making actions performed by robots or computer programs to achieve a specific goal.

Execution of the plan is about choosing a sequence of tasks with a high probability of accomplishing a specific task.

Block-world planning problem

- The block-world problem is known as the Sussmann anomaly.
- The non-interlaced planners of the early 1970s were unable to solve this problem. Therefore it is considered odd.
- When two sub-goals, G1 and G2, are given, a non-interleaved planner either produces a plan for G1 that is combined with a plan for **G2** or vice versa.
- In the block-world problem, three blocks labeled 'A', 'B', and 'C' are allowed to rest on a flat surface. The given condition is that only one block can be moved at a time to achieve the target.

The start position and target position are shown in the following diagram.



Components of the planning system

The plan includes the following important steps:

- Choose the best rule to apply the next rule based on the best available guess.
- Apply the chosen rule to calculate the new problem condition.
- Find out when a solution has been found.
- Detect dead ends so they can be discarded and direct system effort in more useful directions.
- Find out when a near-perfect solution is found.

Target stack plan

- It is one of the most important planning algorithms used by STRIPS.
- Stacks are used in algorithms to capture the action and complete the target. A knowledge base is used to hold the current situation and actions.
- A target stack is similar to a node in a search tree, where branches are created with a choice of action.

The important steps of the algorithm are mentioned below:

1. Start by pushing the original target onto the stack. Repeat this until the pile is empty. If the stack top is a mixed target, push its unsatisfied sub-targets onto the stack.
2. If the stack top is a single unsatisfied target, replace it with action and push the action precondition to the stack to satisfy the condition.

iii. If the stack top is an action, pop it off the stack, execute it and replace the knowledge base with the action's effect.

If the stack top is a satisfactory target, pop it off the stack.

Non-linear Planning

This Planning is used to set a goal stack and is included in the search space of all possible sub-goal orderings. It handles the goal interactions by the interleaving method.

Advantages of non-Linear Planning

Non-linear Planning may be an optimal solution concerning planning length (depending on the search strategy used).

Disadvantages of Nonlinear Planning

It takes a larger search space since all possible goal orderings are considered.

Complex algorithm to understand.

Algorithm

1. Choose a goal 'g' from the goal set
2. If 'g' does not match the state, then
    - Choose an operator 'o' whose add-list matches goal g
    - Push 'o' on the OpStack
    - Add the preconditions of 'o' to the goal set
3. While all preconditions of the operator on top of OpenStack are met in a state
    - Pop operator o from top of opstack
    - state = apply(o, state)
    - plan = [plan; o]

Partial-Order Planning

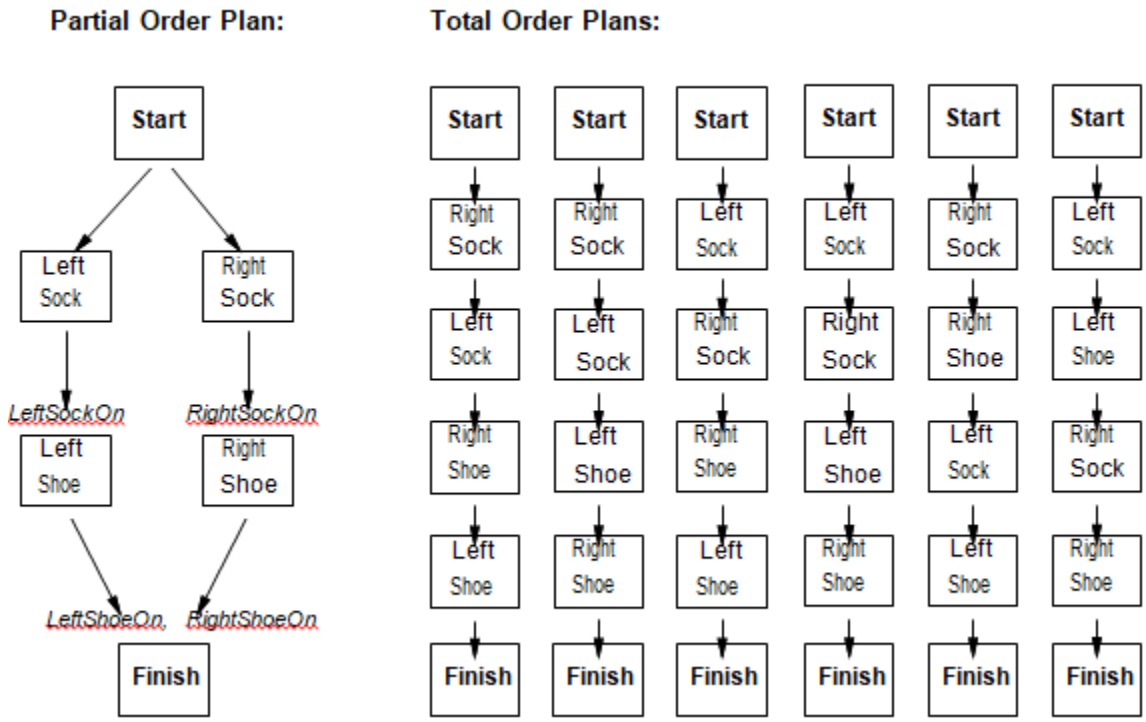
State-Space vs. Plan-Space

- **State-space** (**situation space**) planning algorithms searchthrough the space of possible states of the world searchingfor a path that solves the problem.
- They can be based on **progression**: a forward search fromthe initial state looking for the goal state.
- Or they can be based on **regression**: a backward searchfrom the goals towards the initial state
- STRIPS is an incomplete regression-based algorithm.

- **Plan-space** planners search through the space of partialplans, which are sets of actions that may not be totally ordered.
- **Partial-order** planners are plan-based and only introduce ordering constraints as necessary (**least committment**) inorder to avoid unecessarily searching through the space ofpossible orderings.

Partial Order Plans

- Plan in which not all actions are ordered



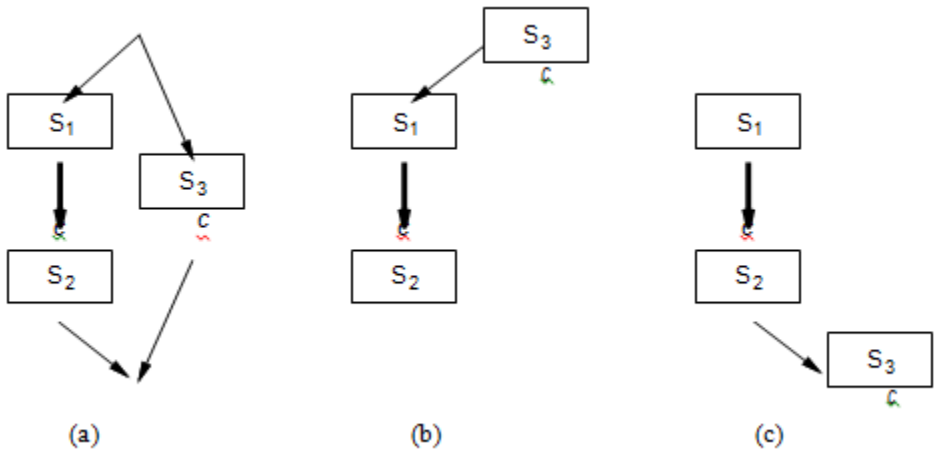
Plans, Causal Links, and Threats

- A *plan* is a three tuple <A, O, L>
  - -A: A set of actions in the plan, $\{A_1, A_2, \ldots A_n\}$
  - -O: A set of ordering constraints on actions
    $\{A_i<A_j, A_k<A_l, \ldots A_m<A_n\}$. These must be *consistent*, i.e.there must be at least one total ordering of actions in A that satisfy all the constraints.
  - -B: A set of variable binding constraints {v=x,...}
  - -L: a set of causal links showing how actions support eachother
- A *causal link*, $A_p \square^{Q} A_c$, indicates that action A_p has aneffect Q that achieves precondition Q for action A_c.
- A *threat*, is an action A_t that can render a causal link$A_p \square^{Q} A_c$ ineffective because:
  - -O □ $\{A_P < A_t < A_c\}$ is consistent
  - -A_t has □Q as an effect

Threat Removal

- Threats must be removed to prevent a plan from failing.
- **Demotion** adds the constraint $A_t < A_p$ to prevent clobbering, i.e. push the clobberer before the producer.

- **Promotion** adds the constraint $A_c < A_t$ to prevent clobbering, i.e. push the clobber after the consumer.

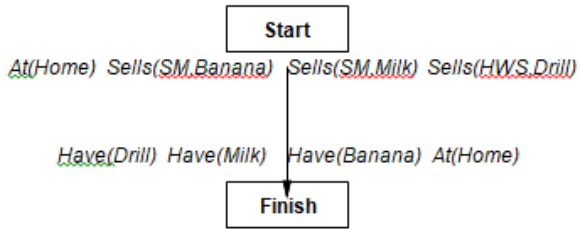


Initial (Null) Plan

- Initial plan has
  -A={ A_0, A□}
  -O={A_0< A□□
  -L ={}
- A_0(Start) has no preconditions but all facts in the initial stateas effects.
- A□ (Finish) has the goal conditions as preconditions and noeffects.

Op(Action: Go(there); Precond: At(here);Effects: At(there),
□At(here))

Op(Action: Buy(x), Precond: At(store), Sells(store,x);Effects: Have(x))



POP Algorithm

- Stated as nondeterministic algorithm where choices must be made. Various search methods can be used (e.g. breadth-first, depth-first iterative deepening etc.) to explorethe space of possible choices.
- Maintains agenda of goals that need to be supported by links, where an agenda element is a pair <Q,A_i> where Q isa precondition of A_i that needs supporting.
- Initialize plan to null plan and agenda to conjunction ofgoals (preconditions of Finish).
- Done when all preconditions of every action in plan aresupported by causal links which are not threatened.

POP(<A,O,L>, agenda)

1) **Termination: If agenda is empty, return <A,O,L>. Usetopological sort to determine a totally ordered plan.**

2) **Goal Selection:** Let $<Q,A_{need}>$ be a pair on the agenda

3) **Action Selection: Let A_{add} be a nondeterministically chosen action that adds Q. It can be an existing action in Aor a new action. If there is no such action return failure.**

$L' = L \square \{A_{add} \square^{Q} A_{need}\} O' = O \square$

$\{A_{add} < A_{need}\}$ if A_{add} is new

then $A' = A \square \{A_{add}\}$ and $O'=O' \square \{A_0 < A_{add} < A \square\}$

else $A' = A$

4) **Update goal set:** Let agenda´= agenda $\square$ $\{<Q,A_{need}>\}$ If A_{add} is new then for each conjunct Q_i of its precondition,add $<Q_i, A_{add}>$ to agenda´

5) **Causal link protection:** For every action A_t that threatens a causal link $A_p \square^{Q} A_c$ add an ordering constraintby choosing nondeterministically either

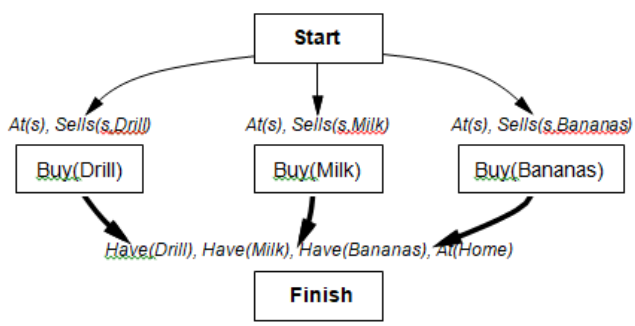
(a) **Demotion:** Add $A_t < A_p$ to O´

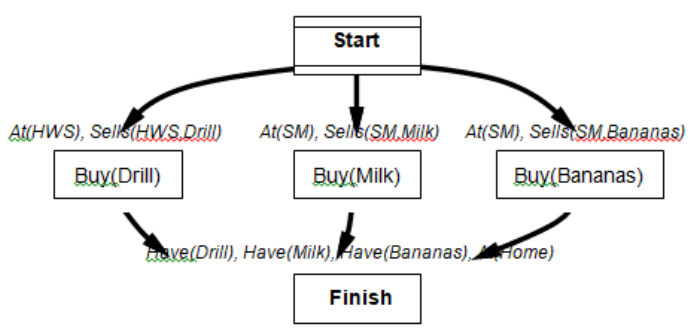
(b) **Promotion:** Add $A_c < A_t$ to O´

If neither constraint is consistent then return failure.

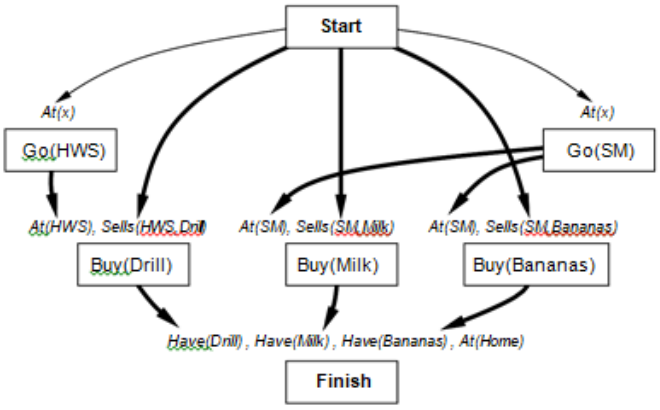
6) **Recurse:** POP(<A´,O´,L´>, agenda´)

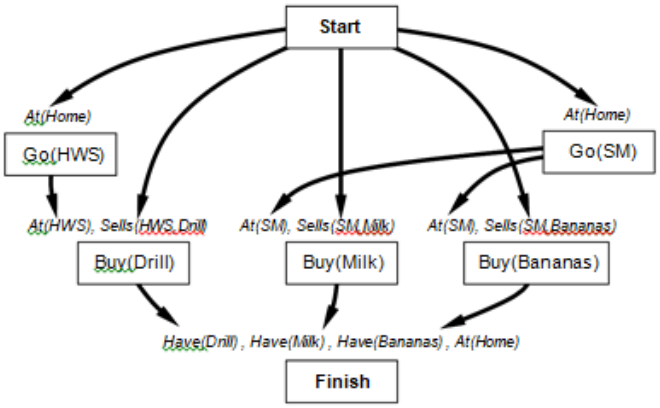
Example

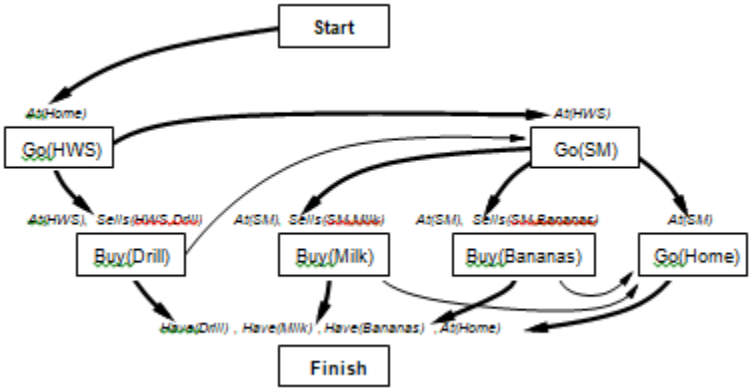




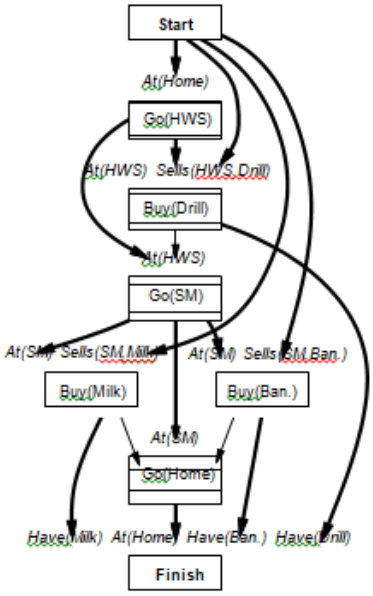
- Add three actions to achieve basic goals. Use initial state toachieve the Sells preconditions.
- Bold links are causal, regular are just ordering constraints







- Cannot resolve threat to At(Home) preconditions of bothGo(HWS) and Go(SM).
- Must backtrack to supporting At(x) precondition of Go(SM)from intial state At(Home) and support it instead from the At(HWS) effect of Go(HWS).
- Since Go(SM) still threatens At(HWS) of Buy(Drill) must promote Go(SM) to come after Buy(Drill). Demotion is notpossible due to causal link supporting At(HWS) precondition of Go(SM)



- Add Go(Home) action to achieve At(Home), use At(SM) toachieve its precondition, and order it after Buy(Milk) and Buy(Banana) to resolve threats to At(SM).

Planning Conclusions

- Experiments confirm that in most cases partial-orderplanning is more efficient than total order.
- Planning techniques have been applied to a number ofrealistic tasks:
  -Logistics planning for Desert Storm
  -Scheduling for the Hubble Space Telescope
  -Planning ground operations for the Space Shuttle
  -Semiconductor manufacturing
  -Cleaning oil spills
- Many issues are involved in interleaving planning andexecution
  -Conditional planning
  -Execution monitoring and dynamic replanning