Introduction to Artificial Intelligence

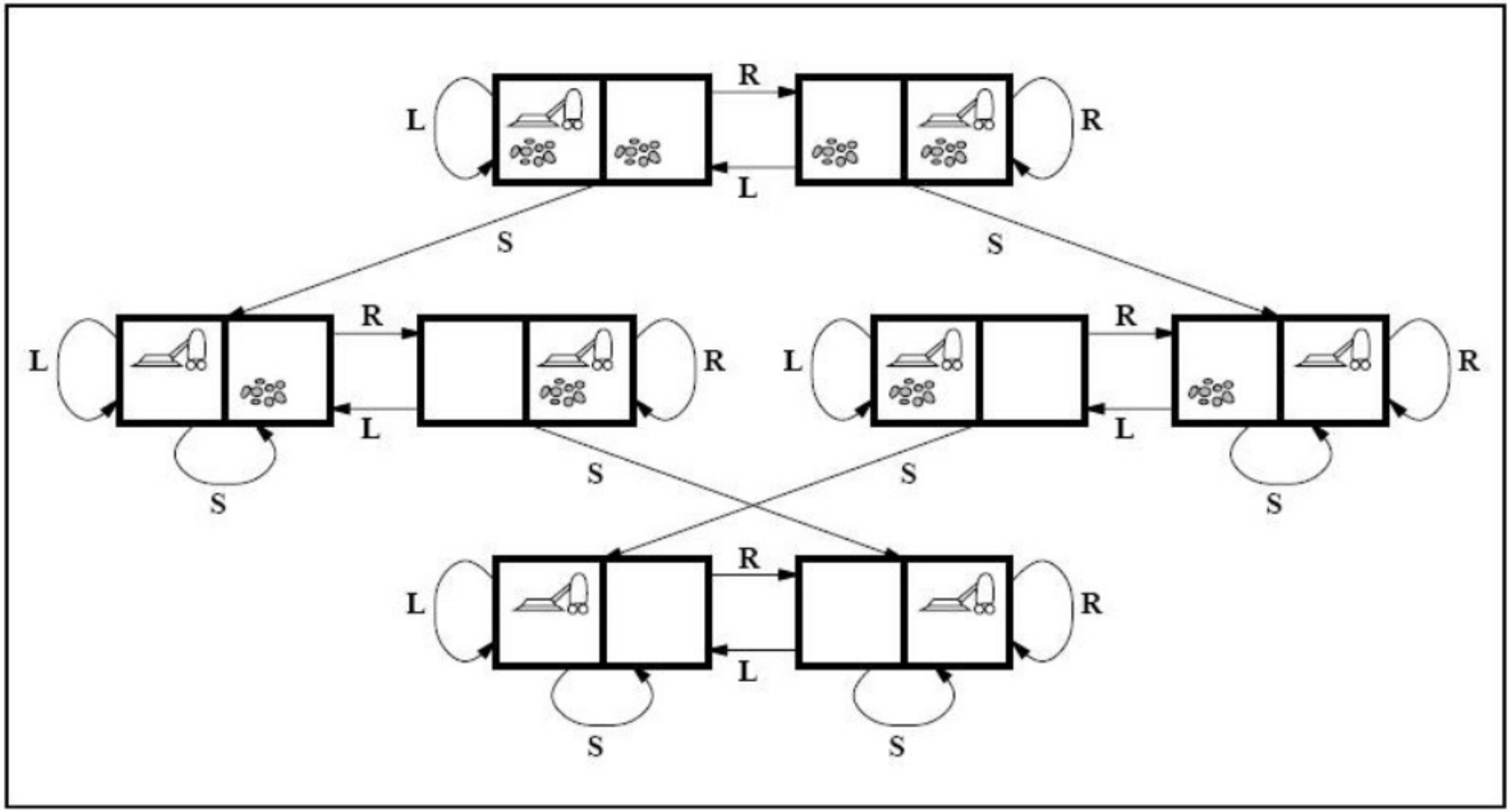
VICTORIA UNIVERSITY OF WELLINGTON
TE HERENGA WAKA

COMP307
Planning and Scheduling:
Tutorial 1

Question

- Which of states below are valid states?
  - $Painted(LeftWall)$
  - $At(x)$
  - $\neg Clean(Cotton)$
  - $Rainy(Tomorrow)\ or\ Cloudy(Tomorrow)$
  - $Passed(DueDate(307A3))$
  - $Hold(Banana)\ and\ At(RoomC)$

PDDL in Vacuum Cleaner's World



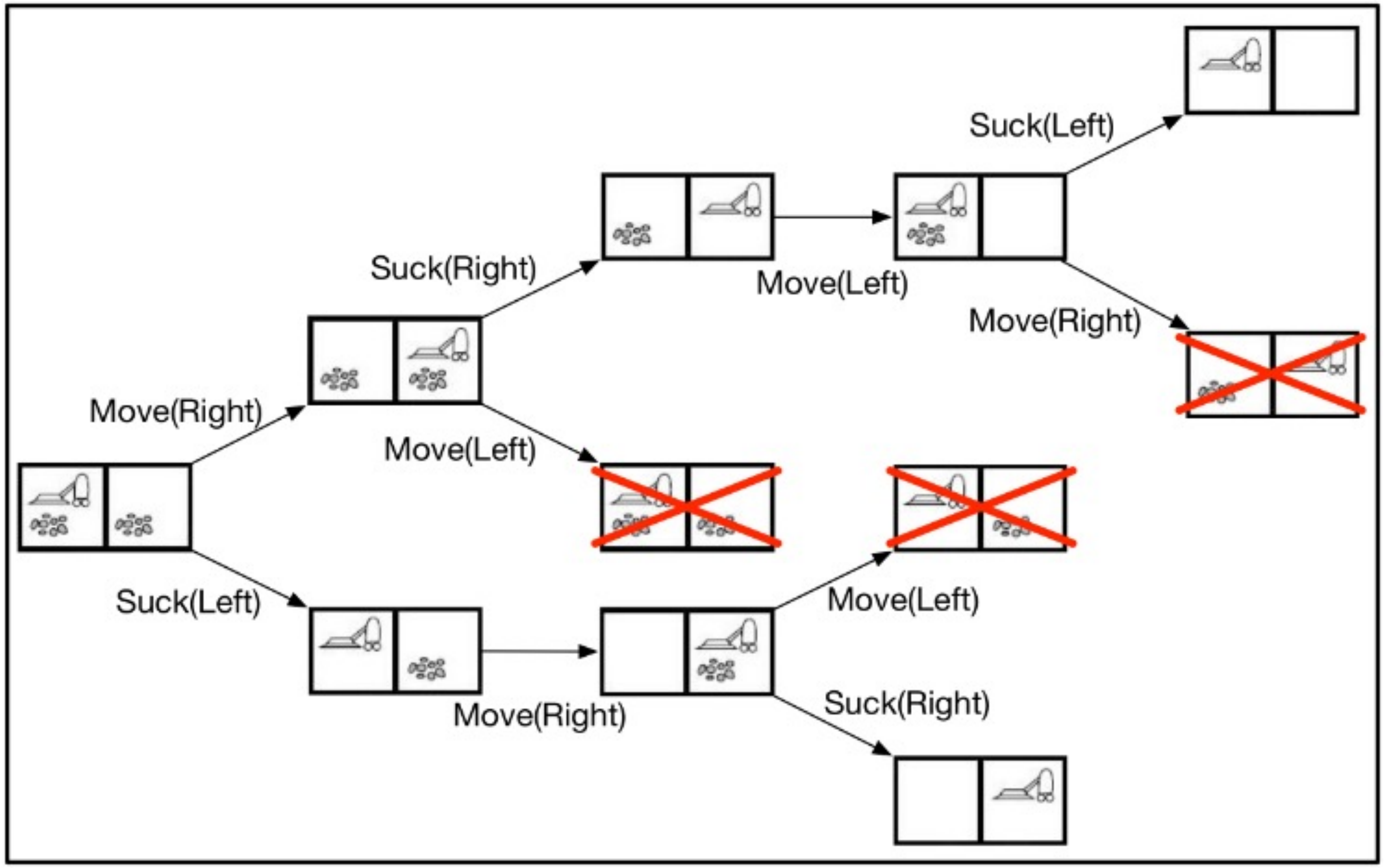
Update State with Action

- Delete list DEL(a): remove the fluents that appear as negative literals in the action's effects
- Add list ADD(a): add the fluents that are positive literals in the action's effects
- $s' = \text{RESULT}(s, a) = (s - \text{DEL}(a)) \cup \text{ADD}(a)$

- Example in the vacuum cleaner's world
  - $s_1 = At(Left)$, $a_1 = MoveRight()$
    - $\text{EFFECT}(a_1) = At(Right) \wedge \neg At(Left)$
    - $s_1 - \text{DEL}(a_1) = \emptyset$
    - $\text{RESULT}(s_1, a_1) = \emptyset \cup \text{ADD}(a_1) = At(Right)$
  - $s_2 = At(Right)$, $a_1 = Suck(Right)$
    - $\text{EFFECT}(a_2) = Clean(Right)$
    - $s_2 - \text{DEL}(a_2) = At(Right)$
    - $\text{RESULT}(s_2, a_2) = At(Right) \cup \text{ADD}(a_2) = At(Right) \wedge Clean(Right)$

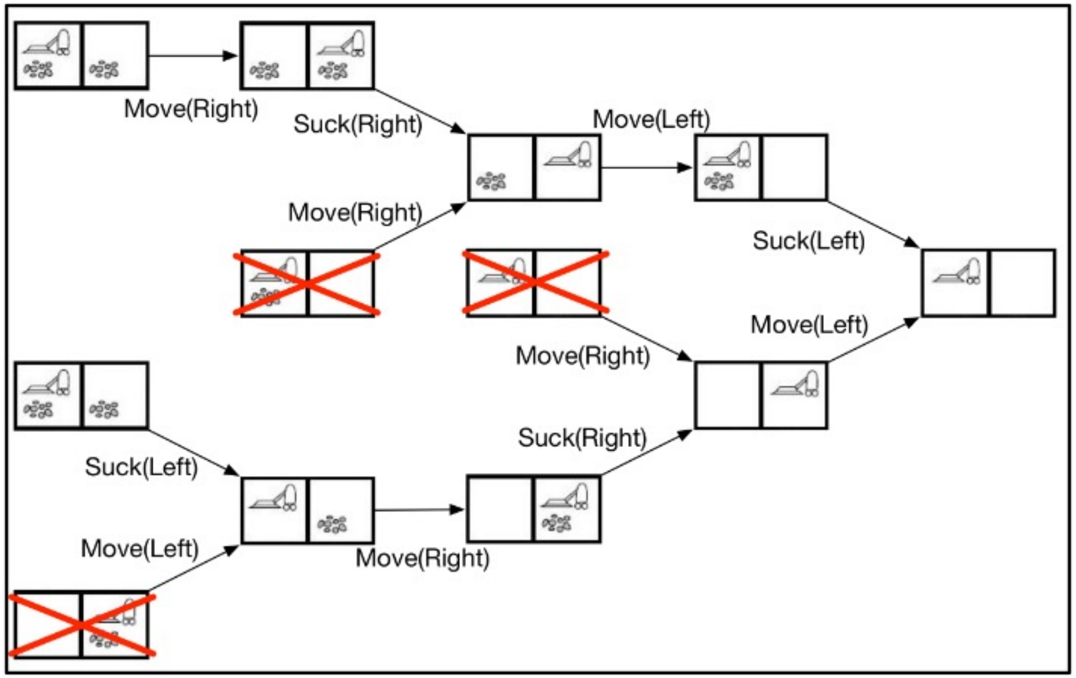
Planning Algorithms as State-Space Search

- Forward (progression) state-space search
  - Start with the initial state
  - Examine all the applicable actions for the current state
  - Avoid loop – never go back to previous states
  - Until reach a goal state



Planning Algorithms as State-Space Search

- Backward (regression) relevant state-space search
  - Start with a goal state (random if there are more than one)
  - Examine all the relevant actions
    - Could be the last step leading to the current state
    - At least one effect is an element of the current state
    - Has no effect that negates an element of the current state
  - Avoid loop
  - Until reach the initial state



Question

- The "have cake and eat cake" planning problem:

$$Init(Have(Cake, A))$$
$$Goal(Eaten(Cake, A) \land Eaten(Cake, B))$$
$$Action(Eat(Cake, x),$$
$$\quad \text{PRECOND}: Have(Cake, x)$$
$$\quad \text{EFFECT}: \neg Have(Cake, x) \land Eaten(Cake, x))$$
$$Action(Bake(Cake, x),$$
$$\quad \text{PRECOND}: \neg Have(Cake, x)$$
$$\quad \text{EFFECT}: Have(Cake, x))$$

- For the initial state, list all the applicable actions and the resultant state for each of them.
- List three different plans (sequences of actions from the initial state to the goal state).

JSS: an Example (Table)

- A solution is a schedule that processes these jobs with the machines (Gantt chart)

Job	Operation	Machine	ProcTime
1	AddEngine1	EngineHoist	30
	AddWheels1	WheelStation	30
	Inspect1	Inspector	10
2	AddEngine2	EngineHoist	60
	AddWheels2	WheelStation	15
	Inspect2	Inspector	10

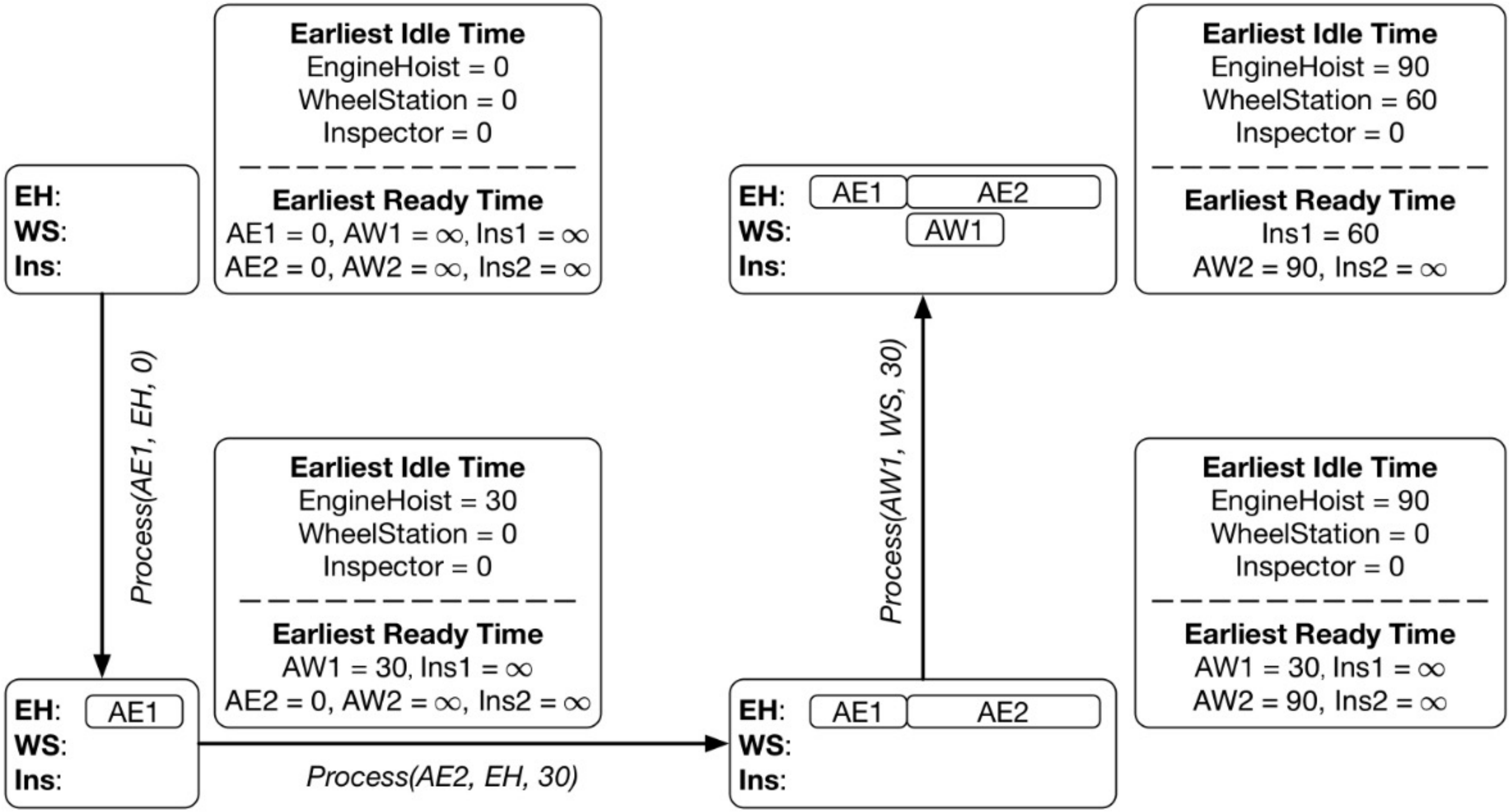
Search for Schedules

- Initial state
  - Empty schedule, t=0, all operations unprocessed
  - The first operation of each job is ready at time 0, all the other operations are not ready
  - All machines are idle at time 0
- Goal state: all operations processed
- Actions: Process(o, m, t)
  - Start processing operation o with machine m at time t
  - Precondition:
    - o unprocessed, and is ready at time t
    - m is idle at t
  - Effect:
    - o processed
    - next(o) (if exists) is ready at time t+ProcTime(o), and m is idle at t+ProcTime(o)
- How to decide t?

Deciding Starting Time of Action

- Non-delay: start the action as soon as possible
  - Operation earliest ready time
  - Machine earliest idle time
  - Earliest starting time: the later between the above two
- Find operation earliest ready time
  - Initial: 0 for the first operation, and infinity for others
  - When Process(o,m,t) is scheduled, then the earliest ready time of next(o) becomes t+ProcTime(o)
- Find machine earliest idle time
  - Initial: 0
  - When Process(o,m,t) is scheduled, then the earliest idle time of machine m becomes t+ProcTime(o)

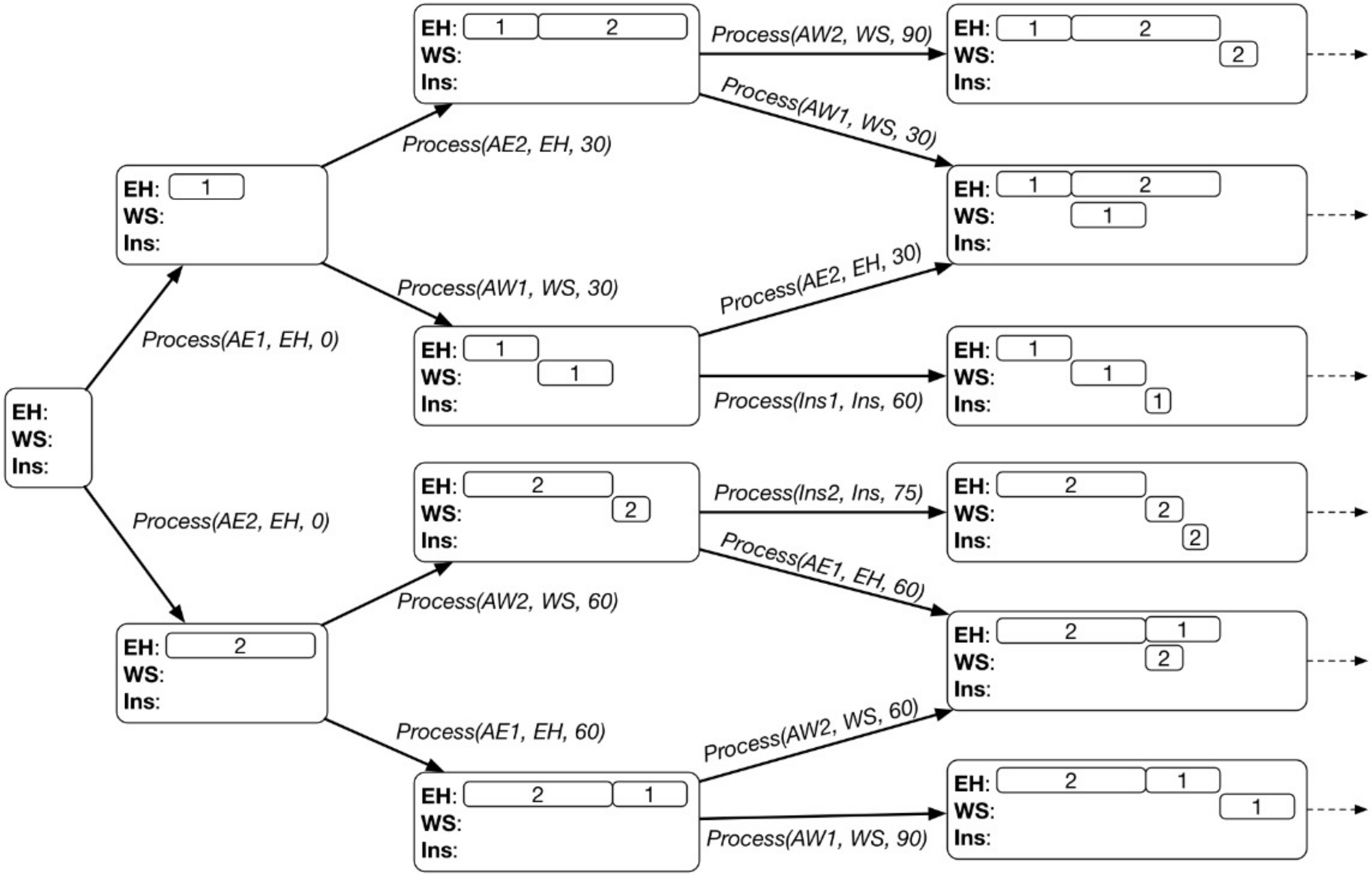
Update Earliest Ready and Idle Time



Forward State-Space Search

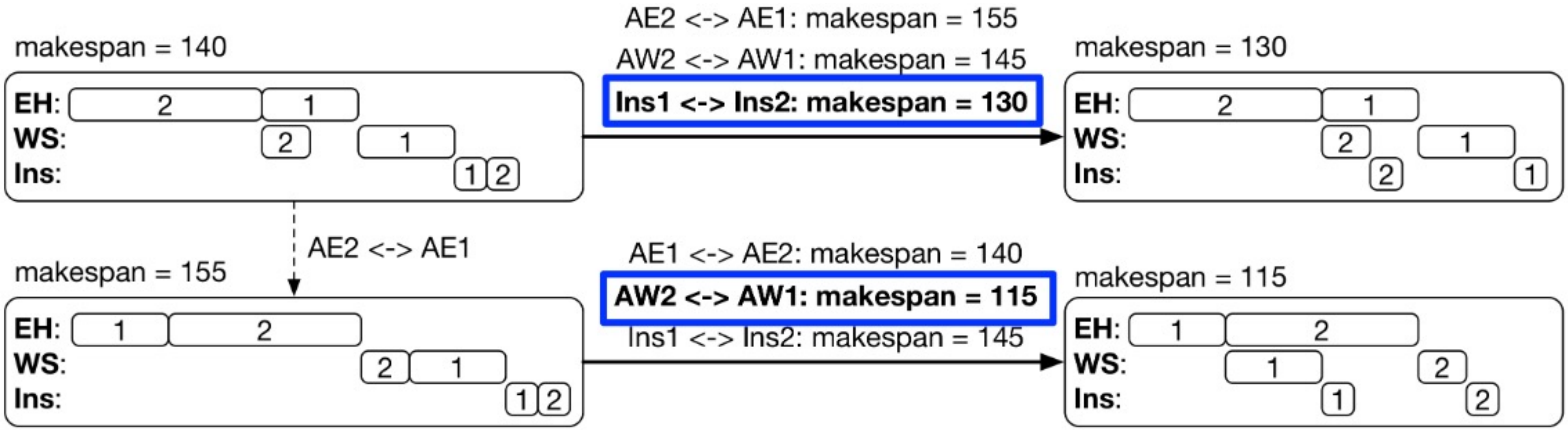
- Start from the initial state
  - Empty schedule, t=0, all operations unprocessed
  - The first operation of each job is ready at time 0, all the other operations are not ready
  - All machines are idle at time 0
- • Examine all the applicable actions Process(o,m,t)
  - Enumerate each unprocessed operation o and its machine m
  - Calculate the earliest starting time t
  - Applicable if $t < \infty$
- All the leaf nodes are goal states (all operations processed)
- Each schedule is a path from the root node to a leaf node

Forward State-Space Search



Local Search (Hill Climbing)

- Step 1. Random generate a scheduling s;
- Step 2. Examine all the neighbors in the neighborhood of s, and select the best neighbor s';
- Step 3. If s' is better than s, set s ← s', and go to Step 2. Otherwise return s.



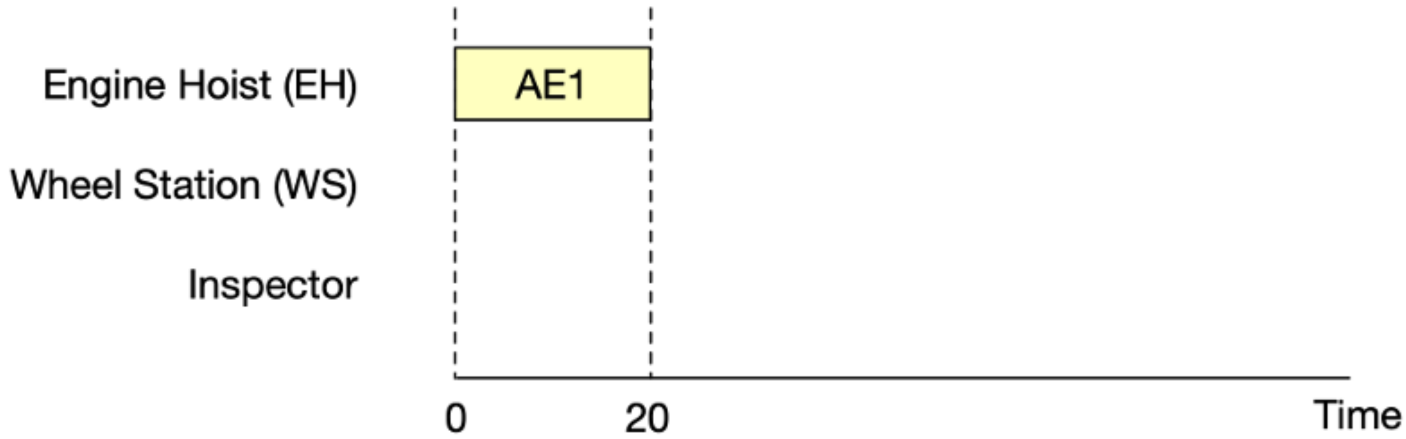
- Jump out of local optima: simulated annealing, genetic algorithms, …

Question

- For the car manufacturing scheduling problem, consider 3 jobs as summarized below:

	Arrival Time	Processing Time		
		AddEngine (AE)	AddWheels (AW)	Inspect (Ins)
Job 1	0	20	30	15
Job 2	0	45	20	20
Job 3	0	25	30	10

- Given the partial schedule below:



- List all applicable actions in this state, formatted as (Operation,Machine,StartTime)