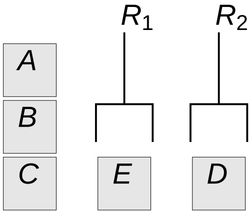
Planning (04)

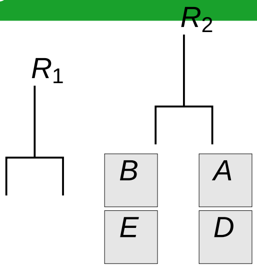
Partial-order planning

Alban Grastien
alban.grastien@rsise.anu.edu.au

Partial planning for flexible plans



Initial state



Goal states

A plan:

$Take(R_1, A, B), Take(R_2, B, C), Put(R_1, A, D), Put(R_2, B, E)$

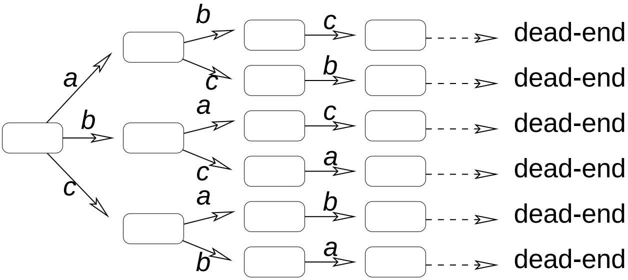
Actually, the order between some events are not important: $Put(R_1, A, D)$ and $Put(R_2, B, E)$ for instance.

- $Take(R_1, A, B) \prec Put(R_1, A, D)$
- $Take(R_1, A, B) \prec Take(R_2, B, C)$
- $Take(R_2, B, C) \prec Put(R_2, B, E)$

- State-space search produces inflexible plans

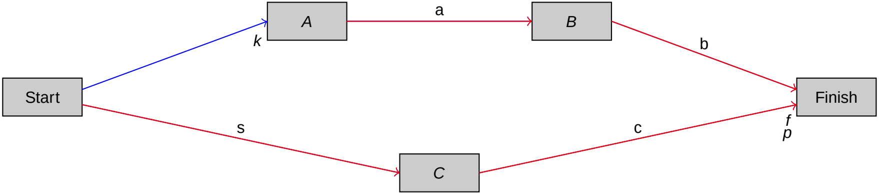
Partial planning for a smaller state-space

State-space search wastes time examining all the possible orderings of the same set of actions:



Least-commitment strategy: don't commit to orderings, instantiations, *etc.* until necessary

What is a partial-order plan?



- Partial-order plan:
  - a set of actions
  - a set of ordering constraints $A \prec B$ (without cycle)
  - a set of causal links (a causal link $A \xrightarrow{\alpha} B$ indicates that the precondition α of B is achieved by A)
  - a set of open preconditions (precondition of an action not achieved by another action)
- The partial-order plan is a solution is the set of open conditions is empty and if there is no conflict

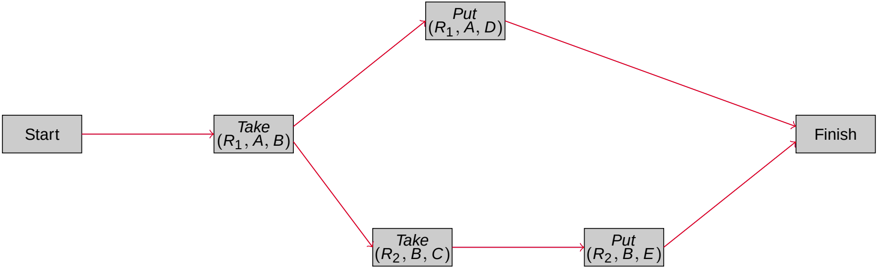
Conflict

- There is a conflict on a partial plan if:
  - there is a causal link $A \xrightarrow{\alpha} B$,
  - there is an action C such that $\neg\alpha$ is an effect of C, and
  - $C \not\prec A \wedge B \not\prec C$.

- α is achieved by A and needed by B. C removes α from the set of literals. C has to be <u>before</u> A or <u>after</u> B but not between A and B.

Linearization

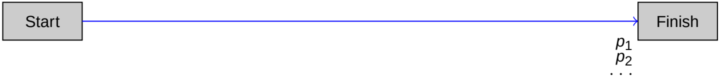
- Definition
  - All sequences of actions from the partial-ordered plan such that the ordering constraints are satisfied
- Example



Search in the space of partial-order plans

- Starting from an *empty* partial-ordered plan, we modify the plan until we find a solution
- Each state of the space of partial-order plans is a plan (and not a state of the system)

Initial (empty) plan



Modification of the current (unfinished) plan

- Choose an open precondition p on an action B
- Choose an existing action A or create a new action A that achieves the precondition B.
- Add the causal link $A \xrightarrow{p} B$ and the ordering $A \prec B$
- If A is a new state, also add Start $\prec A$ and add the preconditions of A to the set of preconditions
- Resolve the conflicts between the new causal link and the existing actions, and between the existing causal link and A if A is a new state. A conflict $A \xrightarrow{p} B$ with C is solved by adding the relation $A \prec C$ or $C \prec B$
- If the ordering constraint contains a loop, then we reached a dead end
- If there is no more precondition, then we have found a solution

Example: spare tire problem

- Changing a flat tire
- Initial state:

  $$At(Flat, Axle) \wedge At(Spare, Trunk)$$

- Actions:
  - $Remove(x, y)$

    $$At(x, y) \longrightarrow \neg At(x, y) \wedge At(x, Ground)$$

  - $PutOn(Spare, y)$

    $$At(Spare, Ground) \wedge \neg At(Flat, y) \longrightarrow$$
    $$\neg At(Spare, Ground) \wedge At(Spare, y)$$

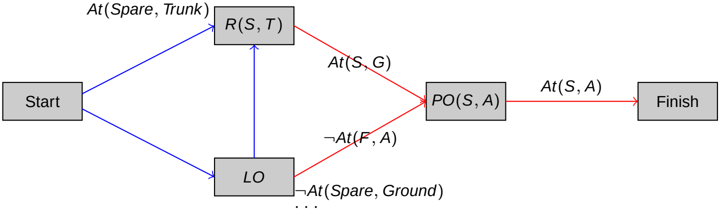
  - $LeaveOvernight$

    $$\longrightarrow \neg At(Spare, Ground) \wedge \neg At(Spare, Trunk) \wedge$$
    $$\neg At(Spare, Axle) \wedge \neg At(Flat, Ground) \wedge$$
    $$\neg At(Flat, Trunk) \wedge \neg At(Flat, Axle)$$

- Goal

  $$At(Spare, Axle)$$

Example



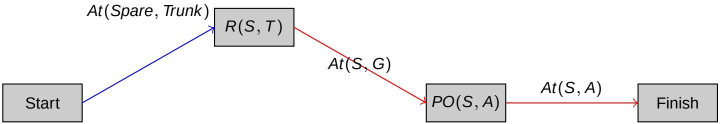
- Precondition $At(Spare, Axle)$
- Action $PutOn(Spare, Axle)$

- Precondition $At(Spare, Ground)$
- Action $Remove(Spare, Trunk)$

- Precondition $\neg At(Flat, Axle)$
- Action $LeaveOvernight$
- Conflict

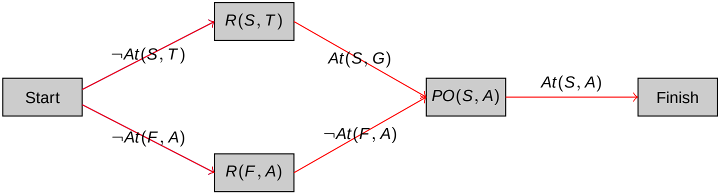
$LeaveOvernight \prec Remove(Spare, Trunk)$

Example



- Precondition $At(Spare, Trunk)$
- Action Start
- Conflict
- Backtracking
- Backtracking
- Backtracking

Example



- Precondition $\neg At(Flat, Axle)$
- Action $Remove(Flat, Axle)$

- Precondition $At(Flat, Axle)$
- Action Start

- Precondition $At(Spare, Trunk)$
- Action Start