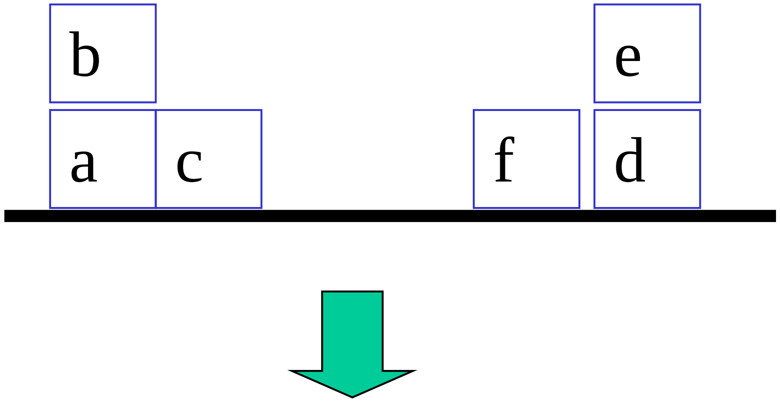
PARTIAL ORDER PLANNING

Ivan Bratko

FRI, University of Ljubljana

2021/22

PARTIAL ORDER PLANNING - EXAMPLE



Start state

Goal

PARTIAL ORDER PLAN

move(b,a,c) $\longrightarrow$ move(a, 1, b)

move(e, d, f) $\longrightarrow$ move(d, 6, e)

Arrows mean time precedence constraints:

A1 $\longrightarrow$ A2

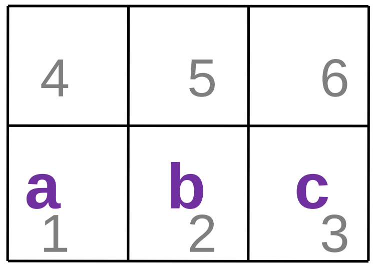
A1 must be executed before A2

TERMINOLOGY:
PARTIAL ORDER PLANNING and
“NONLINEAR PLANNING”

- Sometimes Partial order planning is also called “non-linear planning” (to emphasise that actions are not linearly ordered)

- The term “non-linear planning” may be confused with non-linearity w.r.t. achieving goals (i.e. STRIPS style of achieving goals “linearly” one-by-one)

ANOTHER EXAMPLE: ROBOTS ON THE GRID



Goal: at(a,3)

Ordered plan: m(b,2,5) $\rightarrow$ m(a,1,2) $\rightarrow$ m(c,3,6) $\rightarrow$ m(a,2,3)

Partially ordered plan:

$$\begin{tikzcd} m(b,2,5) \arrow[r] & m(a,1,2) \arrow[r] & m(a,2,3) \\ & m(c,3,6) \arrow[ur] & \end{tikzcd}$$

THE PRINCIPLE OF PARTIAL ORDER PLANNING, 3 ROBOTS EXAMPLE

- Start with empty plan (no actions). Then, to achieve goal at(a,3), add action A1 = m(a,2,3), to be executed at time T1. Then add other actions:

- At time T1: A1 = m(a,2,3), achieves goal at(a,3)
- At time T2: A2 = m(c,3,6), achieves c(3) for A1
- At time T3: A3 = m(a,1,2), achieves at(a,2) for A1
- At time T4: A4 = m(b,2,5), achieves c(2) for A3

- Time constraints (each action takes 1 unit of time):

  $T1 + 1 \leq FIN$ (Finishing time)

  $T2 + 1 \leq T1$

  $T3 + 1 \leq T1$

  $T4 + 1 \leq T3$
- All times nonnegative, minimize FIN
- Implementation with CLP, demo next slide

IMPLEMENTING TIME CONSTRAINTS WITH CLP(R) IN PROLOG

```
% Question to Prolog: state constraints
?- { T1 + 1 =< FIN,                   % Action A1 must end before finishing time
     T2 + 1 =< T1,                    % Action A2 must end before A1
     T3 + 1 =< T1,                    % Action A3 must end before A1
     T4 + 1 =< T3,                    % Action A4 must end before A3
     T1 >= 0, T2 >= 0, T3 >= 0, T4 >= 0 },   % All times nonnegative
     minimize(FIN).                   % Minimize finishing time

% Answer
T1 = 2.0,
T3 = 1.0,
T4 = 0.0,
FIN = 3.0,
{T2=<1.0},  {T2>=0.0}                 % T2 is anything in interval [ 0, 1 ]
```

Not for exam!

NOW SUPPOSE ROBOTS b AND c ARE A LITTLE SLOWER THAN a

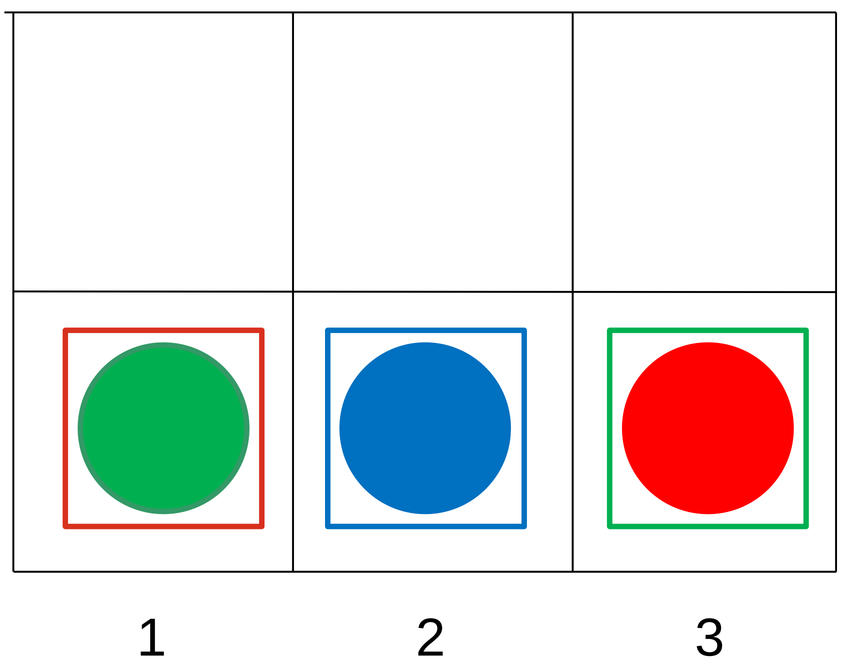
```
?- { T1+1 =< FIN,
      T2+1.5 =< T1,    % Robot c needs time 1.5 to move from 3 to 6
      T3+1 =< T1,
      T4+1.2 =< T3,    % Robot b needs time 1.2 to move from 2 to 5
      T1 >= 0, T2 >= 0, T3 >= 0, T4 >= 0 },   % All times nonnegative
   minimize(FIN).

% Answer
T1 = 2.2,
T3 = 1.2,
T4 = 0.0,
FIN = 3.2,
{T2=<0.7},
{T2>=0.0} ?     % Action A2 may start at any time in interval [ 0, 0.7]
```

Not for exam!

Robots on grid, more complex example

- Robots: red, blue, green; Locations at bottom: 1, 2, 3
- Start state: at(green, 1), at(blue, 2), at (red, 3)
- Goal: at(green, 3), at(blue ,2), at(red, 1)

  Goal locations are indicated by colour squares
- How many time steps are needed for this task?



PARTIAL ORDER PLANNING

DETAILS OF POP ALGORITHM

PARTIAL ORDER PLAN

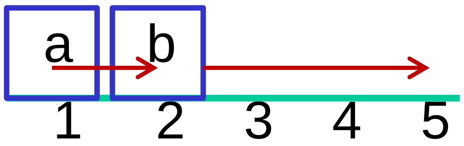
Each POP is defined by:

- set of actions $\{A_i, A_j, ...\}$
- set of ordering constraints e.g. $A_i < A_j$ (A_i before A_j)
- set of *causal links*

- Causal links are of form

  causes(A_i, P, A_j)

  read as: A_i achieves P for Aj



- Example causal link:

  causes(move(b, 2, 5), clear(2), move(a, 1, 2))

CAUSAL LINKS AND CONFLICTS

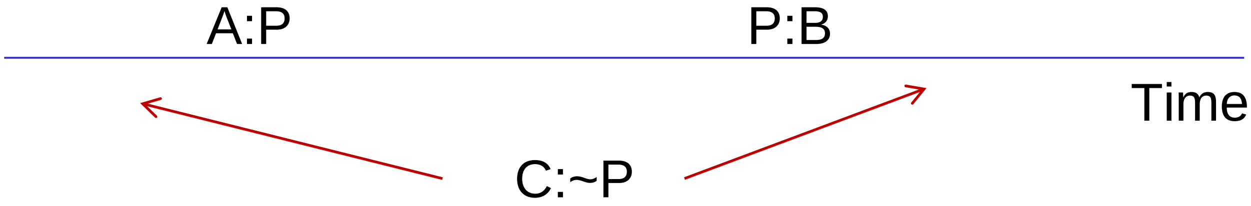
- Causal link causes(A, P, B) “protects” P in interval between A and B

- Action C *conflicts with* causes(A, P, B) if C’s effect is inconsistent with P (and B \= C). E.g. deletes(C, P), or adds(C, Q) and P&Q is contradiction; e.g. P = at(robot1, loc1), Q = at(robot1, loc3)

- Such conflicts are resolved by additional ordering constraints:

  C < A   or   B < C

  This ensures that C is outside interval A..B



PLAN CONSISTENT

- A plan is *consistent* if there is no cycle in the ordering constraints and no conflict

- E.g. a plan that contains

  $A<B$ and $B<C$ and $C<A$

  contains a cycle (therefore is not consistent, obviously impossible to execute!)

- Property of consistent plans:

  Every linearisation of a consistent plan is a total-order solution whose execution from the start state will achieve the goals of the plan

PARTIAL ORDER PLANNING ALGORITHM OUTLINE

- Search space of possible partial order plans (POP)
- Start plan is { Start, Finish}
- Start and Finish are virtual actions:
  - effect of Start is start state of the world, precond. of Start is true
  - precond. of Finish is goals of plan

true :: Start:: StartState $\longrightarrow$ ..... $\longrightarrow$ Goals :: Finish

SUCCESSOR RELATION BETWEEN POPs

A successor of a POP Plan is obtained as follows:

- Select an open precondition P of an action B in Plan (i.e. a precondition of B not guaranteed by other actions in Plan)
- Find an action A that achieves P

$\xrightarrow[A]{} P \qquad\qquad P \xrightarrow[B]{}$ "A causes P for B"

- A may be an existing action in Plan, or a new action; if new then add A to Plan and constrain: Start < A, A < Finish
- Add to Plan causal link causes(A,P,B) and constraint A < B
- Add appropriate ordering constraints to resolve all conflicts between:
  - new causal link and all existing actions, and
  - A (if new) and existing causal links

SEARCHING A SPACE OF POPs

- POP with no open precondition is a solution to our planning problem

- Some questions:

  - Heuristics for this search?

    Maybe: min. number of open preconditions, min. number of ordering constraints
  - How to handle “durative” actions?
  - Means-ends planning for game playing? How to take into account opponent’s actions?

- Heuristic estimates can be extracted from *planning graphs*;

  GRAPHPLAN is an algorithm for constructing planning graphs

EXAMPLE

- In robots on grid world 3x2, let start state be defined by:

  start([at(a,1), at(b,2), at(c,4), c(3), c(5), c(6)]).

- Goals: at(a,4), at(c,6)

- Simulate POP planner to find a POP plan for this problem

- Demo: A sketch of an implementation of plan extraction with CLP(R)

PART OF TRACE OF POP ALG.

Start state

c 4	5	6
a 1	**b** 2	3

Goals: at(a,4), at(c,6)

ACTIONS

START :: at(a,1), at(b,2), at(c,4), c(3), c(5), c(6)

FINISH

A1 = move(a,1,4)

A2 = move(c,5,6)

A3 = move(c,4,5)

OPEN PRECONDITIONS

at(a,4), at(c,6) :: FINISH

at(a,1), c(4) :: A1

at(c,5), c(6) :: A2

....

ORDER CONSTRAINTS

START < FINISH

A1 < FINISH

START < A1

A2 < FINISH

....

CAUSAL LINKS

causes(A1, at(a,4), FINISH)

causes(A2, at(c,6), FINISH)

causes(A3, c(4), A1)

...

```
% Solving ordering constraints with CLP in Prolog
% Problem: robots on grid
% Initial state: at(a,1),at(b,2),at(c,4),c(5),c(6),c(3)
% Goal: at(a,4), at(c,6)

plan( A1/T1, A2/T2, A3/T3, FT)  :-    % Ai/Ti = Action/Time, FT = finish
  A1 = m(a,1,4),                      % Achieves at(a,4) for Fin
     {0 =< T1, T1+1 =< FT},
  A2 = m(c,5,6),                      % Achieves at(c,6) for Fin
     {0 =< T2, T2+1 =< FT},
  A3 = m(c,4,5),                      % Achieves c(2) for A1
     {0 =< T3, T3+1 =< T1},
     {T3+1 =< T2},                    % Because A3 achieves at(c,5) for A2
  minimize( FT).                      % Minimize finishing time
```

Not for exam!

Question to Prolog with Constraint Solver CLP(R)

?- plan(A1/T1, A2/T2, A3/T3, FT).

Prolog answers as follows, including the times of actions:

A1 = m(a,1,4),

A2 = m(c,5,6),

A3 = m(c,4,5),

T1 = 1.0

T2 = 1.0

T3 = 0.0

FT = 2.0

Not for exam!