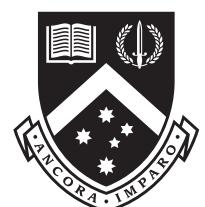


# Basic Planning

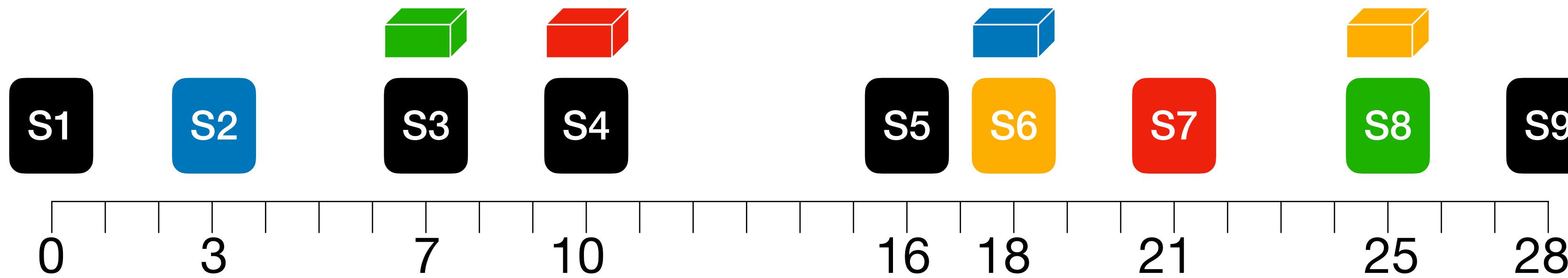
Peter Stuckey & Guido Tack



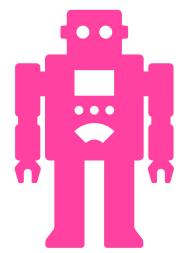
MONASH  
University

# Robot Pick-up and Drop-Off

- We have a robot that moves along a linear rail. It needs to visit a set of work stations.



**Finished!**



- Some work stations have items that must be taken to a given target work station. Other stations just need to be visited. The robot can carry multiple items.

# A PDDL domain

```
(:predicates
  (at ?s - workstation)
  (visited ?s - workstation)
  (pickup-loc ?p - packet ?s - workstation)
  (dropoff-loc ?p - packet ?s - workstation)
  (carrying ?p - packet)
  (delivered ?p - packet)
)
```

# A PDDL domain

```
(:action move
  :parameters (?from - workstation ?to - workstation)
  :precondition (and (at ?from) (not (= ?from ?to)))
  :effect (and
            (not (at ?from))
            (at ?to)
            (visited ?to)
  )
)
```

# A PDDL domain

```
(:action pickup
  :parameters (?p - packet ?s - workstation)
  :precondition (and (pickup-loc ?p ?s)
                      (at ?s)
                      (not (delivered ?p)))
                      (not (carrying ?p)))
  :effect (carrying ?p)
)

(:action dropoff
  :parameters (?p - packet ?s - workstation)
  :precondition (and (dropoff-loc ?p ?s)
                      (at ?s)
                      (carrying ?p)))
  :effect (and (delivered ?p) (not (carrying ?p)))
)
```

# A MiniZinc model

- Fix a time horizon
- Predicates are modelled as Boolean variables
- Actions are modelled as decision variables over an enumerated type (plus additional variables for parameters)
- Constraints link action at each time point to preconditions and effects
- Effects on all predicates need to be made explicit
- Additional NONE action is added to pad up to time horizon

# Cost-optimal planning

- Find a plan for the robot with the shortest travel distance
- PDDL: add a fluent for the cost, define an effect in the move action and a metric in the problem definition.
- MiniZinc: add an integer variable for each time step to represent the fluent, and an optimisation goal.

# Key insights to improve the model

- Each workstation must be visited exactly once
- Each packet requires exactly one pickup and one dropoff action
- We have a fixed time horizon!

# Further improvements

- We can turn some of the Boolean variables into finite domain variables
- Some of the predicates are only used to track if something happens eventually — turn into constraints

# One additional insight

- This is essentially a TSP
- Model simply as the sequence of workstations to be visited
- Add precedence constraints for the pick-up and delivery locations

# Summary

- Planning problems *can* be modelled in a “planning-as-satisfiability” style
- Requires fixed horizon
- Easy to express some aspects using integer (or enum) variables
- Some constraints can be expressed over the entire plan (instead of using individual Boolean variables)
- CP solvers are strong when the horizon is tight (essentially a permutation problem)