Advanced planning

Jana Tumova

Recap: Classical planning

- PDDL

$Init(Mobile(Robot) \wedge At(Robot, Left))$

$Goal(\ Clean(Left) \wedge Clean(Right))$

$Action(L(r),$

$\quad PRECOND: Mobile(r)$

$\quad EFFECT: At(r, Left))$

$Action(R(r),$

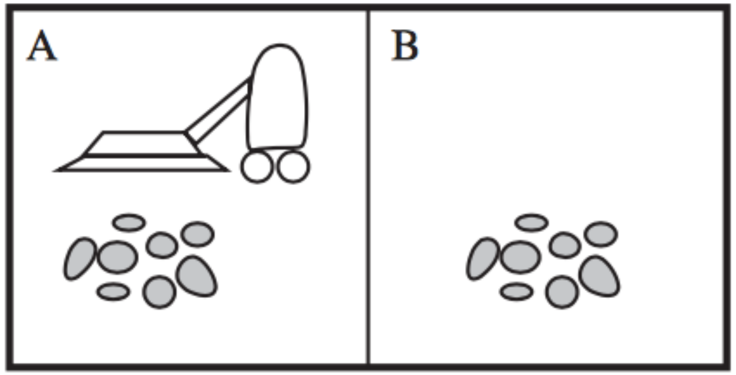
$\quad PRECOND: Mobile(r)$

$\quad EFFECT: At(r, Right)$

$Action(S(r, p),$

$\quad PRECOND: Mobile(r) \ \wedge At(r, p)$

$\quad EFFECT: Clean(p)$



Recap: Classical planning

- Database semantics

$\boxed{Mobile(Robot), At(Robot, Left)}$

vs.

$$\boxed{\begin{array}{l} Mobile(Robot), \\ At(Robot, Left), \neg At(Robot, Right) \\ \neg Clean(Left), \neg Clean(Right), \\ \neg Clean(Robot), \neg At(Robot, Robot) \\ \neg Mobile(Left), \neg Mobile(Right), \\ \neg At(Left, Left), \neg At(Left, Right), \\ \neg At(Right, Left), \neg At(Right, Right), \\ \neg At(Left, Robot), \neg(Right, Robot) \end{array}}$$

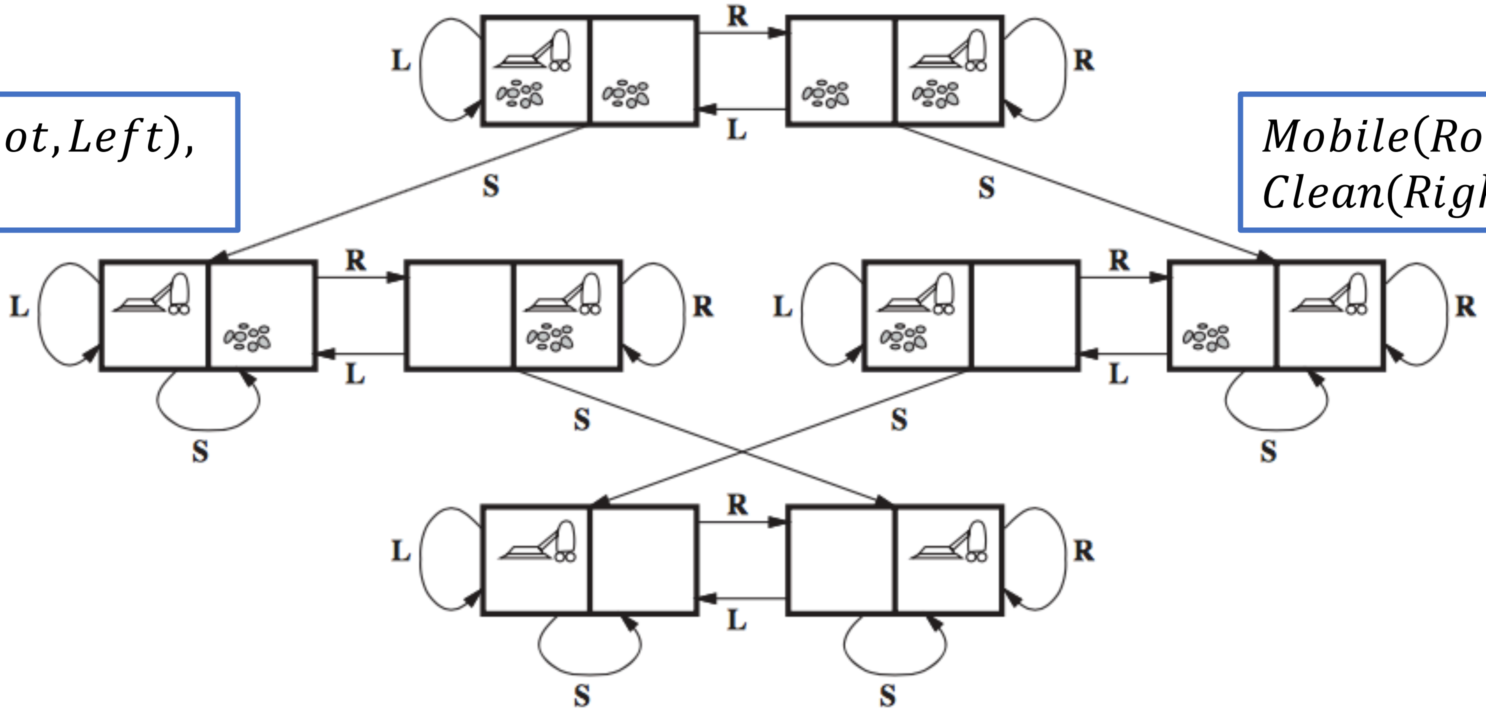
Classical planning via forward search

Mobile(Robot), At(Robot, Left)

Mobile(Robot), At(Robot, Right)

Mobile(Robot), At(Robot, Left), Clean(Left)

Mobile(Robot), At(Robot, Right), Clean(Right)



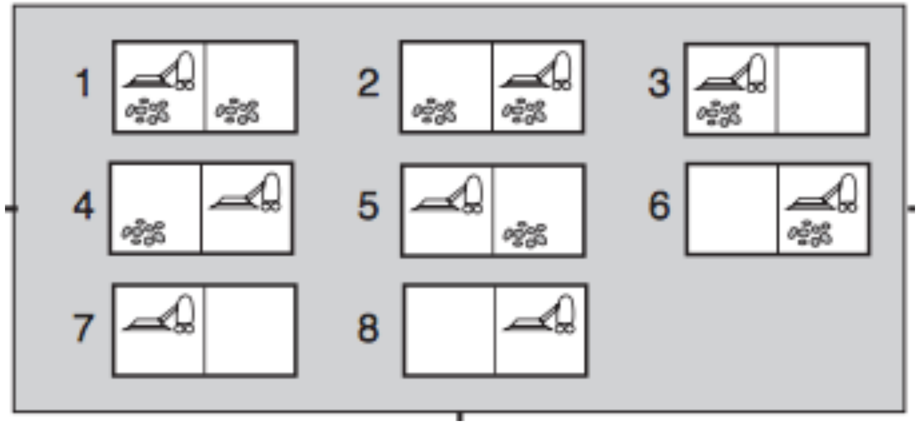
Mobile(Robot), At(Robot, Left), Clean(Left), Clean(Right)

Mobile(Robot), At(Robot, Right), Clean(Left), Clean(Right)

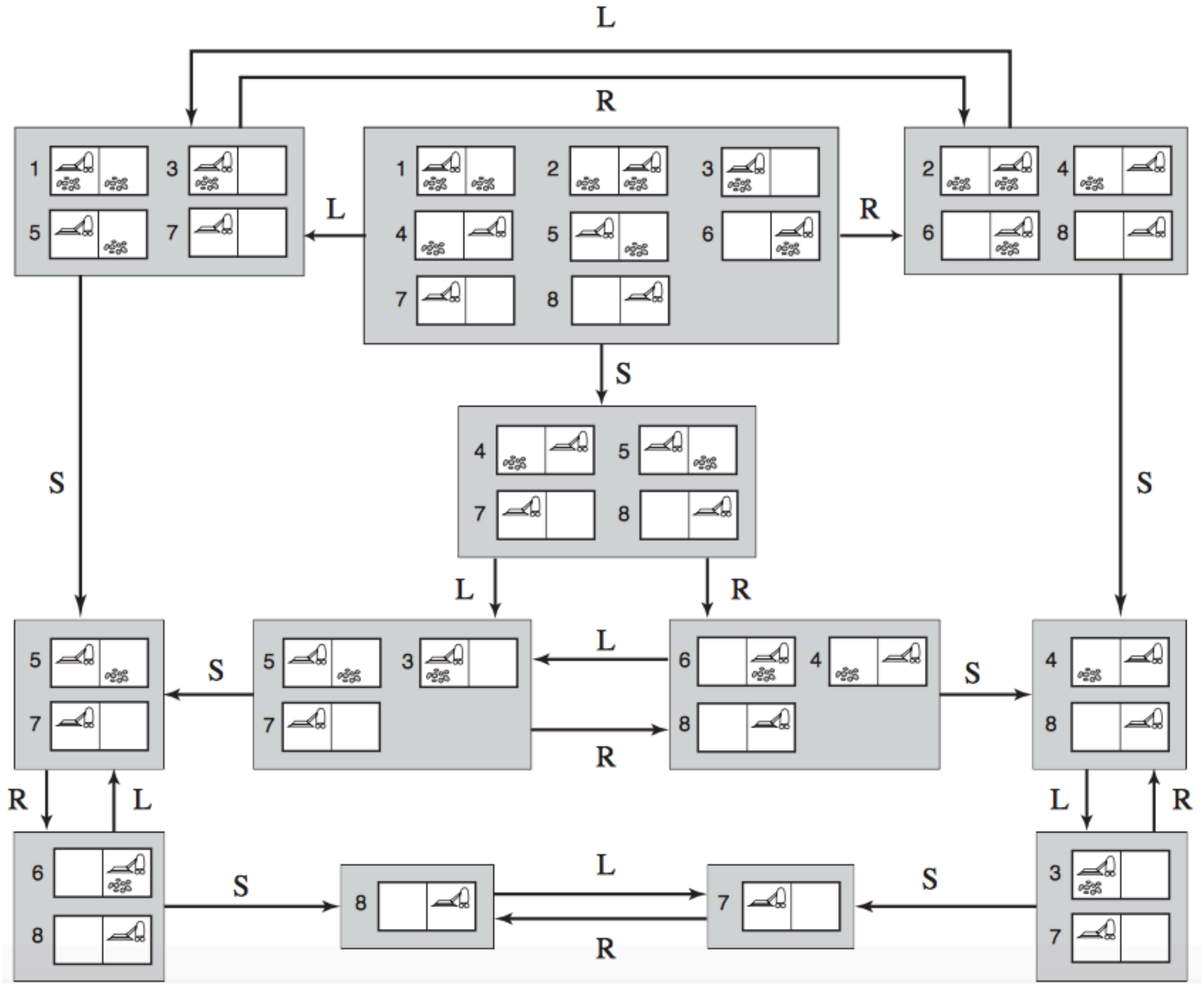
Sensorless agent

- What if the agent does cannot observe the truth values of all ground fluents?
- Uncertainty
- A set of possible states = belief state

Example: What if the agent cannot observe where it is and whether a spot is dirty or clean?



Belief state space



Sensorless agent

- PDDL, but **no database semantics any more**

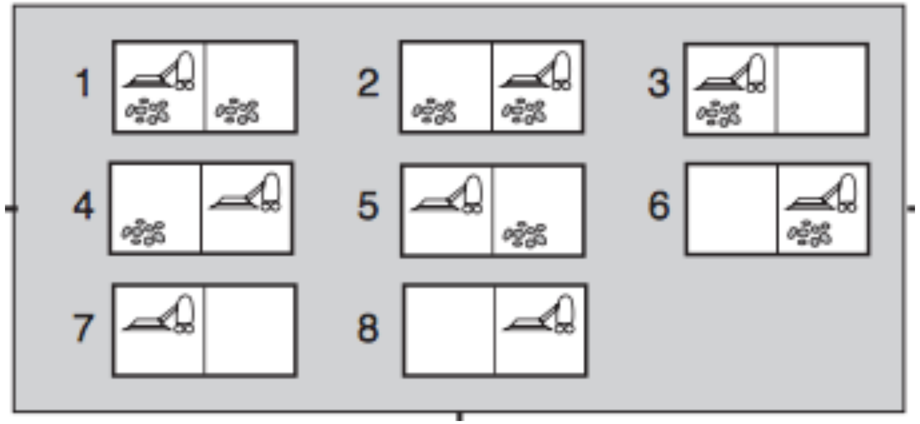
$$Init(Mobile(Robot))$$

$$Goal(\, Clean(Left) \wedge Clean(Right))$$

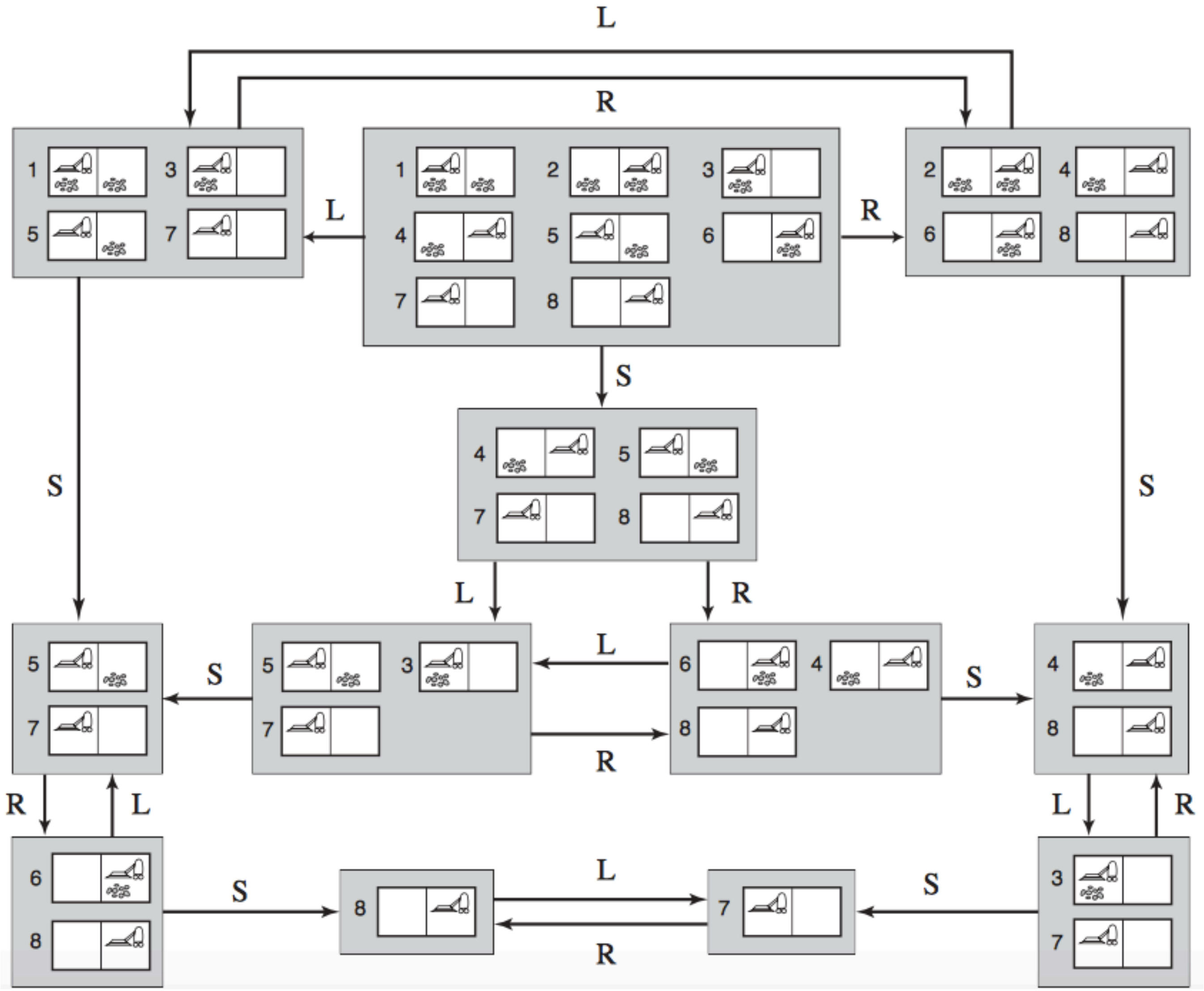
$$Action(L(r),$$
$$\quad PRECOND: Mobile(r)$$
$$\quad EFFECT: At(r, Left))$$

$$Action(R(r),$$
$$\quad PRECOND: Mobile(r)$$
$$\quad EFFECT: At(r, Right)$$

$$Action(S(r,p),$$
$$\quad PRECOND: Mobile(r) \;\wedge At(r,p)$$
$$\quad EFFECT: Clean(p)$$



Belief state space



Q: What are space requirements for remembering a belief state?

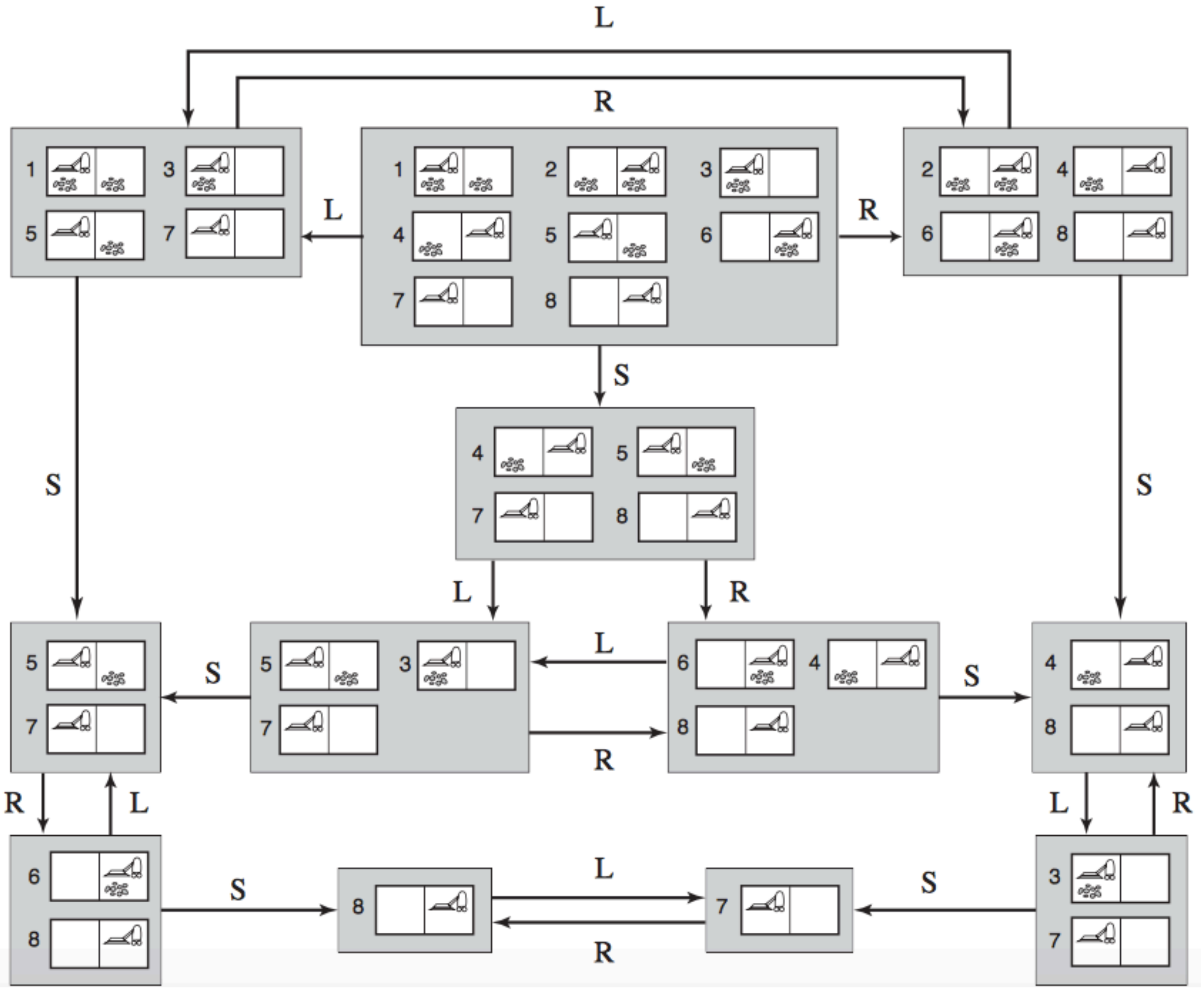
Belief state space

$O(n)$ vs. $O(2^n)$

$Mobile(Robot)$

$Mobile(Robot), At(Robot, Left)$

$Mobile(Robot), At(Robot, Right)$



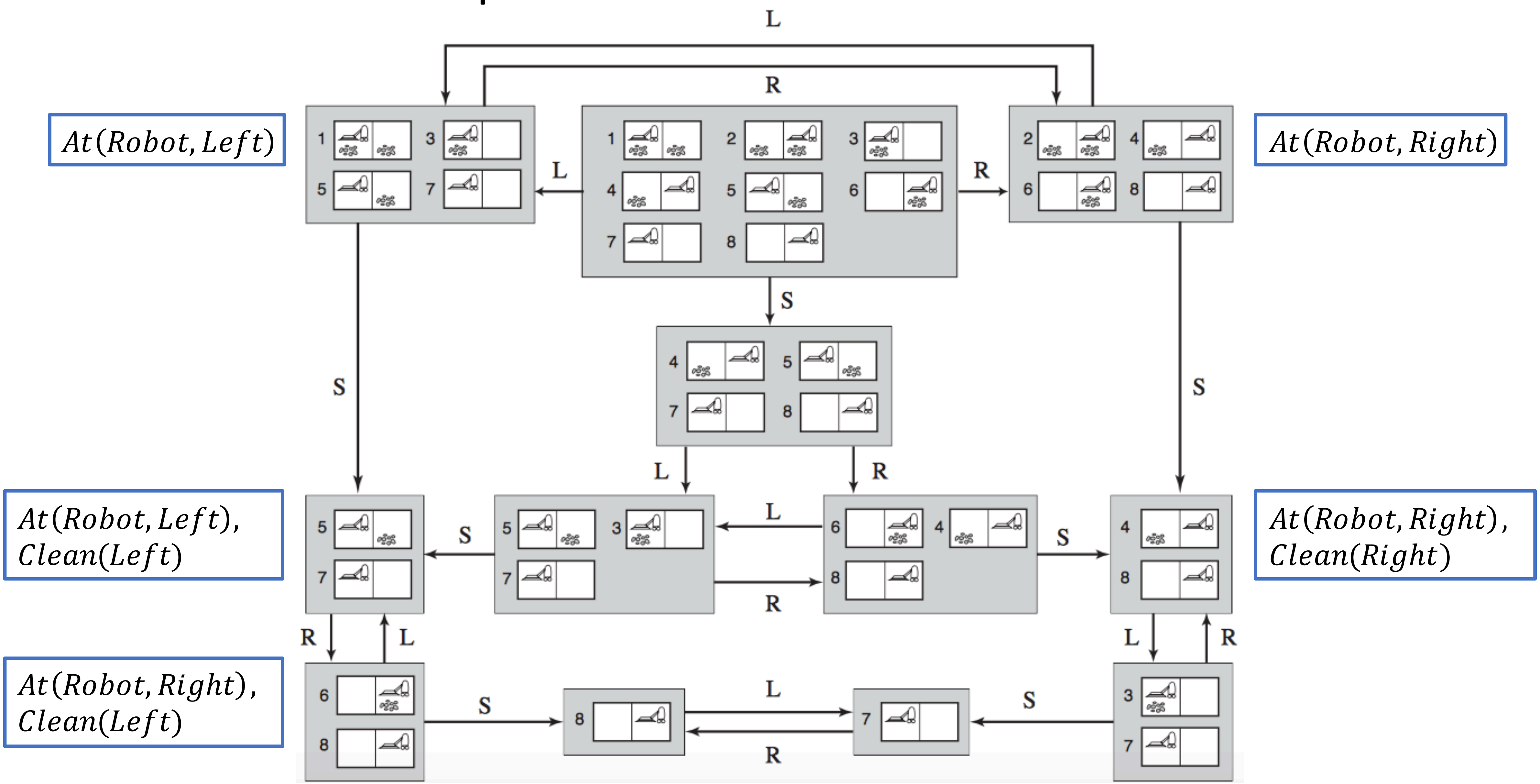
$Mobile(Robot), At(Robot, Left), Clean(Left)$

$Mobile(Robot), At(Robot, Right), Clean(Right)$

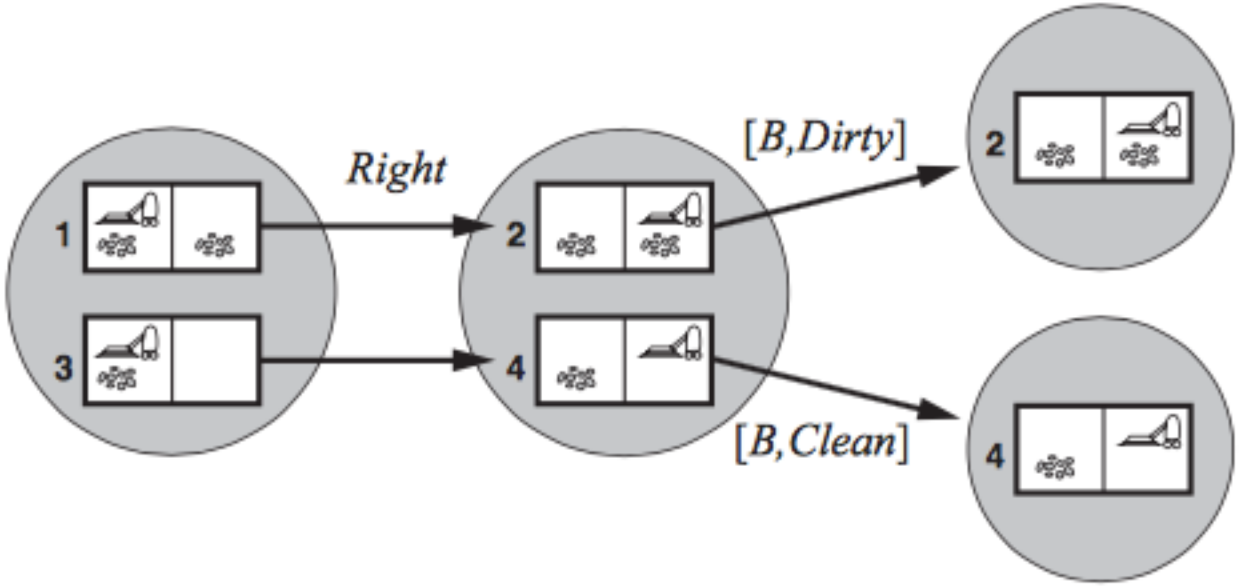
$Mobile(Robot), At(Robot, Right), Clean(Left)$

Belief state space

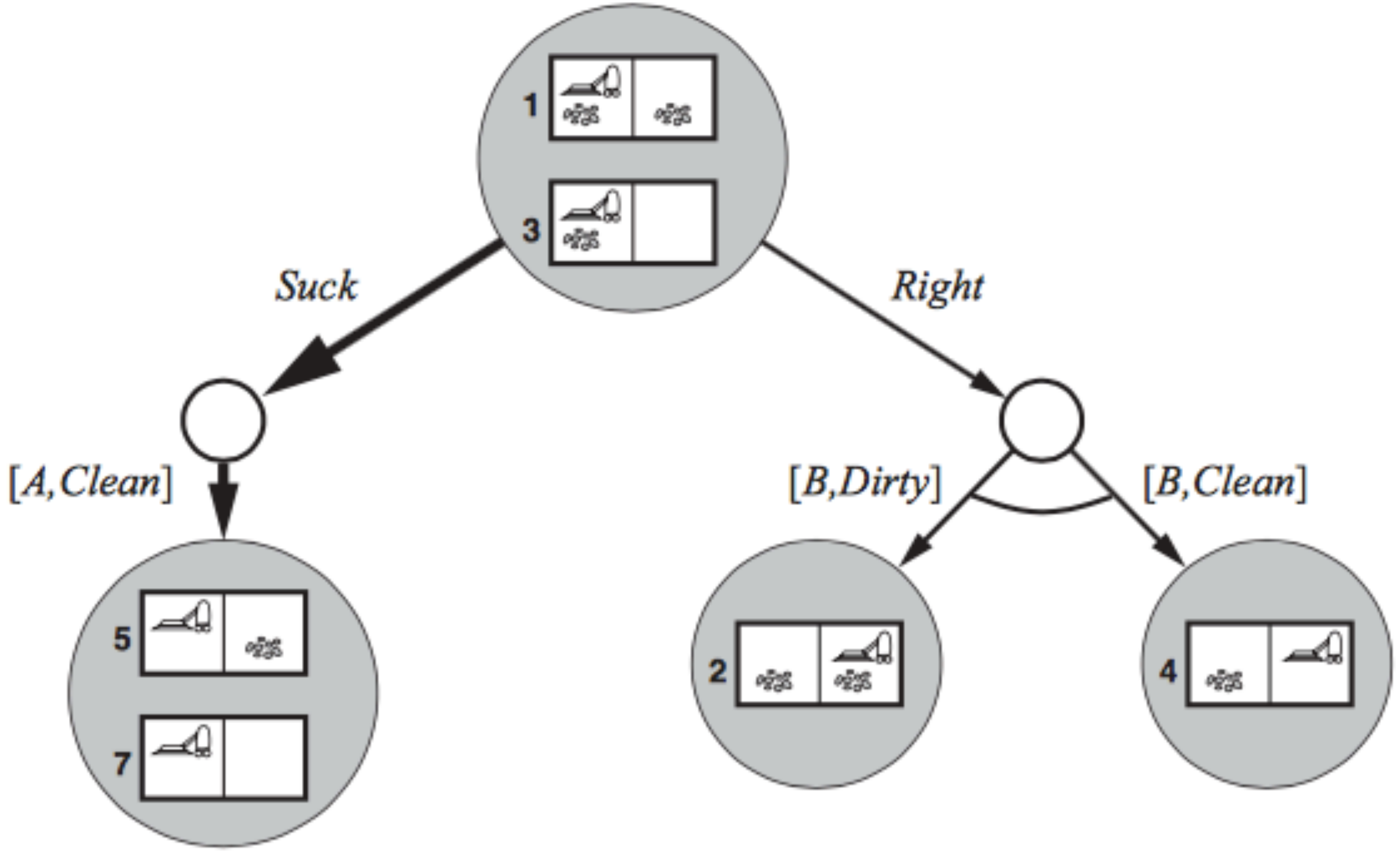
$O(n)$ vs. $O(2^n)$



Local sensing



Uncertainty in action execution



Q: Is the classical search through the belief state space good enough?

Recap: Classical planning

- PDDL

$Init(Object(Table) \land Object(Chair) \land Can(C1) \land Can(C2) \land$

$\quad Color(Table, Green) \land Color(Chair, Green) \land Color(C1, Blue) \land Color(C2, Red))$

$Goal(Color(Chair, Blue) \land Color(Table, Blue))$

$Action(RemoveLid(can),$

$\quad PRECOND: Can(can)$

$\quad EFFECT: Open(can))$

$Action(Paint(x, can, c),$

$\quad PRECOND: Object(x) \land Can(can) \land Color(can, c) \land Open(can)$

$\quad EFFECT: Color(x, c)$

$Object(Table), Object(Chair),$
$Can(C1), Can(C2),$
$Color(Table, Green), Color(Chair, Green),$
$Color(C1, Blue), Color(C2, Red)$

Recap: Classical planning

- PDDL

$Init(Object(Table) \land Object(Chair) \land Can(C1) \land Can(C2) \land$
$\quad Color(Table, Green) \land Color(Chair, Green) \land Color(C1, Blue) \land Color(C2, Red))$

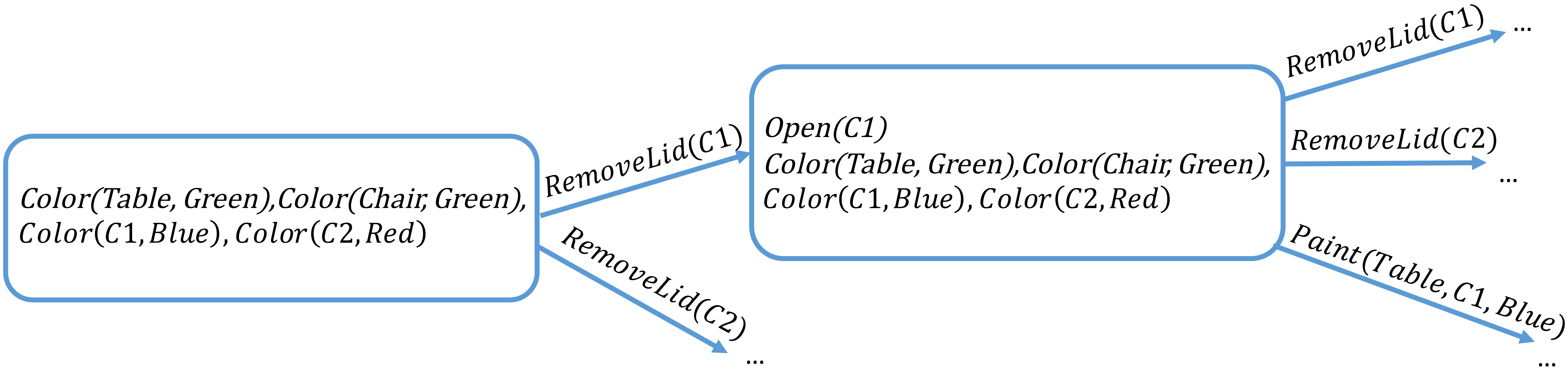
$Goal(Color(Chair, Blue) \land Color(Table, Blue))$

$Action(RemoveLid(can),$
$\quad PRECOND: Can(can)$
$\quad EFFECT: Open(can))$

$Action(Paint(x, can, c),$
$\quad PRECOND: Object(x) \land Can(can) \land Color(can, c) \land Open(can)$
$\quad EFFECT: Color(x, c)$

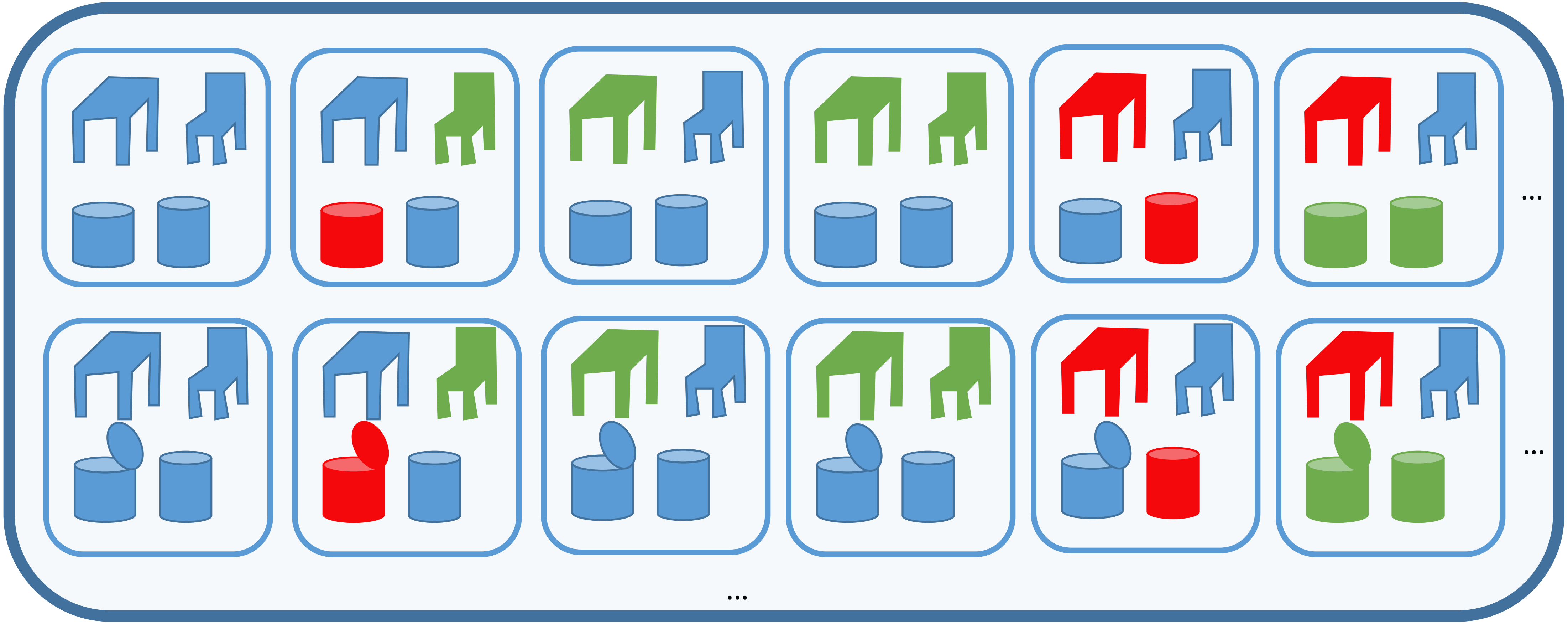
$Color(Table, Green), Color(Chair, Green),$
$Color(C1, Blue), Color(C2, Red)$

Recap: Forward search

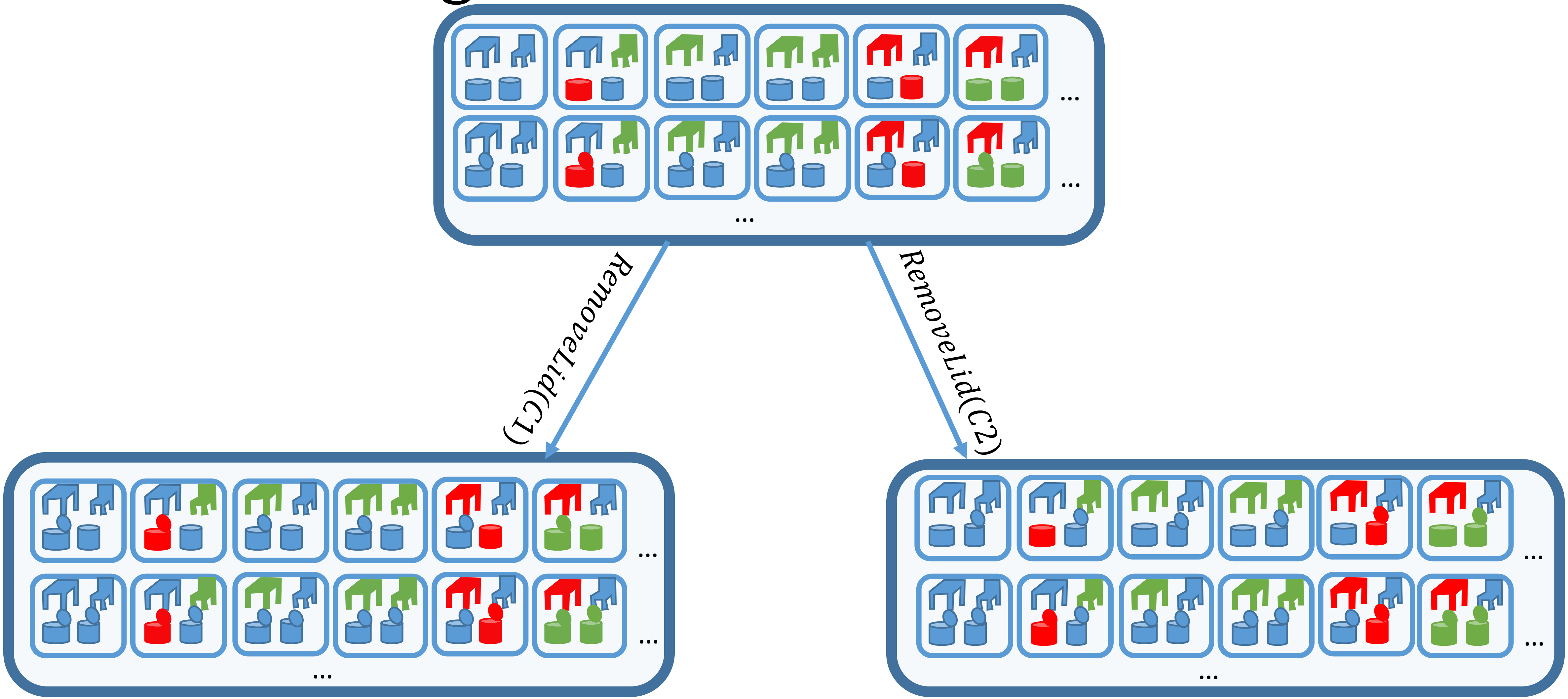


Sensorless agent

- What if the agent cannot observe a color of the object and whether a can is open or not?

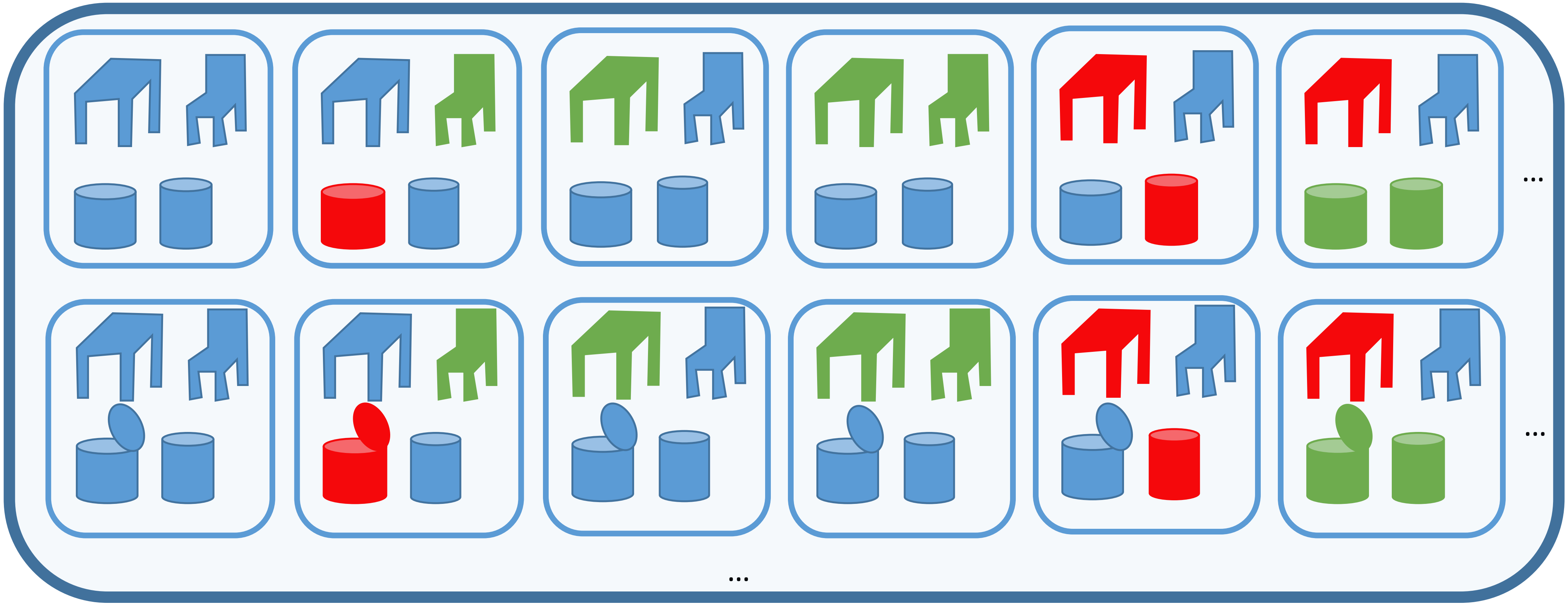


Sensorless agent



Sensorless planning

- No more database semantics in PDDL

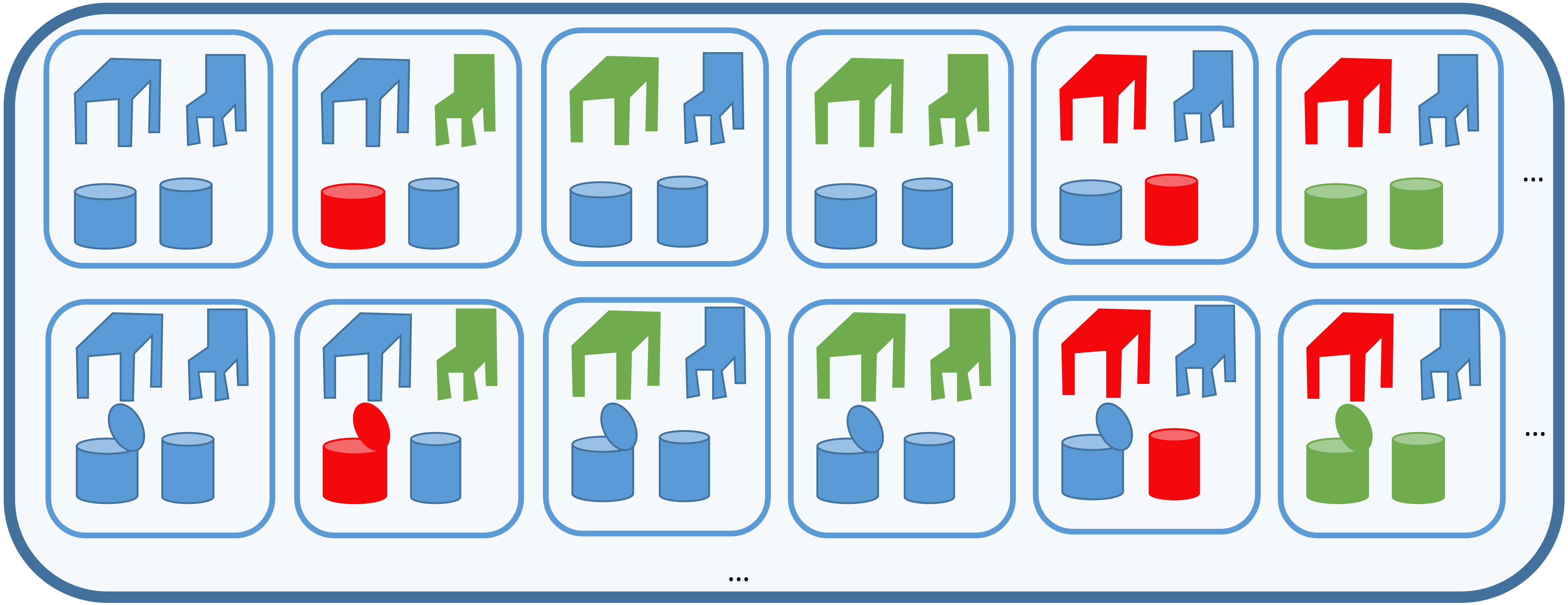


$Object(Table), Object(Chair), \neg Object(C1), \neg Object(C2)$
$Can(C1), Can(C2), \neg Can(Table), \neg Can(Chair)$

Sensorless planning

- No more database semantics in PDDL

$O(2^n)$



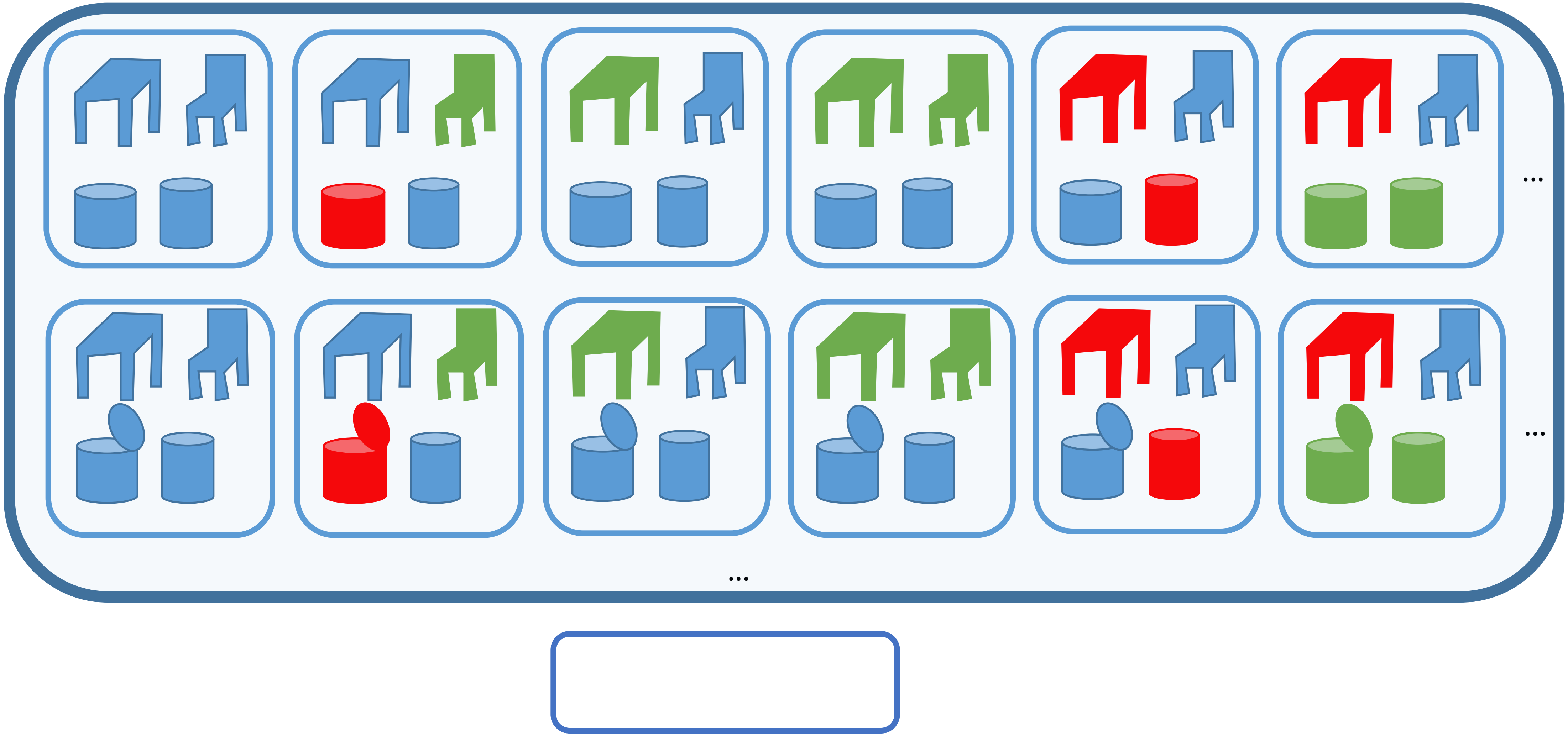
$O(n)$

$Object(Table), Object(Chair), \neg Object(C1), \neg Object(C2)$
$Can(C1), Can(C2), \neg Can(Table), \neg Can(Chair)$

Sensorless planning

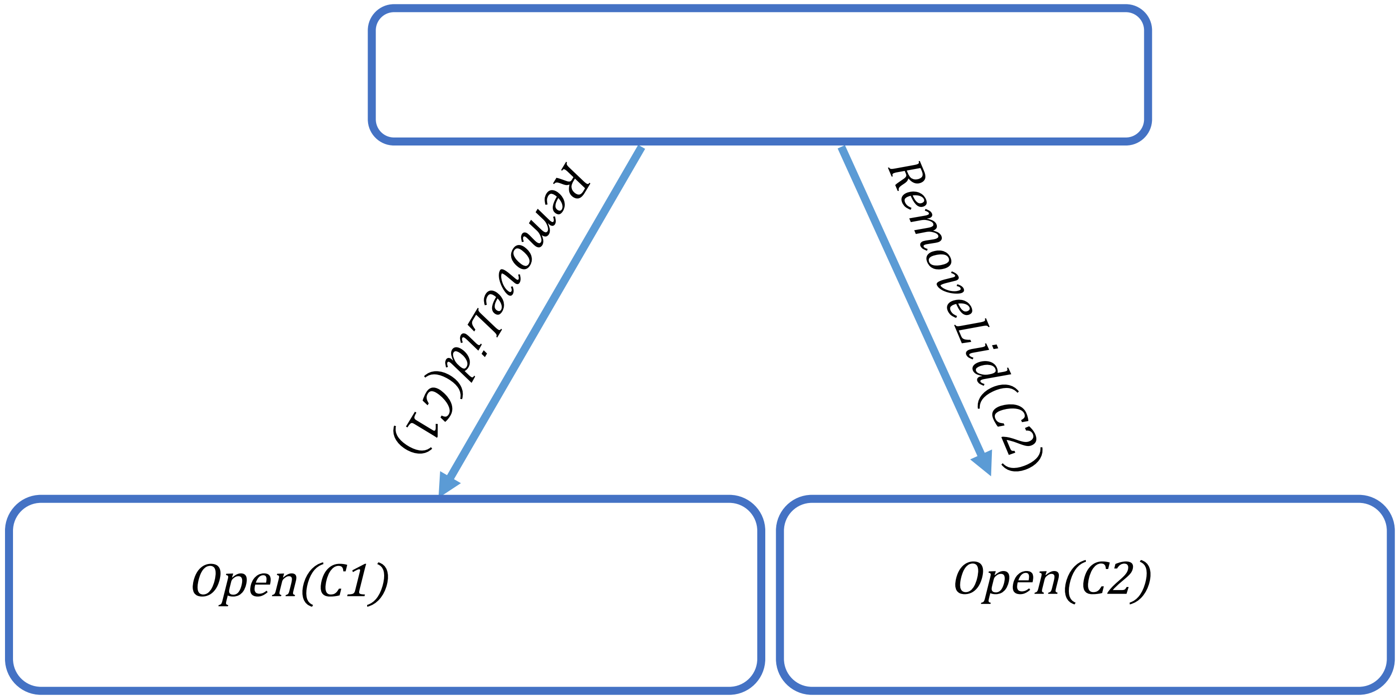
- No more database semantics in PDDL

$O(2^n)$



$O(n)$

Sensorless planning



Planning with limited observation

- What if we have limited sensing? What if we only see the table and what if we do not know what are the colors of the paints in the cans?

$Init(Object(Table) \land Object(Chair) \land Can(C1) \land Can(C2) \land InView(Table))$

$Goal(Color(Chair, c) \land Color(Table, c))$

$Action(RemoveLid(can),$
$\quad PRECOND: Can(can)$
$\quad EFFECT: Open(can))$

$Action(Paint(x, can),$
$\quad PRECOND: Object(x) \land Can(can) \land Color(can, c) \land Open(can)$
$\quad EFFECT: Color(x, c))$

$Percept(Color(x, c),$
$\quad PRECOND: Object(x) \land InView(x)$

$Percept(Color(can, c),$
$\quad PRECOND: Can(can) \land InView(can) \land Open(can)$

$Action(LookAt(x),$
$\quad PRECOND: InView(y) \land x \neq y$
$\quad EFFECT: InView(x) \land \neg InView(y)$

Planning in partially observable environments

- More topics:
  - Contingent planning
    - With conditional branching based on percepts
  - Online replanning
    - To reduce the complexity of contingent planning
    - If the agent's model of the workd us ubcirrect

Scheduling and planning with limited resources

$$
\begin{aligned}
&Jobs(\{AddEngine1 \prec AddWheels1 \prec Inspect1\},\\
&\qquad \{AddEngine2 \prec AddWheels2 \prec Inspect2\})
\end{aligned}
$$

$$Resources(EngineHoists(1), WheelStations(1), Inspectors(2), LugNuts(500))$$

$$
\begin{aligned}
&Action(AddEngine1, \text{DURATION}{:}30,\\
&\qquad \text{USE}{:}EngineHoists(1))\\
&Action(AddEngine2, \text{DURATION}{:}60,\\
&\qquad \text{USE}{:}EngineHoists(1))\\
&Action(AddWheels1, \text{DURATION}{:}30,\\
&\qquad \text{CONSUME}{:}LugNuts(20), \text{USE}{:}\ WheelStations(1))\\
&Action(AddWheels2, \text{DURATION}{:}15,\\
&\qquad \text{CONSUME}{:}LugNuts(20), \text{USE}{:}\ WheelStations(1))\\
&Action(Inspect_i, \text{DURATION}{:}10,\\
&\qquad \text{USE}{:}Inspectors(1))
\end{aligned}
$$

Hierarchical planning

$$
\begin{aligned}
&Refinement(Go(Home, SFO),\\
&\quad \text{STEPS: } [Drive(Home, SFOLongTermParking),\\
&\qquad\qquad\quad Shuttle(SFOLongTermParking, SFO)]\ )\\
&Refinement(Go(Home, SFO),\\
&\quad \text{STEPS: } [Taxi(Home, SFO)]\ )
\end{aligned}
$$

Planning gets even more interesting when more aspects are involved

- Limited resources: time, cost, capacity,…
- Uncertainty
- Multiple agents
- Different criteria: optimality,…
- Robotics: dynamical constraints

- Integration into context, integration with other AI methods

Tons of PDDL tools and extensions

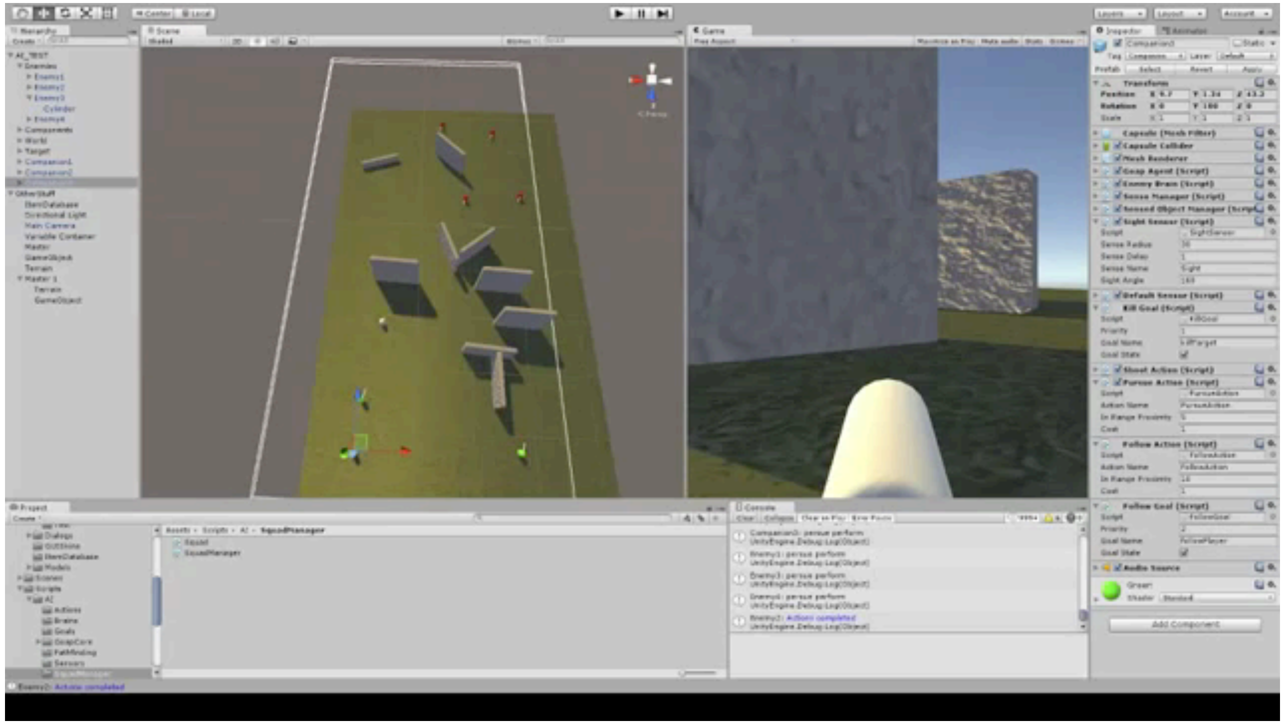
- PDDL, PDDL 2.1, PDDL 3, PDDL +
- Classical: LAMA, HSP, FF, MetricFF, SATplan, FastDownward...
- Temporal heuristic estimates, linear constraints: LPG, RFD, SAPA, POPF, COLIN
- LTL: OPTIC (POPF), Hplan-P
- Non-linear constraints: MIP, UPMurphi, DiNo, SMTplan
- Integration into ROS: ROSPlan

Source: AAAI ROSPlan tutorial by Cashmore, Magazzeni, 2017.

Some examples of the state of the art

STRIPS in Games: Goal-Oriented Action Planning

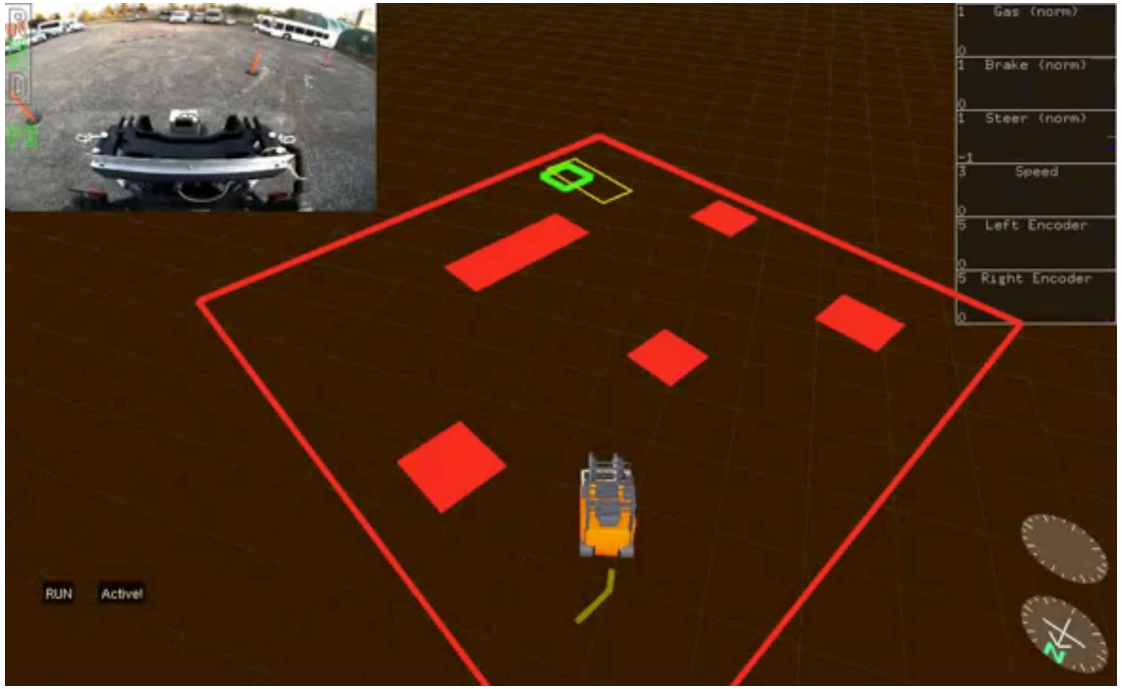
- F.E.A.R and others
- http://alumni.media.mit.edu/~jorkin/goap.html



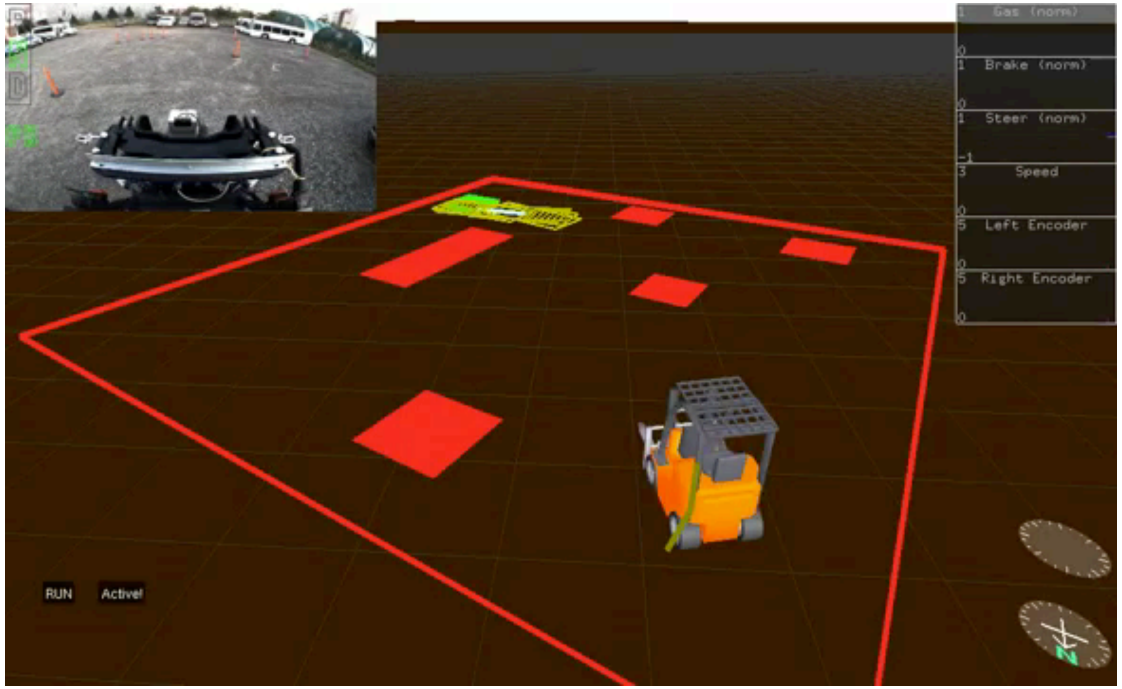
https://www.youtube.com/watch?v=xq2i4m5d_I8

Motion planning

Sampling-based motion planning

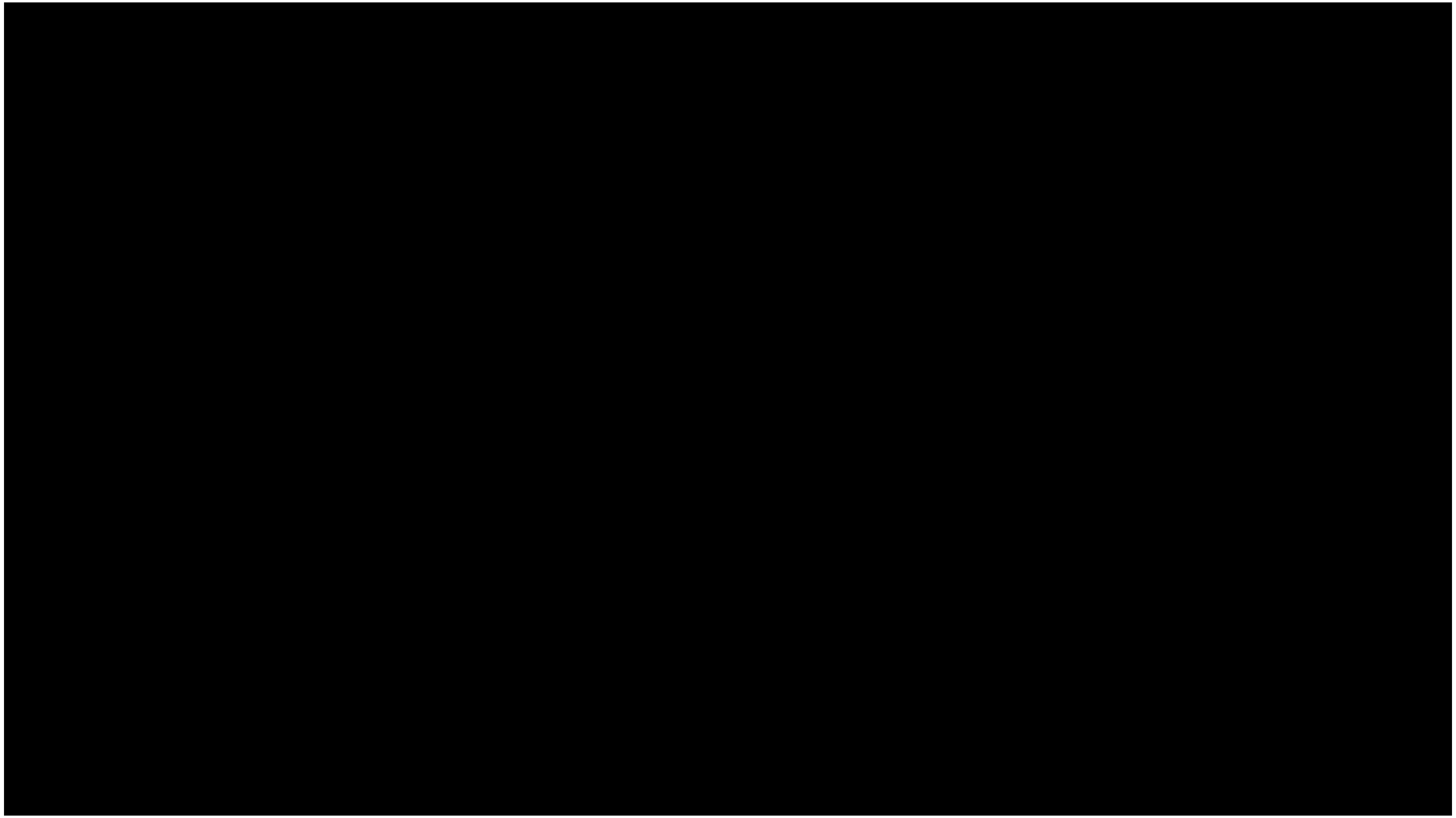


https://www.youtube.com/watch?v=LKL5qRBiJaM



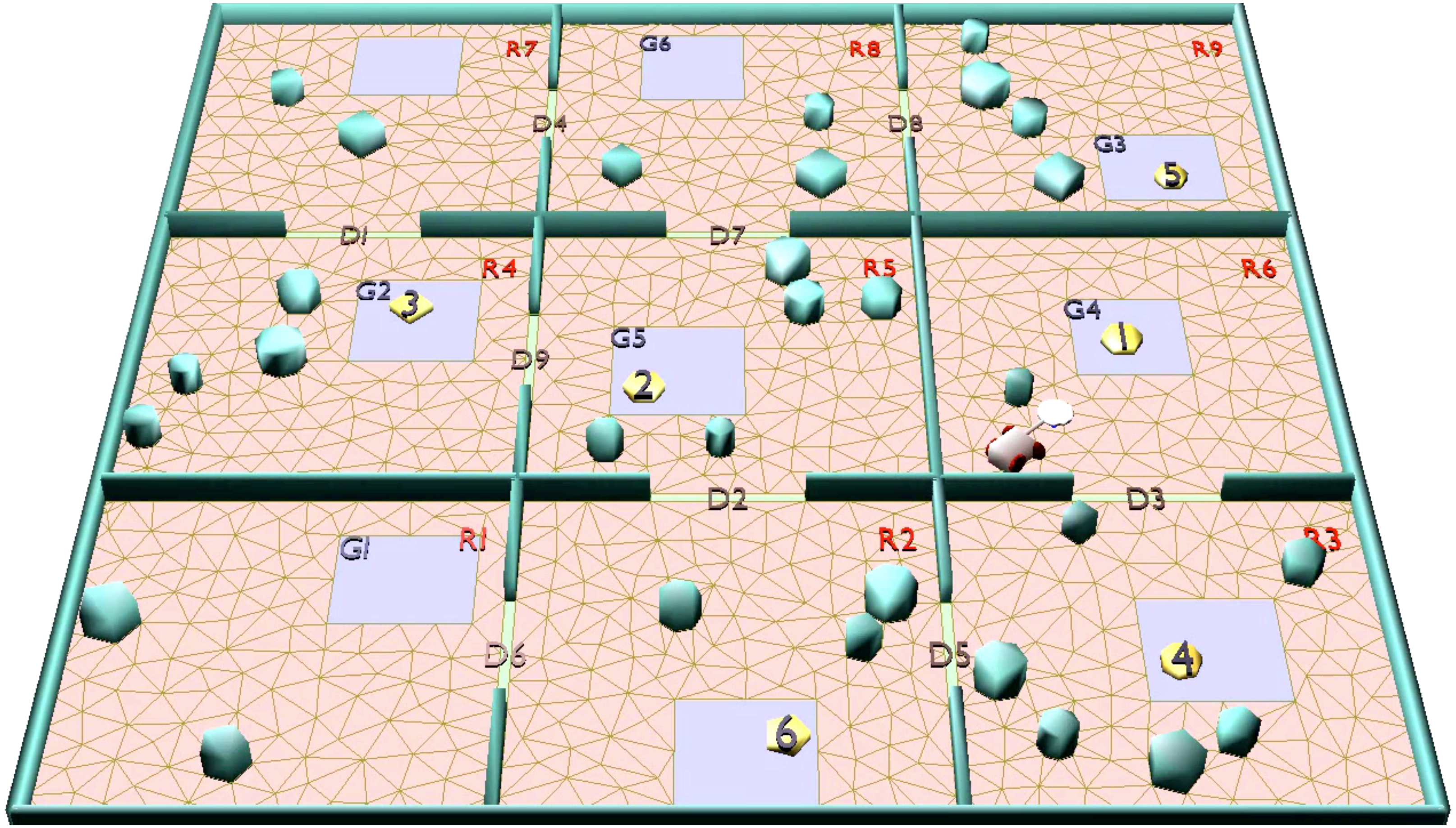
https://www.youtube.com/watch?v=6Pngam882hM

Robot motion planning: RRT vs. RRT*



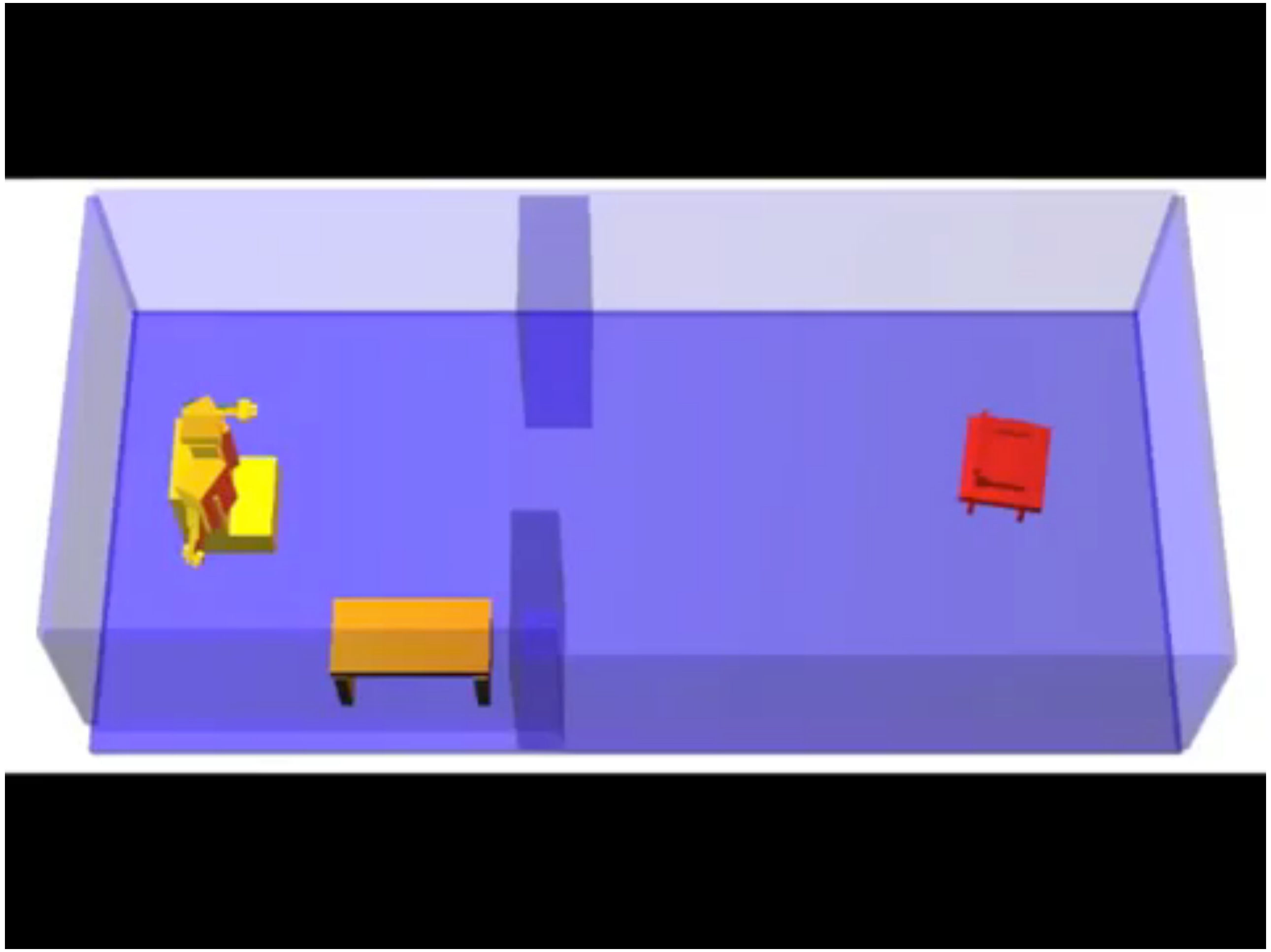
https://www.youtube.com/watch?v=ag-txw4KUgo

Motion planning with PDDL



https://www.youtube.com/watch?v=kb79tR5bmlE

Integrated task and motion planning



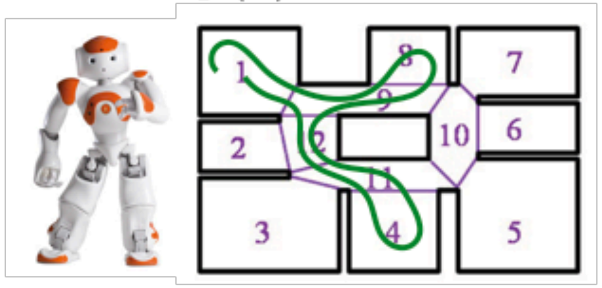
http://lis.csail.mit.edu/

Temporal Logic Planning

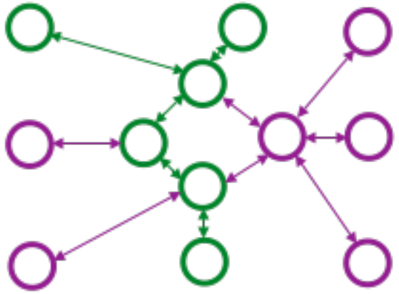
System model

$$\dot{p}(t) = u(t) \qquad p(t) \in P \subseteq \mathbb{R}^2 \qquad u(t) \in U \subseteq \mathbb{R}^2$$

$$p(0) = P_1$$



Discrete model



Behavior specification

Periodically visit P_1, P_4, P_8
and never enter P_{10}

Linear Temporal Logic (LTL) formula

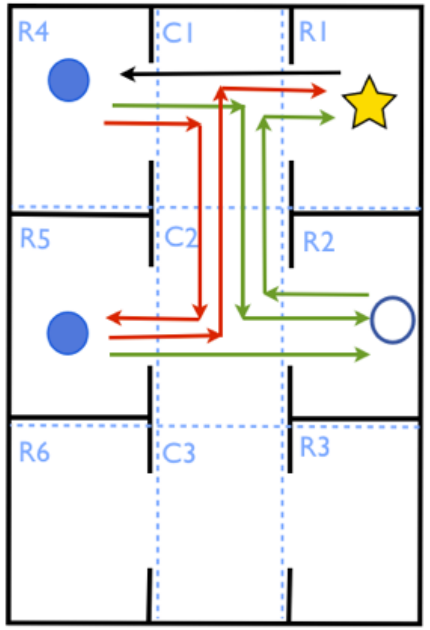
$$\mathcal{GF}\, P_1 \;\wedge\; \mathcal{GF}\, P_4 \;\wedge\; \mathcal{GF}\, P_8 \;\wedge\; \mathcal{G}\,\neg P_{10}$$

[KFP09] Kress-Gazit, Fainekos, Pappas, "Temporal-Logic-Based Reactive Mission and Motion Planning," *TRO*, 2009.

X

Maximally satisfying temporal logic planning

$$\mathsf{GF}((R_4 \wedge grab \vee R_5 \wedge grab) \wedge \mathsf{F}(R_2 \wedge drop)) \wedge \mathsf{GF}\, lighten.$$



Joint work with A. Marzinotto, D. Dimarogonas, D. Kragic

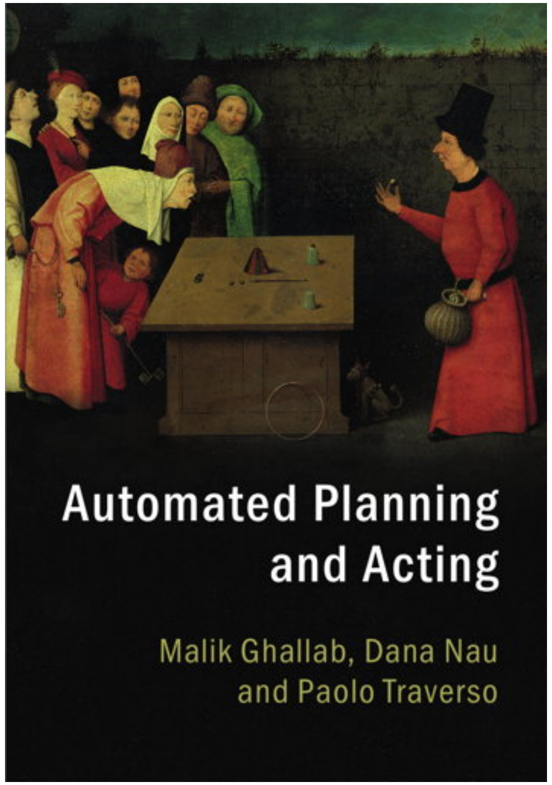
Assignment 3:

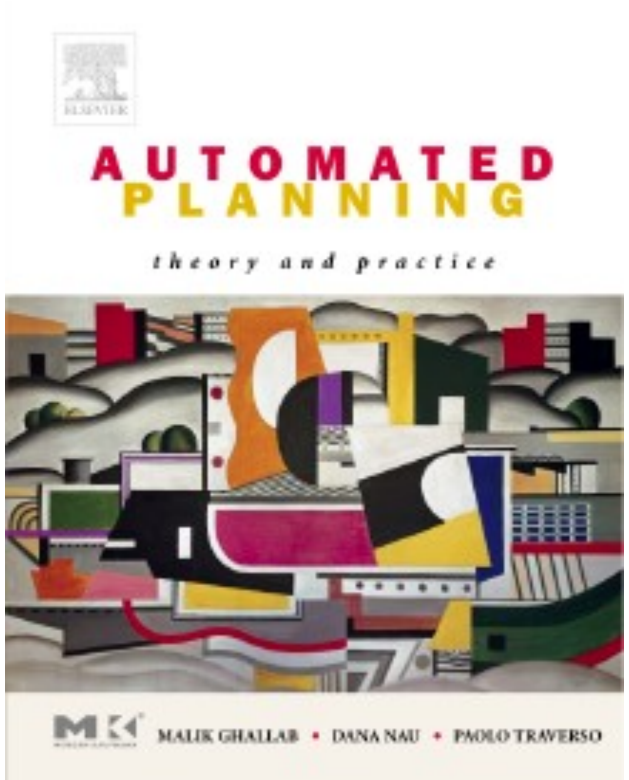
A3. PP Pen&paper variant	A3.PR Project variant
Individual	In teams
E-B level	A level

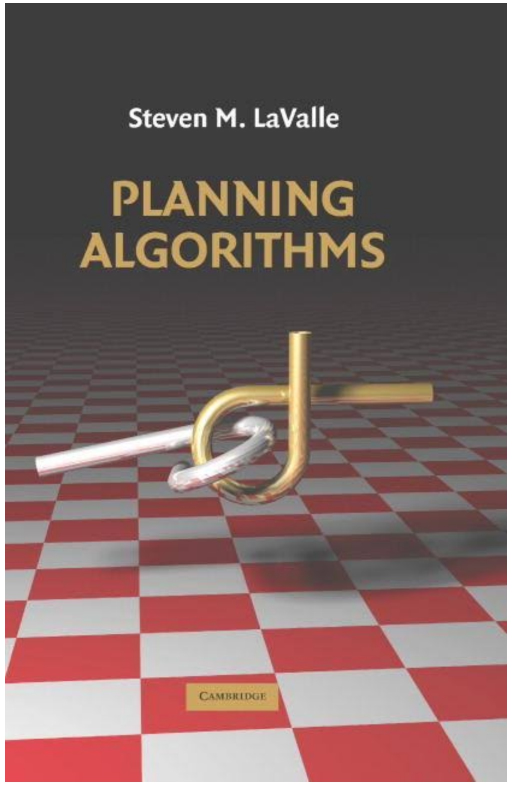
Further sources of Inspiration

- For state-of-the art challenges in planning
  - http://icaps16.icaps-conference.org
- For your ”case studies”
  - https://helios.hud.ac.uk/scommv/IPC-14/domains_sequential.html

Further reading on principles







Further interesting links

- On state-of-the art approaches in planning
  - http://icaps-conference.org/index.php/Main/Competitions
- On applications of AI planning
  - http://icaps16.icaps-conference.org/proceedings/spark16.pdf
- NASA planning and scheduling research
  - https://ti.arc.nasa.gov/tech/asr/planning-and-scheduling/
  - https://github.com/nasa/europa/wiki
- On planning in robotics
  - Go to a program of any major conference (ICRA, IROS), look for task and motion planning tracks, http://drops.dagstuhl.de/opus/volltexte/2017/7245/pdf/dagrep_v007_i001_p032_s17031.pdf