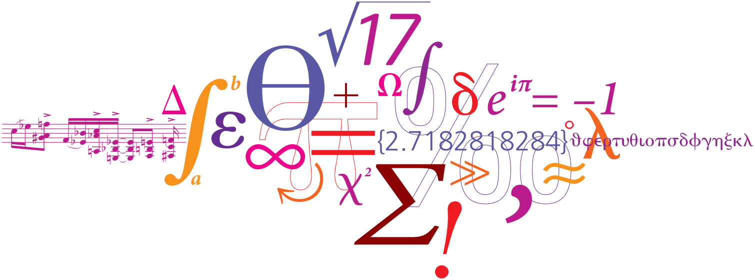
From Classical to Epistemic Planning

Thomas Bolander, DTU Compute, Technical University of Denmark



DTU Compute
Department of Applied Mathematics and Computer Science

Running example: Birthday present

Automated planning: Computing plans (sequences of actions) leading to some desired goal.

Planning example. A father ordered a present for his daughter's birthday. It is now at the post office. His **goal** is to give it to her on her birthday the following day.

Epistemic planning example. The father might be uncertain about which post office the parcel is at. He might also want to ensure that his daughter doesn't get to know about the parcel.

Structure of this talk

From classical planning to planning based on dynamic epistemic logic (DEL):

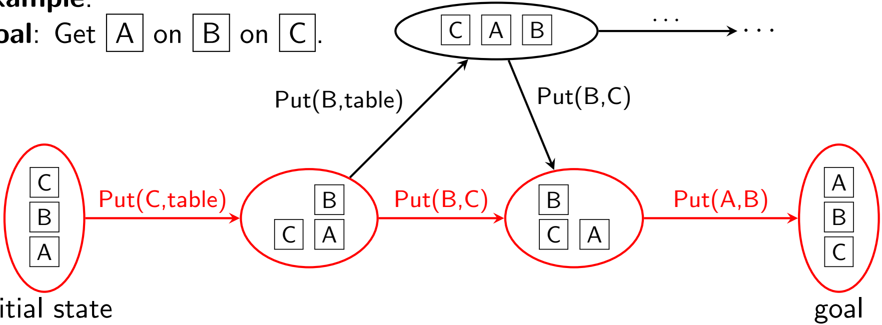
1. Classical planning domains and planning tasks.
2. STRIPS planning.
3. Propositional planning.
4. Belief states, partial observability and conditional actions.
5. (Dynamic) epistemic logic (DEL).
6. Epistemic planning tasks.
7. Types of epistemic planning tasks and types of solutions.
8. Complexity issues.
9. Alternative approaches to epistemic planning.

Automated planning

Automated planning (or, simply, **planning**):

- Aims at generating **plans (sequences of actions)** leading to desired outcomes.
- More precisely: Given a **goal formula**, an **initial state** and some **possible actions**, an **automated planner** outputs a plan that leads from the initial state to a state satisfying the goal formula.

Example.
Goal: Get $\boxed{\text{A}}$ on $\boxed{\text{B}}$ on $\boxed{\text{C}}$.



Birthday example (non-epistemic version)

- **Initial state**:
  - Father at home.
  - Present at post office.
  - Present not wrapped.
- **Goal**:
  - Father at home.
  - Father has present.
  - Present wrapped.
- **Actions**:
  - Go from location *from* to location *to*.
  - Pick up object *obj* at location *from*.
  - Wrap object *obj*.

To formally reason about such **planning tasks**, we need an appropriate formalism. The must basic approach is to use state-transition systems...

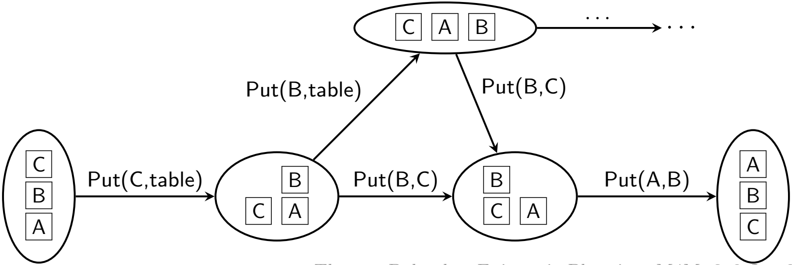
State-transition systems

Definition ([Ghallab et al., 2004])

A **(restricted) state-transition system** (also called a a **classical planning domain** or simply a **state space**) is $\Sigma = (S, A, \gamma)$, where:

- S is a finite or recursively enumerable set of **states**.
- A is a finite or recursively enumerate set of **actions**.
- $\gamma : S \times A \hookrightarrow S$ is a computable partial **state-transition function**.

When $\gamma(s, a)$ is defined, a is said to be **applicable** in S.
When $\pi = a_1; \cdots ; a_n$ let $\gamma(s, \pi) := \gamma(\cdots\gamma(\gamma(s, a_1), a_2), \ldots, a_n)$.



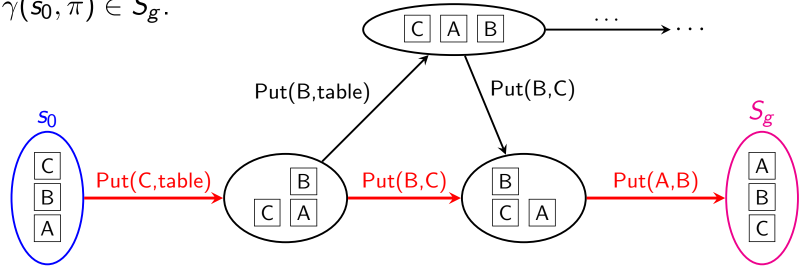
Classical planning tasks

Definition ([Ghallab et al., 2004])

A **classical planning task** is a triple (Σ, s_0, S_g), where

- $\Sigma = (S, A, \gamma)$ is restricted state-transition system (a classical planning domain).
- $s_0 \in S$ is the **initial state**.
- $S_g \subseteq S$ is the set of **goal states**.

A **solution** to a classical planning task $((S, A, \gamma), s_0, S_g)$ is a finite sequence of actions (a **plan**) $\pi = a_1; \cdots ; a_n$ from A such that $\gamma(s_0, \pi) \in S_g$.

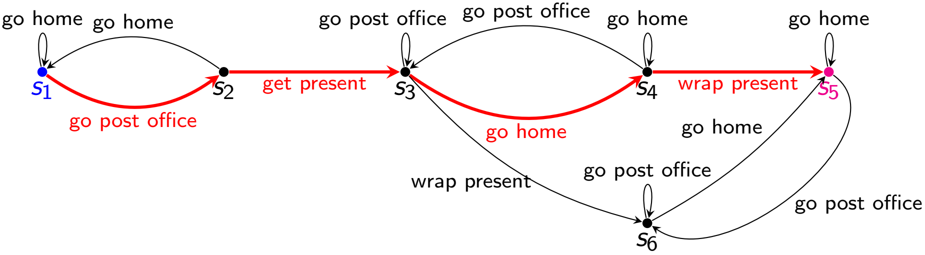


Classical planning task example: Birthday present

The birthday present example can be represented as the classical planning task $((S, A, \gamma), s_0, S_g)$ with

- $S = \{s_1, s_2, s_3, s_4, s_5, s_6\}$.
- $A = \{$go post office, go home, get present, wrap present$\}$.
- γ as given below.
- $s_0 = s_1$.
- $S_g = \{s_5\}$.

A solution is $\pi =$ go post office; get present; go home; wrap present.



Weaknesses of planning via state-transition systems

- **Unmanageable state space sizes**. With n parcels to take home from the post office, the state space would be of size $\geq 2^n$. But shortest solution still linear in n.
- **No structure on states and actions to guide search**. To avoid computing the entire state space, we need *heuristics* (e.g. number of parcels still at post office = *goal count heuristics*). To compute these automatically, we need structure on states and actions...

$43 * 10^{18}$ states

Logical structure on states/actions: STRIPS

STRIPS [Fikes and Nilsson, 1971]: The classical language for describing states and actions in the field of automated planning.

- **STRIPS state**: Set of ground atoms of a function-free first-order language $\mathcal{L}$.
- **STRIPS action**: Ground instance of an **action schema** specified by **name**, **precondition** and **effect**. Precondition and effects: conjunctions of literals of $\mathcal{L}$.

Example (birthday pres.). State s_0 and action schemas Go and PickUp:

$$\begin{aligned} s_0 = \quad & \{\mathrm{At}(\mathrm{Father}, \mathrm{Home}), \mathrm{At}(\mathrm{Present}, \mathrm{PostOffice}), \mathrm{IsAgt}(\mathrm{Father}), \\ & \mathrm{IsLoc}(\mathrm{Home}), \mathrm{IsLoc}(\mathrm{PostOffice}), \mathrm{IsObj}(\mathrm{Present})\} \end{aligned}$$

$\textsc{Action}$: $\mathrm{Go}(agt, from, to)$
$\textsc{Precond}$: $\mathrm{At}(agt, from) \wedge \mathrm{IsAgt}(agt) \wedge \mathrm{IsLoc}(from) \wedge \mathrm{IsLoc}(to)$
$\textsc{Effect}$: $\mathrm{At}(agt, to) \wedge \neg \mathrm{At}(agt, from)$

$\textsc{Action}$: $\mathrm{PickUp}(agt, obj, from)$
$\textsc{Precond}$: $\mathrm{At}(agt, from) \wedge \mathrm{At}(obj, from) \wedge \neg \mathrm{Has}(agt, obj) \wedge \mathrm{IsAgt}(agt) \wedge \mathrm{IsObj}(obj) \wedge \mathrm{IsLoc}(from)$
$\textsc{Effect}$: $\mathrm{Has}(agt, obj) \wedge \neg \mathrm{At}(obj, from)$

Example: Application of a STRIPS action in a state

Recall action schema for Go:

$\textsc{Action}: \mathsf{Go}(\mathit{agt}, \mathit{from}, \mathit{to})$
$\textsc{Precond}: \mathsf{At}(\mathit{agt}, \mathit{from}) \wedge \mathsf{IsAgt}(\mathit{agt}) \wedge \mathsf{IsLoc}(\mathit{from}) \wedge \mathsf{IsLoc}(\mathit{to})$
$\textsc{Effect}: \mathsf{At}(\mathit{agt}, \mathit{to}) \wedge \neg \mathsf{At}(\mathit{agt}, \mathit{from})$

Example of ground instance (= action):

$\textsc{Action}: \mathsf{Go}(\mathsf{Father}, \mathsf{Home}, \mathsf{PostOffice})$
$\textsc{Precond}: \mathsf{At}(\mathsf{Father}, \mathsf{Home}) \wedge \mathsf{IsAgt}(\mathsf{Father}) \wedge \mathsf{IsLoc}(\mathsf{Home}) \wedge \mathsf{IsLoc}(\mathsf{PostOffice})$
$\textsc{Effect}: \mathsf{At}(\mathsf{Father}, \mathsf{PostOffice}) \wedge \neg \mathsf{At}(\mathsf{Father}, \mathsf{Home})$

Then:

At(Father, Home), At(Present, PostOffice), IsAgt(Father), IsLoc(Home), IsLoc(PostOffice), IsObj(Present) $\quad s_0$

$\downarrow$ Go(Father, Home, PostOffice)

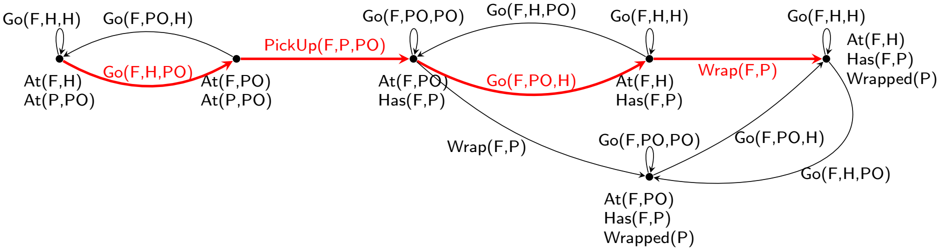
At(Father, PostOffice), At(Present, PostOffice), IsAgt(Father), IsLoc(Home), IsLoc(PostOffice), IsObj(Present) $\quad s_1$

State-transition system induced by action schemas

Any finite set of STRIPS action schemas A **induce** a state-transition system $\Sigma = (S, A', \gamma)$ by:

- $S = 2^P$, where P is the set of ground atoms of $\mathcal{L}$.
- $A' = \{$all ground instances of action schemas in $A\}$.
- $\gamma(s, a) =$

$$\begin{cases} (s - \{\phi \mid \neg\phi \text{ is a neg. literal of } \textsc{Effect}(a)) \cup & \\ \quad \{\phi \mid \phi \text{ is a pos. literal of } \textsc{Effect}(a)\} & \text{if } s \models \textsc{Precond}(a) \\ \text{undefined} & \text{otherwise} \end{cases}$$



Compactness of STRIPS representation

STRIPS representation is compact: We can add any number of parcels, locations and agents without any change in the size of the STRIPS planning domain (no change to the action schemas). But the induced state-transition system is exponential in each of these.

STRIPS planning tasks and solutions

Definition. A **STRIPS planning task** on a function-free first-order language $\mathcal{L}$ is (A, s_0, ϕ_g) where

- A, the set of **actions**, is a set of STRIPS action schemas over $\mathcal{L}$.
- s_0, the **initial state**, is a set of ground atoms over $\mathcal{L}$.
- ϕ_g, the **goal formula**, is a conjunction of ground literals over $\mathcal{L}$.

Any STRIPS planning task (A, s_0, ϕ_g) **induces** a classical planning task (Σ, s_0, S_g) by letting Σ be the state-transition system induced by A and letting $S_g = \{s \in S \mid s \models \phi_g\}$. A **solution** to a STRIPS planning task is then a solution to the induced classical planning task.

Example (birthday present). STRIPS planning task (A, s_0, ϕ_g) where

- A contains action schemas for Go, PickUp and Wrap.
- s_0 is the earlier shown initial state.
- $\phi_g = \mathrm{At(Father, Home)} \wedge \mathrm{Has(Father, Present)} \wedge \mathrm{Wrapped(Present)}$.

Solution $\pi = \mathrm{Go(F, H, PO); PickUp(F, P, PO); Go(F, PO, H); Wrap(F, P)}$.

Propositional planning tasks

Definition. A **propositional planning task** on a finite set of atomic propositions P is (A, s_0, ϕ_g), where

- A is a finite set of **actions** $a = \langle pre(a), post(a) \rangle$. $pre(a)$, the **precondition** of a, and $post(a)$, the **postcondition** of a, are conjunctions of propositional literals over P.
- $s_0 \subseteq P$ is the **initial state** (a propositional state over P).
- ϕ_g is the **goal formula**, a propositional formula over P.

A propositional planning task (A, s_0, ϕ_g) on P **induces** a classical planning task $((S, A, \gamma), s_0, S_g)$ in the expected way:

- $S = 2^P$ (all propositional states over P)
- $\gamma(s, a) =$
$$\begin{cases} (s - \{p \mid \neg p \text{ is a negative literal of } post(a)) \cup \\ \quad \{p \mid p \text{ is a positive literal of } post(a)\} & \text{if } s \models pre(a) \\ \text{undefined} & \text{otherwise} \end{cases}$$
- $S_g = \{s \in S \mid s \models \phi_g\}$.

A **solution** to a propositional planning task is any solution to the induced classical planning task.

Grounding: Propositional planning tasks induced by STRIPS planning tasks

For any function-free first-order language $\mathcal{L}$, let $P_{\mathcal{L}}$ denote the set of ground atoms of $\mathcal{L}$.

Any quantifier-free ground formula of $\mathcal{L}$ is then at the same time a formula of propositional logic over $P_{\mathcal{L}}$.

Any STRIPS planning task (A, s_0, ϕ_g) on $\mathcal{L}$ **induces** a propositional planning task (A', s_0, ϕ_g) on $P_{\mathcal{L}}$ by simply letting:

$$A' = \{\langle \textsc{Precond}(a), \textsc{Effect}(a) \rangle \mid a \text{ is a ground instance of an action schema in } A\}.$$

It is easy to show that the STRIPS planning task (A, s_0, ϕ_g) and its induced propositional planning task (A', s_0, ϕ_g) both induce the same classical planning task.

$$\begin{tikzcd}
\text{STRIPS plann. task} \arrow[rr] \arrow[dr] & & \text{classical plann. task} \\
& \text{propositional plann. task} \arrow[ur] &
\end{tikzcd}$$

Propositional plann. task example: Birthday present

The birthday present example can be represented as the propositional planning task (A, s_0, ϕ_g) where (omitting the Wrap action):

$Agt \supseteq \{\text{Father}\}$ is a set of agent names, $Loc \supseteq \{\text{Home}, \text{PostOffice}\}$ is a set of locations, and $Obj \supseteq \{\text{Present}\}$ is a set of objects.

$$\begin{aligned} A = \; & \{\text{Go}(agt, from, to) \mid agt \in Agt \;\&\; from, to \in Loc\} \cup \\ & \{\text{PickUp}(agt, obj, from) \mid agt \in Agt \;\&\; obj \in Obj \;\&\; from \in Loc\} \end{aligned}$$

where, for all $agt \in Agt$, all $from, to \in Loc$ and all $obj \in Obj$,

$$\text{Go}(agt, from, to) = \langle \text{At}(agt, from), \text{At}(agt, to) \wedge \neg \text{At}(agt, from) \rangle$$

$$\text{PickUp}(agt, obj, from) = \\ \langle \text{At}(agt, from) \wedge \text{At}(obj, from) \wedge \neg \text{Has}(agt, obj), \text{Has}(agt, obj) \rangle$$

$s_0 = \{\text{At}(\text{Father}, \text{Home}), \text{At}(\text{Present}, \text{PostOffice})\}$.

$\phi_g = \text{At}(\text{Father}, \text{Home}) \wedge \text{Has}(\text{Father}, \text{Present}) \wedge \text{Wrapped}(\text{Present})$.

Plan existence problem in propositional planning

- Propositional planning tasks (A, s_0, ϕ_g) can be **exponentially more succinct** than their induced classical planning tasks/state spaces $((S, A, \gamma), s_0, S_g)$: Most often A is polynomial in P and S exponential in P.
- We want to do planning directly based on the compact propositional planning task descriptions.

Definition. The **plan existence problem in propositional planning** is the following decision problem: "Given a propositional planning task (A, s_0, ϕ_g), does it have a solution?"

Theorem [Bylander, 1994]. *The plan existence problem in propositional planning is PSPACE-complete.*

- Note that this is the complexity measured in terms of the **succinct task description**.
- In planning based on temporal epistemic logics, e.g. ATEL [van der Hoek and Wooldridge, 2002], planning is measured in size of **state space**.

Belief states

?

$$Loc = \{\text{Home}, \text{PostOffice1}, \text{PostOffice2}\}.$$

The father doesn't know in which post office the parcel is. Uncertainty represented by **belief states**: sets of propositional states.

The initial belief state of the father:

$$\begin{aligned} s_0 = \ & \{\{\text{At}(\text{Father}, \text{Home}), \text{At}(\text{Present}, \text{PostOffice1})\}, \\ & \ \{\text{At}(\text{Father}, \text{Home}), \text{At}(\text{Present}, \text{PostOffice2})\}\}. \end{aligned}$$

In line with modal logic, we call the elements of belief states **(possible) worlds**.

Truth in belief states and internal perspective

For belief states s and prop. formulas ϕ:

$$s \models \phi \quad \stackrel{def}{\Leftrightarrow} \quad \phi \text{ is true in all states of } s$$

Example (birthday present). Let again

$$\begin{aligned} s_0 = \; & \{\{\mathrm{At(Father, Home)}, \mathrm{At(Present, PostOffice1)}\}, \\ & \{\mathrm{At(Father, Home)}, \mathrm{At(Present, PostOffice2)}\}\}. \end{aligned}$$

Then

(1) $s_0 \models \mathrm{At(Father, Home)}$

(2) $s_0 \not\models \mathrm{At(Present, PostOffice1)}$

(3) $s_0 \not\models \mathrm{At(Present, PostOffice2)}$

(4) $s_0 \models \mathrm{At(Present, PostOffice1)} \vee \mathrm{At(Present, PostOffice2)}$.

s_0 represents the father's **internal perspective** on the initial state: He can verify (knows) that he is home (1) and can verify (knows) that the present is in PostOffice1 or PostOffice2 (4), but doesn't know which (2–3).

Planning under partial observability and conditional actions

Planning in the space of belief states is called **planning under partial observability**.

- Actions also need to be represented from the internal perspective of the planning agent.
- In s_0, the father doesn't know whether attempting to pick up the parcel at PostOffice1 will be succesful.
- He has two represent multiple possible outcomes of executing the action. It is a **conditional action**.
- **Conditional actions** can be represented by sets of pairs $\langle pre(a), post(a)\rangle$ (called **events** in line with dynamic epist. logic).

Example.

$$
\begin{array}{l}
\text{TryPickUp}(agt, obj, from) = \{ \\
\quad \langle \text{At}(agt, from) \wedge \text{At}(obj, from) \wedge \neg\text{Has}(agt, obj), \text{Has}(agt, obj) \wedge \neg\text{At}(obj, from)\rangle, \\
\quad \langle \text{At}(agt, from) \wedge \neg\text{At}(obj, from), \top\rangle \\
\}
\end{array}
$$

From propositional states and actions to belief states and conditional actions

Note how we got from propositional states to belief states:

propositional state	**epistemic state**
propositional valuation	set of such valuations

We applied the **exact same trick** to actions:

propositional action	**conditional action**
pair $\langle pre, post \rangle$ where *pre* is propositional formula and *post* is conjunction of propositional literals	set of such pairs $\langle pre, post \rangle$

Generalised transition function

Given a belief state (set of worlds) s and a conditional action (set of events) a, we can define a generalised transition function by:

$$\gamma(s, a) = \{\gamma(w, e) \mid w \in s, e \in a, w \models pre(e)\}.$$

{At(Father, Home), At(Present, PostOffice1)},
{At(Father, Home), At(Present, PostOffice2)} s_0

↓ Go(Father, Home, PostOffice1)

{At(Father, PostOffice1), At(Present, PostOffice1)},
{At(Father, PostOffice1), At(Present, PostOffice2)} s_1

↓ TryPickUp(Father, Present, PostOffice1)

{At(Father, PostOffice1), Has(Father, Present)},
{At(Father, PostOffice1), At(Present, PostOffice2)} s_2

What is the problem in the belief state representation above?

Run time versus plan time uncertainty

Consider the partial state-transition system from before:

s_0: {At(Father, Home), At(Present, PostOffice1)},
{At(Father, Home), At(Present, PostOffice2)}

↓ Go(Father, Home, PostOffice1)

s_1: {At(Father, PostOffice1), At(Present, PostOffice1)},
{At(Father, PostOffice1), At(Present, PostOffice2)}

↓ TryPickUp(Father, Present, PostOffice1)

s_2: {At(Father, PostOffice1), Has(Father, Present)},
{At(Father, PostOffice1), At(Present, PostOffice2)}

In s_0, the father has **run time uncertainty** about which of the two worlds is the actual: Even at execution time he can not distinguish.

In s_2, the father should only have **plan time uncertainty**: At plan time he can not distinguish, but at execution time he can.

We need to formally be able to distinguish...

Models of observability

We need a way to model observability: which worlds and events are (run time) distinguishable by the planning agent.

Standard approaches in the planning literature:

(1) Observability is a **static partition** on the set of possible worlds (e.g. [Ghallab et al., 2004]). Worlds in the same partition are indistinguishable. **Example**: I always see my cards, but never your cards.

(2) Each possible world determines a **percept** (or **observation**) (e.g. [Russell and Norvig, 1995, Ghallab et al., 2004]). Worlds with identical percepts are indistinguishable. Equivalent to (1).

Not sufficient for our purposes. Why not?

We need a more expressive framework for clearly separating run time vs plan time indistinguishable worlds and events. We move to (dynamic) epistemic logic...

Epistemic language and epistemic models

The **epistemic language** on propositions P and agents $\mathcal{A}$, denoted $\mathcal{L}_{\mathsf{KC}}(P, \mathcal{A})$ (or simply $\mathcal{L}_{\mathsf{KC}}$), is generated by the following BNF:

$$\phi ::= \top \mid \bot \mid p \mid \neg\phi \mid \phi \wedge \phi \mid K_i\phi \mid C\phi,$$

where $p \in P$ and $i \in \mathcal{A}$.

Definition. An **epistemic model** on $P, \mathcal{A}$ is $\mathcal{M} = (W, (\sim_i)_{i \in \mathcal{A}}, L)$ where

- The **domain** W is a non-empty finite set of **worlds**.
- $\sim_i \subseteq W \times W$ is an equivalence relation called the **indistinguishability relation** for agent $i \in \mathcal{A}$.
- $L : W \to 2^P$ is a **labelling function** assigning a propositional valuation (a set of propositions) to each world.

Local and global (epistemic) states

Epistemic state (or simply **state**): A pair $(\mathcal{M}, W_d)$ for some set of **designated worlds** $W_d \subseteq W$ (will be denoted $\odot$).

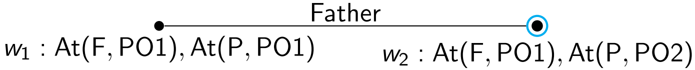
Global state: A state $(\mathcal{M}, W_d)$ with $W_d = \{w\}$ for some w called the **actual world**.

Local state for agent i: A state $(\mathcal{M}, W_d)$ where W_d is closed under $\sim_i$.

Associated local state of agent i of state $s = (\mathcal{M}, W_d)$:
$s^i \stackrel{def}{=} (\mathcal{M}, \{v \mid v \sim_i w \text{ and } w \in W_d\})$.

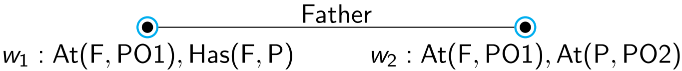
Example. Global state representing situation after Go(F, PO1) from initial state (with parcel at PO2):

$s_1 =$



Associated local state of Father (internal representation of father):

$s_1^{\text{Father}} =$



Run time vs plan time indistinguishability

Before attempting to pick up the parcel at PostOffice1:

$$s_1^{\text{Father}} = \begin{tikzcd} w_1 : \mathrm{At}(F, PO1), \mathrm{At}(P, PO1) \arrow[r, dash, "\text{Father}"] & w_2 : \mathrm{At}(F, PO1), \mathrm{At}(P, PO2) \end{tikzcd}$$

Plan time representation of the result of executing TryPickUp(Father, Present, PO1):

$$s_2^{\text{Father}} = \quad w_1 : \mathrm{At}(F, PO1), \textcolor{blue}{\mathrm{Has}(F, P)} \qquad w_2 : \mathrm{At}(F, PO1), \mathrm{At}(P, PO2)$$

Let $s = (\mathcal{M}, W_d)$ be local state of agent i and $w_1, w_2 \in W_d$. Worlds w_1 and w_2 are **run time indistinguishable** to agent i if $w_1 \sim_i w_2$. Otherwise **plan time indistinguishable** (or **run time distinguishable**).

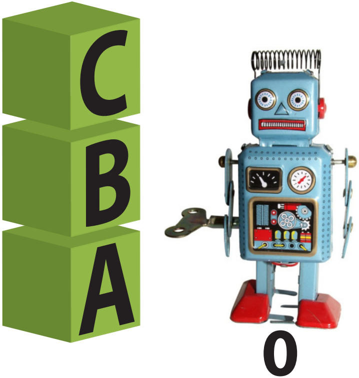
Epistemic states induced by belief states

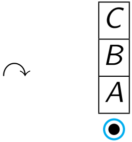
Any belief state

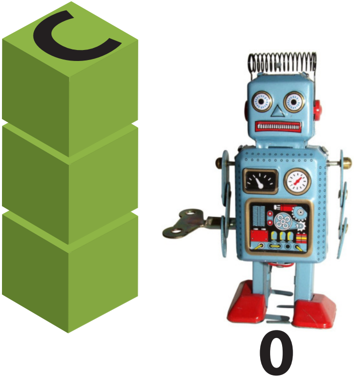
$$B = \{w_1, \ldots, w_n\}$$

canonically **induces** an epistemic state $((W, \sim, L), W_d)$ with

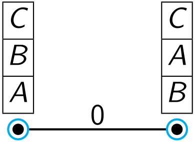
- $W = \{w'_1, \ldots, w'_n\}$.
- $\sim = W \times W$.
- $L(w'_i) = w_i$ for all $i = 1, \ldots, n$.
- $W_d = W$.

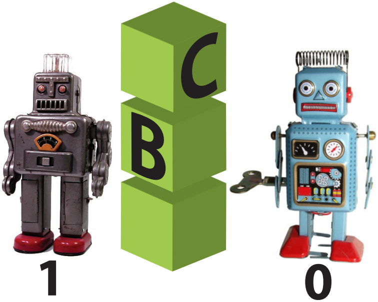

C
B
A
0


C
B
A

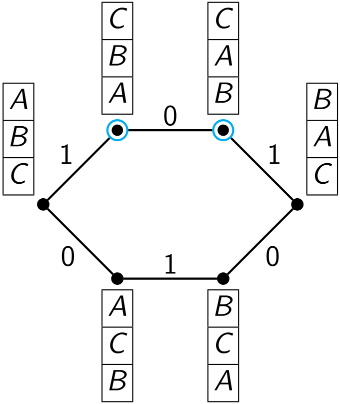


$\curvearrowright$





$\curvearrowright$



Truth in epistemic states

Let $(\mathcal{M}, W_d)$ be a state on $P, \mathcal{A}$ with $\mathcal{M} = (W, (\sim_i)_{i \in \mathcal{A}}, L)$. For $i \in \mathcal{A}$, $p \in P$ and $\phi, \psi \in \mathcal{L}_{\mathrm{KC}}(P, \mathcal{A})$, we define truth as follows:

$$
\begin{array}{lll}
(\mathcal{M}, W_d) \models \phi & \text{iff} & (\mathcal{M}, w) \models \phi \text{ for all } w \in W_d \\
(\mathcal{M}, w) \models p & \text{iff} & p \in L(w) \\
(\mathcal{M}, w) \models \neg\phi & \text{iff} & \mathcal{M}, w \not\models \phi \\
(\mathcal{M}, w) \models \phi \wedge \psi & \text{iff} & \mathcal{M}, w \models \phi \text{ and } \mathcal{M}, w \models \psi \\
(\mathcal{M}, w) \models K_i\phi & \text{iff} & (\mathcal{M}, v) \models \phi \text{ for all } v \sim_i w \\
(\mathcal{M}, w) \models C\phi & \text{iff} & (\mathcal{M}, v) \models \phi \text{ for all } v \sim^* w
\end{array}
$$

where $\sim^*$ is the transitive closure of $\bigcup_{i \in \mathcal{A}} \sim_i$.

Example. Let

$s_2^{\text{Father}} =$ $w_1 : \mathrm{At}(\mathrm{F}, \mathrm{PO1}), \mathrm{Has}(\mathrm{F}, \mathrm{P})$ $\quad$ $w_2 : \mathrm{At}(\mathrm{F}, \mathrm{PO1}), \mathrm{At}(\mathrm{P}, \mathrm{PO2})$

In this state, the father knows whether the parcel is at PostOffice1:

$$s_2^{\text{Father}} \models K_{\text{Father}}\mathrm{At}(\mathrm{P}, \mathrm{PO2}) \vee K_{\text{Father}}\neg\mathrm{At}(\mathrm{P}, \mathrm{PO2})$$

Planning with multiple agents and epistemic goals

The generalisation from belief states to epistemic states: multi-agent planning, epistemic goals.

Example. The father might want to make sure his daughter doesn't come to know about the present (it's meant to be a surprise):

$$\begin{aligned}\phi_g = \quad & \mathsf{At}(\mathsf{Father}, \mathsf{Home}) \wedge \mathsf{Has}(\mathsf{Father}, \mathsf{Present}) \wedge \\ & \mathsf{Wrapped}(\mathsf{Present}) \wedge \neg K_{\mathsf{Daughter}} \mathsf{Has}(\mathsf{Father}, \mathsf{Present}).\end{aligned}$$

Action models

Definition. An **action model** on $P, \mathcal{A}$ is $\mathcal{E} = (E, (\sim_i)_{i \in \mathcal{A}}, pre, post)$ where

- The **domain** E is a non-empty finite set of **events**.
- $\sim_i \subseteq E \times E$ is an equivalence relation called the **indistinguishability relation** for agent $i \in \mathcal{A}$.
- $pre : E \to \mathcal{L}_{\mathsf{KC}}(P, \mathcal{A})$ assigns a **precondition** to each event.
- $post : E \to \mathcal{L}_{\mathsf{KC}}(P, \mathcal{A})$ assigns a **postcondition** to each event. For all $e \in E$, $post(e)$ is a conjunction of literals over P.

Epistemic action (or simply **action**): A pair $(\mathcal{E}, E_d)$ for some set $E_d \subseteq E$ of **designated events** (will be denoted $\odot$).

Global action: An action $(\mathcal{E}, E_d)$ with $E_d = \{e\}$ for some e called the **actual event**.

Local action for agent i: An action $(\mathcal{E}, E_d)$ where E_d is closed under $\sim_i$.

Associated local action of agent i of action $a = (\mathcal{E}, E_d)$:
$a^i \overset{def}{=} (\mathcal{E}, \{f \in E \mid f \sim_i e \text{ for some } e \in E_d\})$.

Epistemic actions induced by conditional actions

Any conditional action

$$a = \{\langle pre(a_1), post(a_1)\rangle, \dots, \langle pre(a_n), post(a_n)\rangle\}$$

canonically **induces** an epistemic action $((E, \sim, pre, post), E_d)$ with

- $E = \{a'_1, \dots, a'_n\}$.
- $\sim = E \times E$.
- $pre(a'_i) = pre(a_i)$ for all $i = 1, \dots, n$.
- $post(a'_i) = post(a_i)$ for all $i = 1, \dots, n$.
- $E_d = E$.

Epistemic action example: Birthday present

We can now finally, using action models, give a satisfactory formal representation of the TryPickUp action.

$\text{TryPickUp}(agt, obj, from) =$

$$e_1 : \langle \text{At}(agt, from) \wedge \text{At}(obj, from) \wedge \neg\text{Has}(agt, obj), \text{Has}(agt, obj) \wedge \neg\text{At}(obj, from)\rangle$$

$$e_2 : \langle \text{At}(agt, from) \wedge \neg\text{At}(obj, from), \top\rangle$$

Note that there is no edge between e_1 and e_2: they are run time distinguishable (using the same definition as for epistemic states).

At run time the father will observe whether the action is succesful (e_1) or not (e_2).

Product update

State-transition function of dynamic epistemic logic: *product update* denoted by an infix $\otimes$ symbol. So $\gamma(s, a) \stackrel{def}{=} s \otimes a$.

Definition. Let a state $s = (\mathcal{M}, W_d)$ and an action $a = (\mathcal{E}, E_d)$ be given with $\mathcal{M} = (W, (\sim_i)_{i \in \mathcal{A}}, L)$ and $\mathcal{E} = (E, (\sim_i)_{i \in \mathcal{A}}, pre, post)$. Then the **product update** of s with a is $s \otimes a = ((W', (\sim'_i)_{i \in \mathcal{A}}, L'), W'_d)$ where

- $W' = \{(w, e) \in W \times E \mid (\mathcal{M}, w) \models pre(e)\}$
- $\sim'_i = \{((w, e), (w', e')) \in W' \times W' \mid w \sim_i w' \text{ and } e \sim_i e'\}$
- $L'((w, e)) = (L(w) - \{p \mid \neg p \text{ is a negative literal of } post(e)) \cup \{p \mid p \text{ is a positive literal of } post(e)\}$
- $W'_d = \{(w, e) \in W' \mid w \in W_d \text{ and } e \in E_d\}$.

$(\mathcal{E}, E_d)$ is **applicable** in $(\mathcal{M}, W_d)$ if for all $w \in W_d$ there is $e \in E_d$ such that $(\mathcal{M}, w) \models pre(e)$.

if s' is the epistemic state induced by a belief state s, and a' is the action model induced by a conditional action a, then $s' \otimes a'$ is the epistemic state induced by $\gamma(s, a)$.

Product update example: Birthday present

$$\underbrace{\overset{\mathsf{At(F,PO1)}}{\underset{\mathsf{At(P,PO1)}}{\bullet}} \overset{}{\text{—Father—}} \overset{\mathsf{At(F,PO1)}}{\underset{\mathsf{At(P,PO2)}}{\bullet}}}_{s_1^{\mathsf{Father}}} \otimes \underbrace{\overset{\langle \mathsf{At(F,PO1)} \wedge \mathsf{At(P,PO1)} \wedge \neg \mathsf{Has(F,P)},\ \mathsf{Has(F,P)} \wedge \neg \mathsf{At(P,PO1)} \rangle}{\bullet} \quad \overset{\langle \mathsf{At(F,PO1)} \wedge \neg \mathsf{At(P,PO1)},\ \top \rangle}{\bullet}}_{\mathsf{TryPickUp(F,P,PO1)}} = \overset{\mathsf{At(F,PO1)}}{\underset{\mathsf{Has(F,P)}}{\bullet}} \quad \overset{\mathsf{At(F,PO1)}}{\underset{\mathsf{At(P,PO2)}}{\bullet}}$$

From belief states and conditional actions to epistemic states and actions

Note how we got from belief states to epistemic states:

belief state	**epistemic state**
set of propositional valuations	multi-set of such valuations with an indistinguishability relation for each agent

We applied the **same trick** to actions:

conditional action	**propositional epistemic action**
sets of pairs $\langle \mathit{pre}, \mathit{post} \rangle$ where *pre* is propositional and *post* is conjunction of literals	multi-set of pairs $\langle \mathit{pre}, \mathit{post} \rangle$ where *pre* is epistemic and *post* as before—and with an indistinguishability relation for each agent

Epistemic planning tasks

Definition

An **epistemic planning task** (or simply **planning task**) is $\Pi = (A, s_0, \phi_g)$, where

- A (actions) is a finite set of epistemic actions.
- s_0 (initial state) is an epistemic state.
- ϕ_g (goal formula) is a formula of epistemic logic.

Global planning task: A planning task (A, s_0, ϕ_g) where s_0 is global.

Planning task for agent i (or i-**local planning task**): A planning task (A, s_0, ϕ_g) where s_0 and all $a \in A$ are local for i.

Associated local planning task of agent i of a planning task $\Pi = (A, s_0, \phi_g)$: $a^i \overset{def}{=} (\{a^i \mid a \in A\}, s_0^i, \phi_g)$.

Induced classical planning tasks

Epistemic planning tasks (A, s_0, ϕ_g) **induce classical planning tasks** $((S, A, \gamma), s_0, S_g)$ in a similar way to propositional planning tasks:

- $\gamma(s, a) = \begin{cases} s \otimes a & \text{if } a \text{ is applicable in } s \\ \text{undefined} & \text{otherwise} \end{cases}$
- $S = \{s_0 \otimes a_1 \otimes \cdots \otimes a_n \mid n \geq 0, a_i \in A\}$
- $S_g = \{s \in S \mid s \models \phi_g\}$

A **solution** to an **epistemic planning task** (A, s_0, ϕ_g) is a solution to the induced classical planning task, that is, a sequence of actions $a_1; \cdots; a_n$ from A such that

- Each a_i is applicable in $s_0 \otimes a_1 \otimes \cdots \otimes a_{i-1}$.
- $s_0 \otimes a_1 \otimes \cdots \otimes a_n \models \phi_g$.

[Bolander and Andersen, 2011]

Planning task example: Birthday present

$s_1 = s_0^{\text{Father}} \otimes$ Go(F, H, PO1) $=$

Father

At(F, PO1), At(P, PO1)	At(F, PO1), At(P, PO2)

$s_2 = s_1 \otimes$ TryPickUp(F, P, PO1) $=$

At(F, PO1), Has(F, P)	At(F, PO1), At(P, PO2)

$s_3 = s_2 \otimes$ Go(F, PO1, PO2) $=$

At(F, PO2), Has(F, P)	At(F, PO2), At(P, PO2)

$s_4 = s_3 \otimes$ TryPickUp(F, P, PO2) $=$

At(F, PO2), Has(F, P)	At(F, PO2), Has(F, P)

$s_6 = s_4 \otimes$ Go(F, PO2, H) $\otimes$ Wrap(F, P) $=$

At(F, H), Has(F, P) Wrapped(P)	At(F, H), Has(F, P) Wrapped(P)

Adding questions

More appropriate solution to the birthday present task: Instead of TryPickUp, then Ask (possibly) followed by PickUp.

We can make a general action for **agent i at location loc asking agent j about whether ϕ**:

$$\mathrm{Ask}(i, j, \phi, loc) = \begin{array}{c} \textit{yes} : \langle \mathrm{At}(i, loc) \wedge \mathrm{At}(j, loc) \wedge K_j \phi, \top \rangle \\[2em] \textit{no} : \langle \mathrm{At}(i, loc) \wedge \mathrm{At}(j, loc) \wedge K_j \neg\phi, \top \rangle \\[2em] ? : \langle \mathrm{At}(i, loc) \wedge \mathrm{At}(j, loc) \wedge \neg K_j \phi \wedge \neg K_j \neg \phi, \top \rangle \end{array}$$

We can then e.g. add an agent Employee to our birthday present planning task and add the proposition At(Employee, PostOffice1) to the initial state of the task.

Example with questions

Consider the new initial state of the birthday present task (from the internal perspective of the father):

$$s_0 = \begin{array}{ccc} \odot & \overset{\text{Father}}{\rule{6em}{0.4pt}} & \odot \\ \mathsf{At}(\mathsf{F},\mathsf{H}) & & \mathsf{At}(\mathsf{F},\mathsf{H}) \\ \mathsf{At}(\mathsf{P},\mathsf{PO1}) & & \mathsf{At}(\mathsf{P},\mathsf{PO2}) \\ \mathsf{At}(\mathsf{E},\mathsf{PO1}) & & \mathsf{At}(\mathsf{E},\mathsf{PO1}) \end{array}$$

None of the actions $\mathsf{Ask}(\mathsf{F},\mathsf{E},\cdot,\cdot)$ are applicable (F and E not in same location). But after going to PO1 the father can ask:

$$\begin{array}{ccc} \odot & \overset{\text{Father}}{\rule{6em}{0.4pt}} & \odot \\ \mathsf{At}(\mathsf{F},\mathsf{H}) & & \mathsf{At}(\mathsf{F},\mathsf{H}) \\ \mathsf{At}(\mathsf{P},\mathsf{PO1}) & & \mathsf{At}(\mathsf{P},\mathsf{PO2}) \\ \mathsf{At}(\mathsf{E},\mathsf{PO1}) & & \mathsf{At}(\mathsf{E},\mathsf{PO1}) \end{array} \otimes \mathsf{Go}(\mathsf{F},\mathsf{H},\mathsf{PO1}) \otimes \mathsf{Ask}(\mathsf{F},\mathsf{E1},\mathsf{At}(\mathsf{P},\mathsf{PO1}),\mathsf{PO1}) =$$

$$\begin{array}{ccc} \odot & \overset{\text{Father}}{\rule{6em}{0.4pt}} & \odot \\ \color{blue}{\mathsf{At}(\mathsf{F},\mathsf{PO1})} & & \color{blue}{\mathsf{At}(\mathsf{F},\mathsf{PO1})} \\ \mathsf{At}(\mathsf{P},\mathsf{PO1}) & & \mathsf{At}(\mathsf{P},\mathsf{PO2}) \\ \mathsf{At}(\mathsf{E},\mathsf{PO1}) & & \mathsf{At}(\mathsf{E},\mathsf{PO1}) \end{array} \otimes \mathsf{Ask}(\mathsf{F},\mathsf{E1},\mathsf{At}(\mathsf{P},\mathsf{PO1}),\mathsf{PO1}) = \begin{array}{ccc} \odot & \qquad & \odot \\ \mathsf{At}(\mathsf{F},\mathsf{PO1}) & & \mathsf{At}(\mathsf{F},\mathsf{PO1}) \\ \mathsf{At}(\mathsf{P},\mathsf{PO1}) & & \mathsf{At}(\mathsf{P},\mathsf{PO2}) \\ \mathsf{At}(\mathsf{E},\mathsf{PO1}) & & \mathsf{At}(\mathsf{E},\mathsf{PO1}) \end{array}$$

What is the father to do next? We need conditional plans...

Approaches to conditional epistemic planning

Epistemic plans as (knowledge-based) programs [Andersen et al., 2012]:

$\mathsf{Go}(\mathsf{Father}, \mathsf{H}, \mathsf{PO1}); \mathsf{TryPickUp}(\mathsf{Father}, \mathsf{Present}, \mathsf{PO1});$
$\textbf{if } K_{\mathsf{Father}}\mathsf{Has}(\mathsf{Father}, \mathsf{Present}) \textbf{ then } \mathsf{Go}(\mathsf{Father}, \mathsf{PO1}, \mathsf{H}); \mathsf{Wrap}(\mathsf{Father}, \mathsf{Present})$
$\textbf{else } \mathsf{Go}(\mathsf{Father}, \mathsf{PO1}, \mathsf{PO2}); \dots$

Epistemic plans as PDL programs [van Eijck, 2014]: The program **if** ϕ **then** π_1 **else** π_2 is shorthand for the PDL program $(\phi?; \pi_1) \cup (\neg\phi?; \pi_2)$

Epistemic plans as policies/strategies/protocols: Mappings from epistemic states to epistemic actions.

Here we consider only policies.

i-local epistemic policies

We use S^{gl} to denote the set of global epistemic states. For any state $s = (\mathcal{M}, W_d)$, we let $Globals(s) = \{(\mathcal{M}, w) \mid w \in W_d\}$.

Definition. Let $\Pi = (A, s_0, \phi_g)$ be a planning task and $i \in \mathcal{A}$ be an agent. An i-**local policy** π for Π is a partial mapping $\pi : S^{\mathrm{gl}} \hookrightarrow A$:

(knowledge of preconditions) If $\pi(s) = a$ then a is applicable in s^i.

(uniformity) If $s^i = t^i$ then $\pi(s) = \pi(t)$.

Definition. An **execution** of a policy π from a global state s_0 is a maximal (finite or infinite) sequence of alternating global states and actions $(s_0, a_1, s_1, a_2, s_2, \dots)$ such that for all $m \geq 0$,

(1) $\pi(s_m) = a_{m+1}$, and

(2) $s_{m+1} \in Globals(s_m \otimes a_{m+1})$.

An execution is called **successful** for a global planning task $\Pi = (A, s_0, \phi_g)$ if it is a finite execution $(s_0, a_1, s_1, \dots, a_n, s_n)$ such that $s_n \models \phi_g$.

i-local epistemic policies

Definition. A policy π for a planning task $\Pi = (A, s_0, \phi_g)$ is called a **solution** to Π if $Globals(s_0) \subseteq \text{dom}(\pi)$ and for each $s \in \text{dom}(\pi)$, any execution of π from s is successful for Π.

Example (birthday present). The father can now branch on the outcome of asking about the location of the parcel:

$$\pi\left(\begin{array}{c} \bullet \\ \text{At}(\text{F}, \text{PO1}) \\ \text{At}(\text{P}, \text{PO1}) \\ \text{At}(\text{E}, \text{PO1}) \end{array} \qquad \begin{array}{c} \bullet \\ \text{At}(\text{F}, \text{PO1}) \\ \text{At}(\text{P}, \text{PO2}) \\ \text{At}(\text{E}, \text{PO1}) \end{array} \right) = \text{PickUp}(\text{F}, \text{P}, \text{PO1})$$

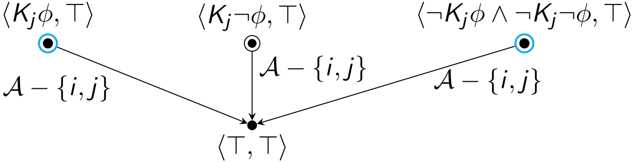
$$\pi\left(\begin{array}{c} \bullet \\ \text{At}(\text{F}, \text{PO1}) \\ \text{At}(\text{P}, \text{PO1}) \\ \text{At}(\text{E}, \text{PO1}) \end{array} \qquad \begin{array}{c} \bullet \\ \text{At}(\text{F}, \text{PO1}) \\ \text{At}(\text{P}, \text{PO2}) \\ \text{At}(\text{E}, \text{PO1}) \end{array} \right) = \text{Go}(\text{Father}, \text{PO1}, \text{PO2}).$$

We could also add an action $\text{CallAsk}(i, j, \phi)$ where agent i calls agent j to ask whether ϕ: As Ask but without At-atoms in the precondition. Then the father's first action could be a phone call, and he would branch on its outcome.

More complicated actions: (semi-)private actions

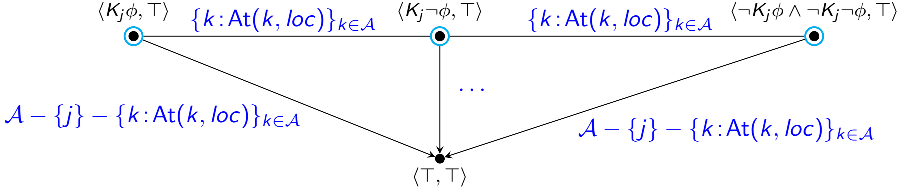
A phone call is normally only observable to the agents involved in the call. Improved version of CallAsk ($\mathcal{A}$ is set of agents):

$\text{CallAsk}(i, j, \phi) =$



Note that the accessibility relation is no longer an equivalence relation!

We could also include that all agents at the location of the caller get to hear the question, but not the answer. Then $\text{CallAsk}(i, j, \phi, loc) =$



Edge-conditioned event models: [Bolander, 2014]. Relates to [Kooi and Renne, 2011].

Achieving the epistemic goal in the birthday example

Suppose wrapping a present is observed exactly by those in the same location (those who are *copresent* with the acting agent):

$$\mathsf{Wrap}(agt, obj, loc) =$$

$$\begin{tikzcd}[column sep=huge] \underset{\langle \mathsf{At}(agt, loc) \wedge \mathsf{Has}(agt, obj) \wedge \neg \mathsf{Wrapped}(obj), \mathsf{Wrapped}(obj) \rangle}{\odot} \arrow[r, "\{i : \neg \mathsf{At}(i, loc)\}_{i \in \mathcal{A}}"] & \underset{\langle \top, \top \rangle}{\bullet} \end{tikzcd}$$

For the planning problem with initial state

$$s_0 = \odot\ \mathsf{At}(\mathsf{Father}, \mathsf{Home}), \mathsf{At}(\mathsf{Present}, \mathsf{PostOffice1}), \mathsf{At}(\mathsf{Daughter}, \mathsf{Home})$$

and goal formula

$$\begin{aligned} \phi_g = \quad & \mathsf{At}(\mathsf{Father}, \mathsf{Home}) \wedge \mathsf{Has}(\mathsf{Father}, \mathsf{Present}) \wedge \\ & \mathsf{Wrapped}(\mathsf{Present}) \wedge \neg K_{\mathsf{Daughter}} \mathsf{Has}(\mathsf{Father}, \mathsf{Present}). \end{aligned}$$

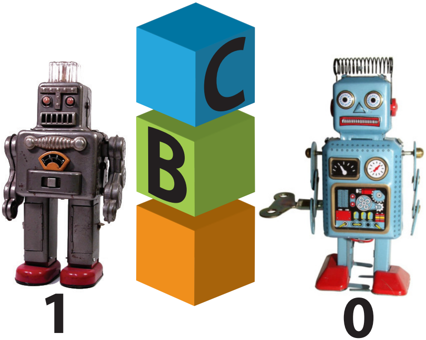
it is then easy to show that:

Not solution: $\mathsf{Go}(\mathsf{F}, \mathsf{H}, \mathsf{PO1}); \mathsf{PickUp}(\mathsf{F}, \mathsf{P}, \mathsf{PO1}); \mathsf{Go}(\mathsf{F}, \mathsf{PO1}, \mathsf{H}); \mathsf{Wrap}(\mathsf{F}, \mathsf{P}, \mathsf{H})$

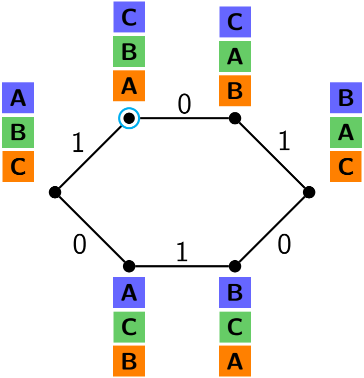
Solution: $\mathsf{Go}(\mathsf{F}, \mathsf{H}, \mathsf{PO1}); \mathsf{PickUp}(\mathsf{F}, \mathsf{P}, \mathsf{PO1}); \mathsf{Wrap}(\mathsf{F}, \mathsf{P}, \mathsf{H}); \mathsf{Go}(\mathsf{F}, \mathsf{PO1}, \mathsf{H})$

Different types of epistemic planning: Centralised

Centralised planning: One omniscient agent planning for everyone.



$s_0 =$



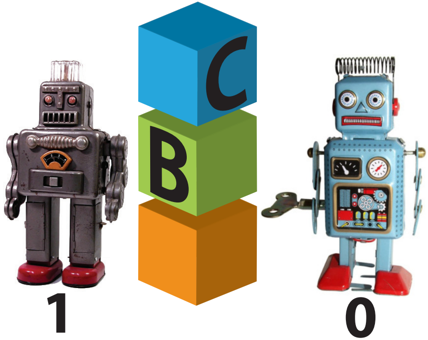
Centralised planning means *global planning task*.

Example. $\Pi = (A, s_0, \phi_g)$ with global s_0 as above and goal formula $\phi_g = \mathrm{On}(A, B) \wedge \mathrm{On}(B, C) \wedge \mathrm{On}(C, \mathrm{Table})$.

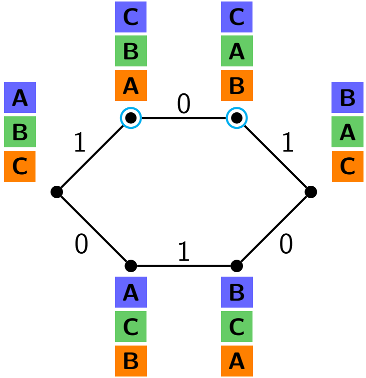
Solution. $\pi = \mathrm{Put}(\mathrm{Blue}, \mathrm{Table}); \mathrm{Put}(\mathrm{Green}, \mathrm{Blue}); \mathrm{Put}(\mathrm{Orange}, \mathrm{Green})$

Different types of epistemic planning: *i*-local

i-**local planning**: One agent, *i*, planning from its local perspective in the system. What we've considered so far.



$s_0^0 =$

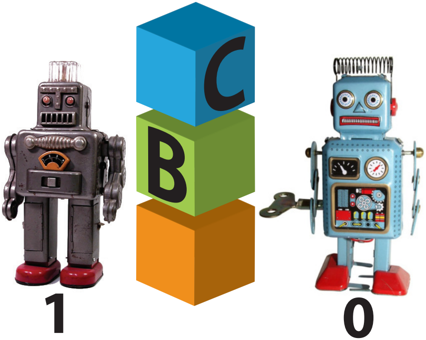


i-local planning means *local planning tasks for agent i*.

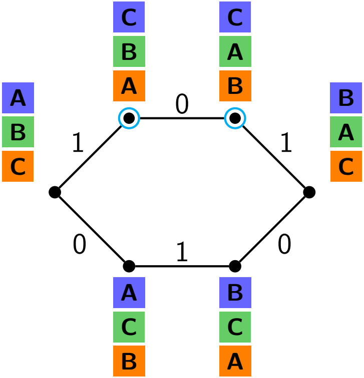
Example. The 0-local planning task $\Pi^0 = (A, s_0^0, \phi_g)$, where A, s_0 and ϕ_g are as on the previous slide.

π is no longer a solution. In $\gamma(s_0^0, \pi)$, both $\begin{array}{|c|}\hline A\\ \hline B\\ \hline C\\ \hline\end{array}$ and $\begin{array}{|c|}\hline B\\ \hline A\\ \hline C\\ \hline\end{array}$ will be designated.

i-local planning cont'd



$s_0^0 =$

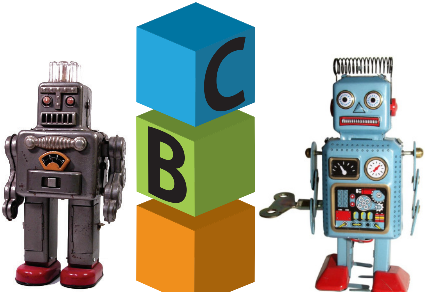


As in the birthday present example we can introduce an action Ask(i, j, ϕ) for agent i asking agent j whether ϕ (and getting a sincere reply).

Then a solution to the 0-local planning task (A, s_0^0, ϕ_g) is:

$\pi_0 =$ Put(Blue, Table); Ask(0, 1, Label(Green, A)); **if** K_0Label(Green, A)
then Put(Green, Table); Put(Orange, Blue); Put(Green, Orange)
else Put(Green, Blue); Put(Orange, Green)

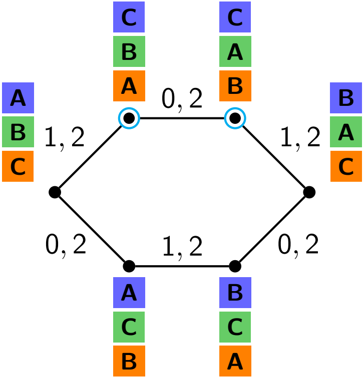
i-local planning cont'd



1 **0**

2

$s_0^0 =$



π_0 is no longer a solution if Ask$(0, 1, \text{Label}(\text{Green}, A))$ is replaced by Ask$(0, 2, \text{Label}(\text{Green}, A))$.

Agent 0 has a Theory of Mind (ToM) [Premack and Woodruff, 1978] of agents 1 and 2 allowing him to infer who to ask. Epistemic planning is planning with ToM capabilities.

i-local planning cont'd: one agent planning for many

i-local planning can also be one agent planning for many.

We can define an **owner function** $\omega : A \to \mathcal{A}$ mapping actions to agents. $\omega(a) = i$ means that the action a is owned by agent i: only agent i can execute it.

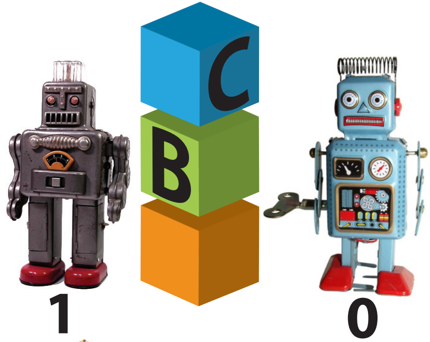
We could e.g. have

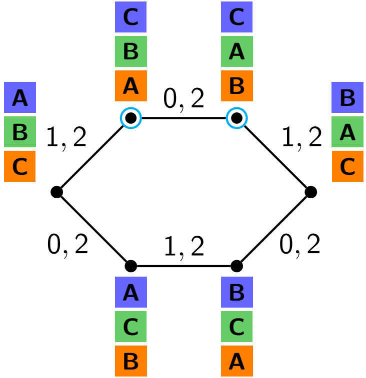
$$\begin{aligned}
&\omega(\mathsf{Put}(\mathsf{Blue}, \cdot)) = 0\\
&\omega(\mathsf{Put}(\mathsf{Green}, \cdot)) = 1\\
&\omega(\mathsf{Put}(\mathsf{Orange}, \cdot)) = 2.
\end{aligned}$$

A 0-local plan would then be computed by agent 0, from agent 0's perspective, and agent 0 would distribute the actions of the plan to the respective owners (agent 0 is the *leader*).

Diff. types of epist. planning: implicit coordination

Planning with implicit coordination: All agents plan for all agents, and plan for how and when to interact. They have a joint goal.



$s_0^0 =$ 

Implicitly coordinated plans and policies will be treated in the contributed talk...

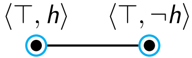
Simple example. The mother tells the father at which post office the parcel is and leaves it to him to pick it up.

Observability

Like in the automated planning literature (e.g. [Rintanen, 2006]), we can distinguish between scenarios that are **fully observable**, **unobservable** or **partially observable**. In the *single-agent case* of epistemic planning:

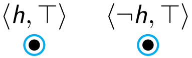
- **Unobservable** states/actions/planning tasks: All pairs of nodes are connected by an indistinguishability edge. For instance a coin toss under a dice cup (h for landing heads):

  `hidden coin toss` :



- **Fully observable** states/actions/planning tasks: No indistinguishability edges between distinct nodes. For instance lifting the cup to observe the outcome of the coin toss:

  `lift cup` :



- **Partially observable**: Anything else.

Plan existence problem in epistemic planning

The **plan existence problem in epistemic planning** is the following decision problem: "Given an epistemic planning task (A, s_0, ϕ_g), does it have a solution?"

Theorem ([Jensen, 2013])

Complexities of the plan existence problem in **single-agent** *epistemic planning:*

- *Arbitrary planning tasks (part. observability): 2EXPTIME-complete.*
- *Fully observable planning tasks: EXPTIME-complete.*
- *Unobservable planning tasks: EXPSPACE-complete.*

These results match, as could be expected, the results from nondeterministic propositional planning (Rintanen, 2006).

Theorem ([Bolander and Andersen, 2011])

The plan existence problem in **multi-agent** *epistemic planning with at least 3 agents is undecidable (on S5 frames).*

Generalised results on (un)decidability of plan existence in epistemic planning

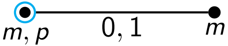
L	transitive	Euclidean	reflexive	
K				
KT			✓	
K4	✓			
K45	✓	✓		← belief
S4	✓		✓	
S5	✓	✓	✓	← knowledge

Theorem
([Aucher and Bolander, 2013])
The figure to the right summarises results on decidability (D) and undecidability (UD) of the plan existence problem in **purely epistemic** *planning (all postconditions are* $\top$*).*

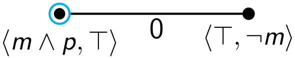
	1 agent	≥ 2 agents
K	**UD**	UD
KT	UD	UD
K4	UD	UD
K45	D	UD
S4	UD	UD
S5	D	UD

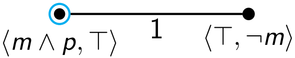
Infinite state spaces in epistemic planning

The following example is based on the coordinated attack problem (Byzantine generals problem). Define

$s_0 =$ 

m means "the messenger is alive". Let A contain the two actions $send_{01}$ and $send_{10}$ given by:

$send_{01} =$ 

$send_{10} =$ 

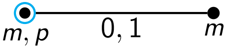
$send_{ij}$ represents agent i sending the message p to agent j via the messenger. Either the message arrives at its destination (the left event) or the messenger gets killed on the way (the right event).

Let $\phi_g = Cp$. Then $\Pi = (A, s_0, \phi_g)$ is a global planning task (centralised planning).

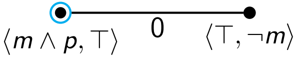
Infinite state spaces cont'd

Recall:

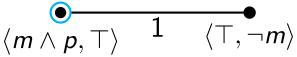
$s_0 =$



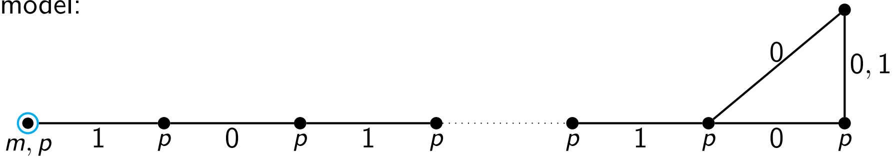
$send_{01} =$



$send_{10} =$



It is easy to show that
$s_0 \otimes send_{01} \otimes send_{10} \otimes \cdots \otimes send_{01} \otimes send_{10} \otimes send_{01}$ is the following model:



Each new application of $send_{01}$ or $send_{10}$ extends the depth of the model by 1, and it is not bisimilar to any smaller model. Π has no solution.

Fragments of epistemic planning

The undecidability result shows that **allowing arbitrary levels of higher-order reasoning leads to undecidability of planning**. We should look for decidable fragments.

Theorem ([Yu et al., 2013])

The plan existence problem for (multi-agent) epistemic planning with only propositional preconditions and no common knowledge is decidable (in NON-ELEMENTARY). More precisely, in $(n+1)$*-EXPTIME for planning tasks in which the goal formula has modal depth* n*.*

Other decidable fragments are found in [Löwe et al., 2011, Bolander and Andersen, 2011, Yu et al., 2013, Maubert, 2014, Bolander et al., 2015].

E.g. planning where the frames of the action models are chains is NP-complete. This covers, among others, private and public announcements, and hence e.g. the actions in the games Cluedo (Clue) and Hanabi. [Bolander et al., 2015].

Alternative approaches to epistemic planning

We can distinguish between:

- **Semantic approaches** (states are semantic objects) and **Syntactic approaches** (states are knowledge-bases).
- **Action model based approaches** and **state-transition system based approaches**.

Epistemic planning based on DEL is *semantic* and *action model based*.

- **Syntactic approaches to epistemic planning**:
  - The (single-agent) PKS planner [Petrick and Bacchus, 2004].
  - The (multi-agent) planning framework of [Muise et al., 2015].
  - The compilation approach of [Kominis and Geffner, 2014].
- **State-transition system based approaches** in logics of strategic ability:
  - ATEL [van der Hoek and Wooldridge, 2002]. Cannot express *de re* knowledge of a strategy.
  - Constructive Strategic Logic (CSL) [Jamroga and Aagotnes, 2007]. Cannot express implicit coordination.

Implemented epistemic planners

- The PKS planner [Petrick and Bacchus, 2004].
- The multi-agent planner of [Muise et al., 2015].
- Epistemic planning based on DEL:
  `https://gkigit.informatik.uni-freiburg.de/tengesser/planner`

Extensions of epistemic planning

There are 4 main ingredients in planning: **states**, **actions**, **plans** and **goals**. We have shown how to generalise the **states** and **actions** of classical propositional planning, but not talked about generalised **plans** and **goals**:

- **Generalised plans in DEL-planning**: Weak and strong **conditional plans** [Andersen et al., 2012]; **Plausibility plans** [Andersen et al., 2015]. **Implicitly coordinated plans/policies** [Engesser et al., 2017].
- **Generalised goals in DEL-planning (extended goals)**: 1) Via constructive model checking of CTL formulas on induced classical planning tasks; 2) using DEL*; 3) Using temporal epistemic logics (e.g. ATEL [van der Hoek and Wooldridge, 2002] and CSL [Jamroga and Aagotnes, 2007]).

Summary

- I have presented a framework for epistemic planning based on DEL.
- It very naturally generalises classical propositional planning and planning under partial observability with conditional actions.
- Epistemic planning is undecidable when no bound can be put on the depth of reasoning required to reach a goal.
- Lots of interesting future work is left, e.g. finding fragments of reasonable complexity and devising suitable domain description languages.

Application example: robotic bartender

A robotic bartender planning using the PKS planner. Reported in [Petrick and Foster, 2013] (best paper award at ICAPS 2013).

Two-counter machines

The undecidability proof is by a reduction of the halting problem for **two-counter machines**:

Configurations: $\boxed{\mathsf{k}}\boxed{\mathsf{l}}\boxed{\mathsf{m}}$, where $k, l, m \in \mathbb{N}$.

R_0 R_1

register 0 register 1

Instruction set: $\mathrm{inc}(0), \mathrm{inc}(1), \mathrm{jump}(j), \mathrm{jzdec}(0, j), \mathrm{jzdec}(1, j), \mathrm{halt}$.

Computation step example:

$$\boxed{\mathsf{k}}\boxed{\mathsf{l}}\boxed{\mathsf{m}} \xrightarrow{\mathsf{inc(0)}} \boxed{\mathsf{k+1}}\boxed{\mathsf{l+1}}\boxed{\mathsf{m}}$$

R_0 R_1 R_0

The halting problem for two-counter machines is undecidable [Minsky, 1967].

Proof idea for undecidability of epistemic planning

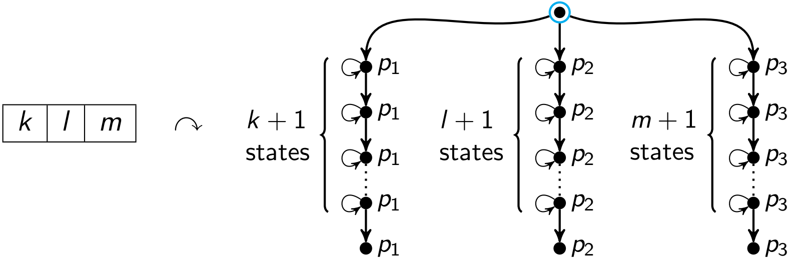
Our proof idea is this. For each two-register machine, construct a corresponding planning task where:

- The **initial state** encodes the initial configuration of the machine.
- The **epistemic actions** encode the instructions of the machine.
- The **goal formula** is true of all epistemic states representing halting configurations of the machine.

Then show that the two-register machine halts iff the corresponding planning task has a solution. (Execution paths of the planning task encodes computations of the machine).

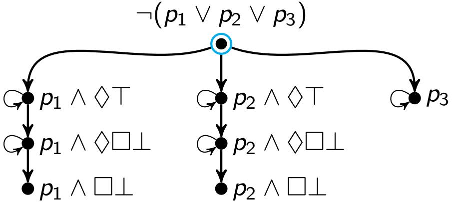
Encodings

Encoding configurations as epistemic states:



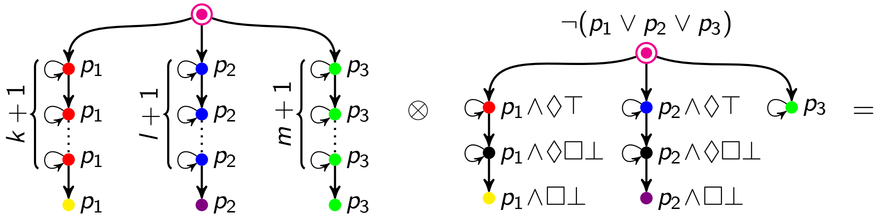
Encoding instructions as epistemic actions (note: only preconditions!):

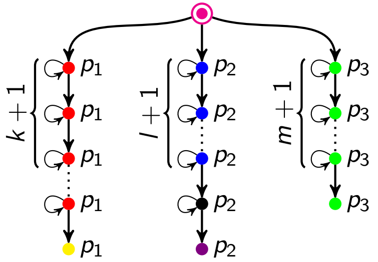
inc(0) ↷



The computation step $\boxed{\begin{array}{c|c|c} k & l & m \end{array}} \xrightarrow{\text{inc}(0)} \boxed{\begin{array}{c|c|c} k+1 & l+1 & m \end{array}}$ is mimicked by:

$$\textit{encoding}\left(\boxed{\begin{array}{c|c|c} k & l & m \end{array}}\right) \otimes \textit{encoding}(\text{inc}(0)) =$$





$$= \textit{encoding}\left(\boxed{\begin{array}{c|c|c} k+1 & l+1 & m \end{array}}\right)$$

Epistemic planning task example

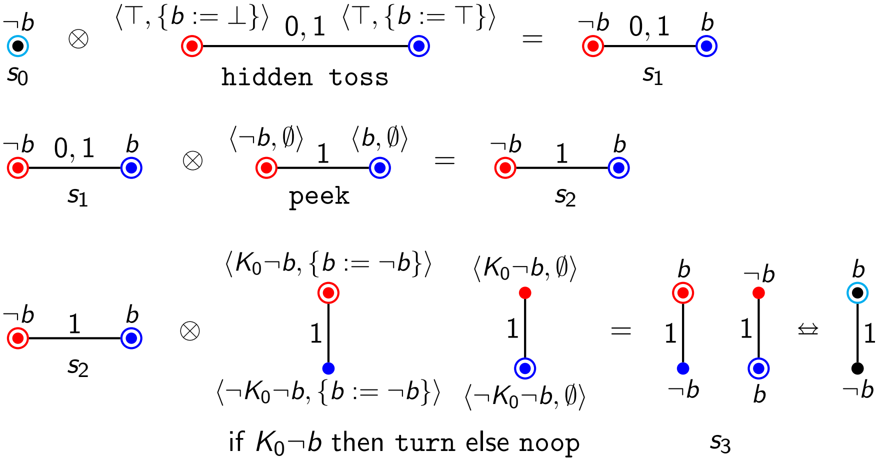
Initial state s_0: $\neg b$

Goal ϕ_g: $\underbrace{K_0 b}_{\text{I know } b} \wedge \underbrace{\neg K_1 b}_{\text{You don't know } b}$

Epistemic actions:

- `noop`: $\langle \top, \emptyset \rangle$
- `turn coin`: $\langle \top, \{b := \neg b\} \rangle$
- `lift cup`: $\langle \neg b, \emptyset \rangle$ $\quad$ $\langle b, \emptyset \rangle$ (a **public sensing action**)
- `hidden toss`: $\langle \top, \{b := \bot\} \rangle \overset{0,1}{\text{———}} \langle \top, \{b := \top\} \rangle$
- `peek`: $\langle \neg b, \emptyset \rangle \overset{1}{\text{———}} \langle b, \emptyset \rangle$

Epistemic planning task example (cont'd)



Goal achieved in s_3! Plan:

hidden toss; peek; if $K_0 \neg b$ then turn else noop.

References I

Andersen, M. B., Bolander, T. and Jensen, M. H. (2012).
Conditional Epistemic Planning.
Lecture Notes in Artificial Intelligence *7519*, 94–106.

Andersen, M. B., Bolander, T. and Jensen, M. H. (2015).
Don't plan for the unexpected: Planning based on plausibility models.
Logique et Analyse *58(230)*.

Aucher, G. and Bolander, T. (2013).
Undecidability in Epistemic Planning.
In Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI) pp. 27–33,.

Bolander, T. (2014).
Seeing is Believing: Formalising False-Belief Tasks in Dynamic Epistemic Logic.
In Proceedings of the European Conference on Social Intelligence (ECSI-2014), (Herzig, A. and Lorini, E., eds), vol. 1283, of CEUR Workshop Proceedings pp. 87–107, CEUR-WS.org.

Bolander, T. and Andersen, M. B. (2011).
Epistemic Planning for Single- and Multi-Agent Systems.
Journal of Applied Non-Classical Logics *21*, 9–34.

Bolander, T., Jensen, M. H. and Schwarzentruber, F. (2015).
Complexity Results in Epistemic Planning.
In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015, (Yang, Q. and Wooldridge, M., eds), pp. 2791–2797, AAAI Press.

References II

Bylander, T. (1994).
The computational complexity of propositional STRIPS planning.
Artificial Intelligence *69*, 165–204.

Engesser, T., Bolander, T., Mattmüller, R. and Nebel, B. (2017).
Cooperative Epistemic Multi-Agent Planning for Implicit Coordination.
In Proceedings of Methods for Modalities Electronic Proceedings in Theoretical Computer Science.

Fikes, R. and Nilsson, N. (1971).
STRIPS: A new approach to the application of theorem proving to problem solving.
Artificial Intelligence *2*, 189–203.

Ghallab, M., Nau, D. S. and Traverso, P. (2004).
Automated Planning: Theory and Practice.
Morgan Kaufmann.

Jamroga, W. and Aagotnes, T. (2007).
Constructive knowledge: what agents can achieve under imperfect information.
Journal of Applied Non-Classical Logics *17*, 423–475.

Jensen, M. H. (2013).
Planning using dynamic epistemic logic: Correspondence and complexity.
In Logic, Rationality, and Interaction pp. 316–320. Springer.

Kominis, F. and Geffner, H. (2014).
Beliefs in multiagent planning: From one agent to many.
In Proc. ICAPS Workshop on Distributed and Multi-Agent Planning.

References III

Kooi, B. and Renne, B. (2011).
Generalized arrow update logic.
In Proceedings of the 13th Conference on Theoretical Aspects of Rationality and Knowledge pp. 205–211, ACM.

Löwe, B., Pacuit, E. and Witzel, A. (2011).
DEL planning and some tractable cases.
In LORI 2011, (van Ditmarsch, H., Lang, J. and Ju, S., eds), vol. 6953, of Lecture Notes in Artificial Intelligence pp. 179–192, Springer.

Maubert, B. (2014).
Fondations logiques des jeux à information imparfaite: stratégies uniformes.
PhD thesis, Université de Rennes 1.

Minsky, M. (1967).
Computation.
Prentice-Hall.

Muise, C., Belle, V., Felli, P., McIlraith, S., Miller, T., Pearce, A. R. and Sonenberg, L. (2015).
Planning Over Multi-Agent Epistemic States: A Classical Planning Approach (Amended Version).
In Distributed and Multi-Agent Planning (DMAP-15) pp. 60–67,.

Petrick, R. P. A. and Bacchus, F. (2004).
PKS: Knowledge-Based Planning with Incomplete Information and Sensing.
In ICAPS 2004.

Petrick, R. P. A. and Foster, M. E. (2013).
Planning for social interaction in a robot bartender domain.
In Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS 2013) pp. 389–397,.

References IV

Premack, D. and Woodruff, G. (1978).
Does the chimpanzee have a theory of mind?
Behavioral and Brain Sciences *1*, 515–526.

Rintanen, J. (2006).
Introduction to automated planning.

Russell, S. and Norvig, P. (1995).
Artificial Intelligence: A Modern Approach.
Prentice Hall.

van der Hoek, W. and Wooldridge, M. (2002).
Tractable Multiagent Planning for Epistemic Goals.
In In Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002) pp. 1167–1174, ACM Press.

van Eijck, J. (2014).
Dynamic epistemic logics.
In Johan van Benthem on Logic and Information Dynamics pp. 175–202. Springer.

Yu, Q., Wen, X. and Liu, Y. (2013).
Multi-agent epistemic explanatory diagnosis via reasoning about actions.
In Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI) pp. 27–33,.