AUTOMATED TASK PLANNING

towards Multi-Agent, Flexible, Temporal, Epistemic and Contingent models

Tiago de Lima [4,5] Frédéric Maris[1,5] Ajdin Sumic[2] Thierry Vidal[2] Bruno Zanutini[3,5]

[1]IRIT, Université Toulouse 3 – Paul Sabatier

[2]LGP, École nationale d'ingénieurs de Tarbes

[3]GREYC, Université de Caen Normandie

[4]CRIL, Université d'Artois

[5]CNRS

European Conference on Artificial Intelligence
Santiago de Compostela, Spain
20 October 2024

Part 1

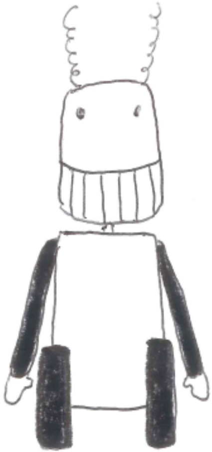
Introduction

General setting

Planning:

- one or several agents
- in some environment
- with goals/missions
- with actuators and sensors

Goal: compute plan of actions



Offline and online phases

Planning problem (offline):

- input: initial state(s), actions, goal
- output: $\pi =$ plan/policy of actions to take from initial state(s) to goal

Execution of π (online):

1. execute first action prescribed by π
2. observe information about environment
3. execute action prescribed by π for history of information so far
4. if goal not reached, goto 2

Important note: planning and execution may well be interleaved

Classical planning

- Initial state fully known
- Goal = set of states
- Only actuators, no sensor
- Effects of actuators deterministic
- Effects of actuators fully known

Typically offline planning: ahead of mission start

Adding nondeterminism

Outcome of action cannot be fully predicted even if state fully known

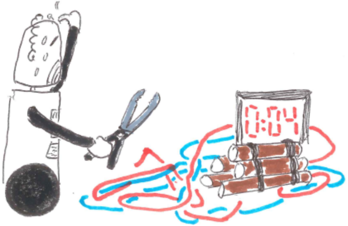
One of the possible outcomes arises each time the action is taken

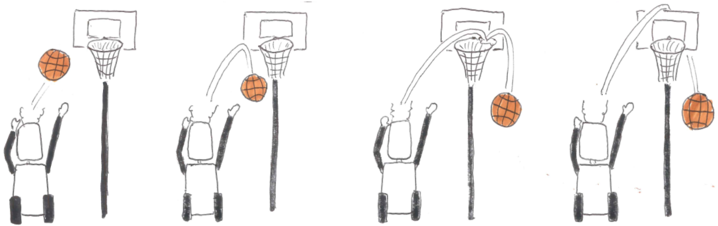
Adding nondeterminism

Outcome of action cannot be fully predicted even if state fully known

One of the possible outcomes arises each time the action is taken

Examples:



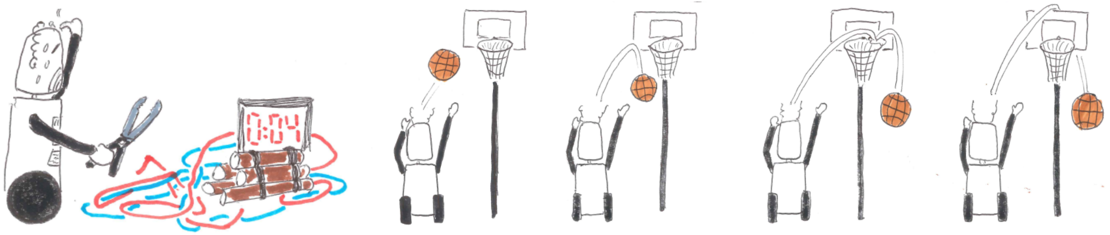


Adding nondeterminism

Outcome of action cannot be fully predicted even if state fully known

One of the possible outcomes arises each time the action is taken

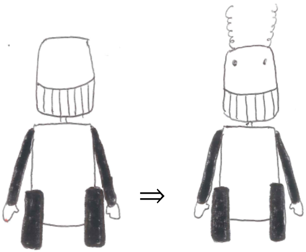
Examples:



Two versions: nondeterministic and probabilistic

→ Conformant planning

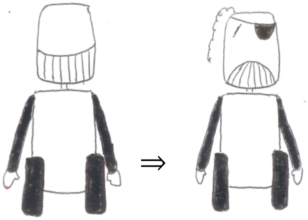
Adding sensing



Using sensor:

- gives information about current state

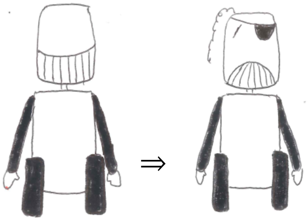
Adding sensing



Using sensor:

- gives information about current state
- but imperfect/noisy in general

Adding sensing

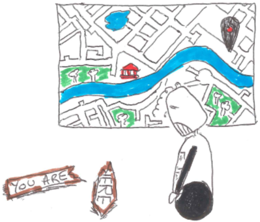


Using sensor:

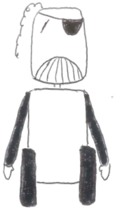
- gives information about current state
- but imperfect/noisy in general

Together with nondeterminism:

- current state cannot be tracked exactly
- plan ⇒ policy of actions
- policy contingent on sensor observations



Adding sensing

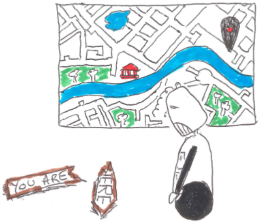
 $\Rightarrow$ 

Using sensor:

- gives information about current state
- but imperfect/noisy in general

Together with nondeterminism:

- current state cannot be tracked exactly
- plan $\Rightarrow$ policy of actions
- policy contingent on sensor observations



$\rightarrow$ Contingent planning

Adding time

Durative actions:

- execution not instantaneous in general
- real problems have deadlines

Adding time

Durative actions:

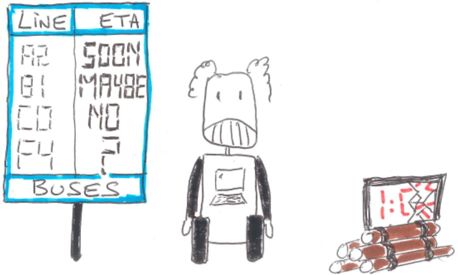
- execution not instantaneous in general
- real problems have deadlines
- parallel execution may be required

→ Temporal planning

Adding contingencies

Exogenous events, other agents. . . :

- constrain the plan
- agent does not control
  - when they occur
  - what they do
- plan must adapt to actual occurrences

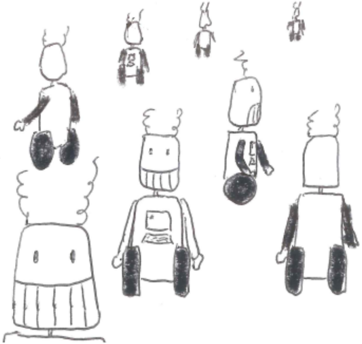


→ Flexible planning

Adding other agents

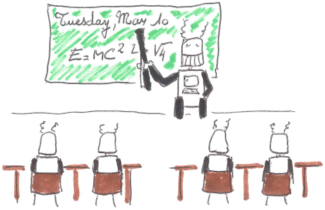
Many combinations:

- plan execution centralized/decentralized
- plan computation centralized/decentralized
- agents collaborate/compete/both
- agents have/do not have explicit communication
- effects are from individual/joint actions
- effects are deterministic/nondet./stochastic
- etc.



$\rightarrow$ Multiagent path finding, decentralized (PO)MDPs, extensive-form games, stochastic games...

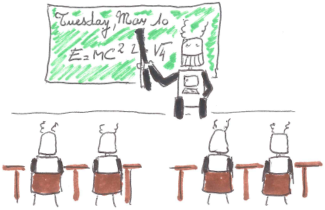
Adding theory of mind



Some problems involve knowledge/beliefs:

- goals to learn sth
- goals to make other agents believe or know sth

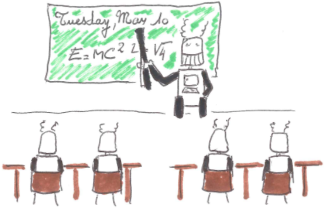
Adding theory of mind



Some problems involve knowledge/beliefs:

- goals to learn sth
- goals to make other agents believe or know sth

Plans may require to

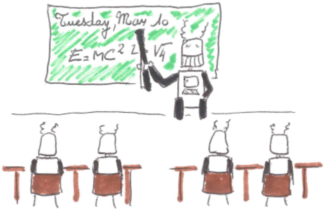


Some problems involve knowledge/beliefs:

- goals to learn sth
- goals to make other agents believe or know sth

Plans may require to sense others' beliefs/knowledge

Adding theory of mind

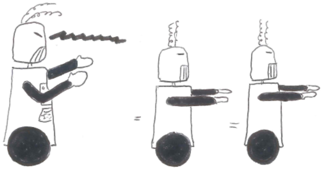


Some problems involve knowledge/beliefs:

- goals to learn sth
- goals to make other agents believe or know sth

Plans may require to

sense others' beliefs/knowledge



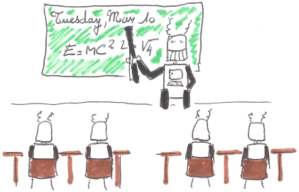
act on others' beliefs/knowledge

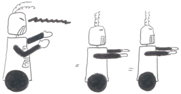
→ Epistemic planning

Focus of this tutorial

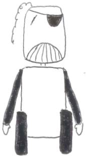
Classical

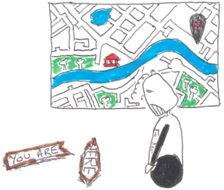
Epistemic



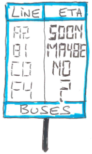


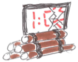
Contingent



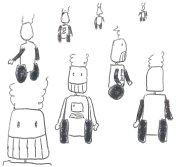


Temporal flexible





Multiagent



Part 2

A little history: classical planning

Table of contents

Section 1

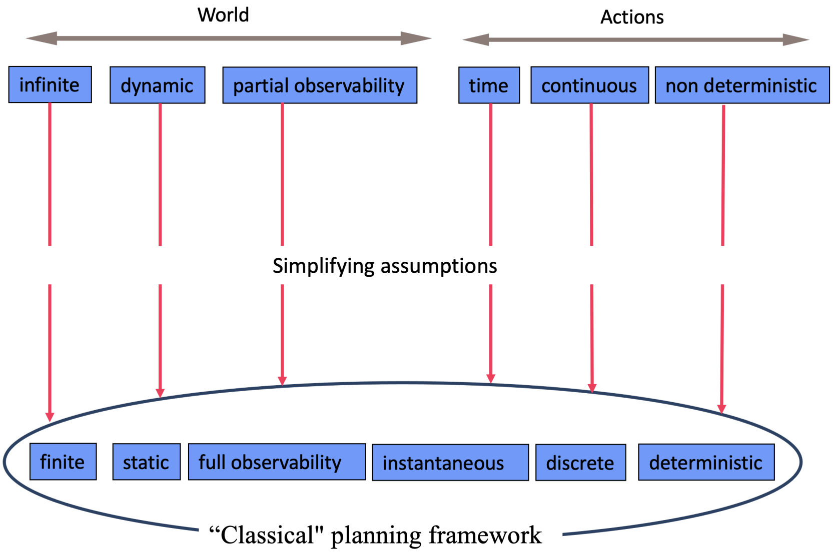
Introduction

The classical framework

The general problem of the synthesis of a solution plan is very complex because planning involves three stages:

- the selection of *applicable actions* (among the many actions available)
- the choice among them of *relevant actions* to move towards the goal (which requires reasoning about their causal dependencies)
- *reasoning on their interactions* to obtain an executable scheduling of these actions

The classical framework



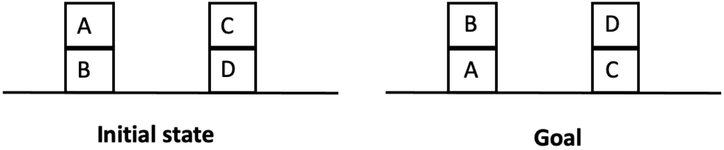
Section 2

Languages for planning

Languages for planning

The STRIPS language: example of the “domain of cubes”

- STRIPS representation of the problem: initial state and goal




```
Initial state : {on(A,B), onTable(B), on(C,D), onTable(D),libre(A), free(C)}
Goal : {on(B, A), onTable(A), on(D, C), onTable(C)}
```

- STRIPS representation of operators (two are required)

```
# pick block ?x which is on block ?y, drop it on block ?z
Move-on-block(?x, ?y, ?z) :
    Prec = {on(?x, ?y), free(?x), free(?z)}
    Add = {on(?x, ?z), free(?y)}
    Del = {on(?x, ?y), free(?z)}

Move-on-table (…) -> UP TO YOU TO COMPLETE IT
```

Languages for planning

ADL language

Subset of first order logic: an operator o is represented by its name and a doublet ⟨preconditions, effects⟩. Additions and deletes are grouped in the effects (additions: positive literals, deletes: negative literals). ADL allows one to use logical connectors and quantifiers.

- in Pre(o) and Eff(o), $\wedge$ represents a conjunction of formulas
- in Eff(o), $\rightarrow$ makes it possible to represent a conditional effect
- in Pre(o) and in the antecedent of conditional effects, $\vee$ allows us to represent a disjunctive precondition
- in Pre(o) and Eff(o), $\forall$ and $\exists$ represent universal quantification and existential quantification

Languages for planning

ADL language: example of the "BlocksWorld"

- ADL representation of operators (one is enough)

```
# pick block ?x which is on ?y (block, table), drop it on ?z (block, table)
Move-on :
     Name(move-on) = move-on(?x, ?y, ?z)
      Pre(move-on) = on(?x, ?y) ∧ free(?x) ∧ free(?z) ∧
           ≠(?x, ?z) ∧ ≠ (?y, ?z)
      Eff(move-on) = on(?x, ?z) ∧ ¬on(?x, ?y) ∧
             (≠(?y, Table) → free(?y)) ∧ (≠(?z, Table) → ¬free(?z))
```

Languages for planning

PDDL language

- Taking into account: durations, time-dependent effects, continuous resources, etc.
- typing
- equality constraints
- conditional effects
- disjunctive preconditions
- universal quantification
- updating state variables…

Languages for planning

The PDDL language: example of the “BlocksWorld”

- PDDL representation of operators (one is enough)

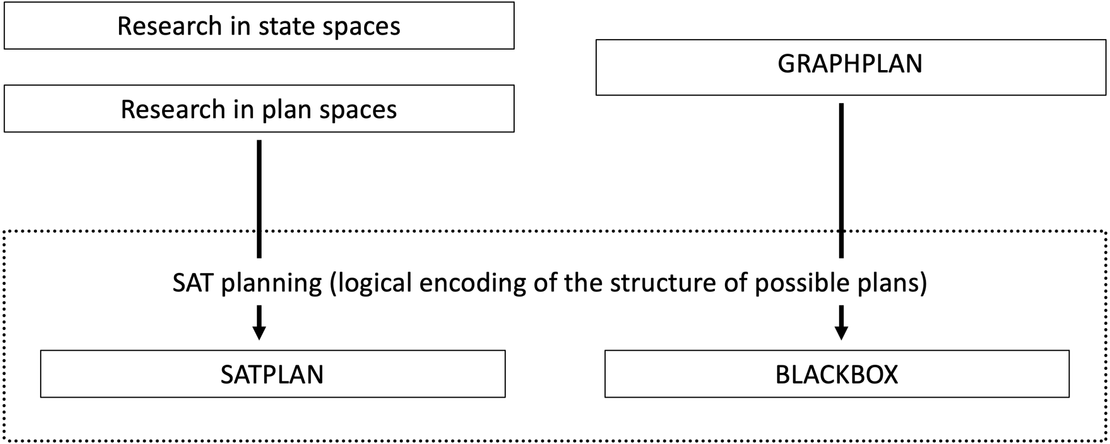
```
(define (domain blocksword)
     (: requirements :strips :equality :conditional-effects)
     (: predicates (on ?x ?y) (clear ?x))

     # pick block ?x which is on ?y (block, table),drop it on ?z (block, table)
     (: action puton
          : parameters (?x ?y ?z)
            : precondition
                (and (on ?x ?y)(clear ?x)(clear ?z)
                  (not (= ?y ?z)) (not (= ?x ?y))
                         (not (= ?x ?z)) (not (= ?x Table)))
            : effect
              (and (on ?x ?z) (not (on ?x ?y))
                     (when (not (= ?y Table)) (clear ?y))
                        (when (not (= ?z Table)) (not (clear ?z)))))
```

Section 3

Main algorithms for plan synthesis

Main algorithms for plan synthesis



Classification of interactions

- Positive interactions:
  - Multiple effects: action that produces several fluents: action a_1
  - Add/Add: $\exists f, f \in Add(a_1) \cap Add(a_2)$: fluent c
  - Add/Prec: $\exists f, f \in Add(a_1) \cap Prec(a_2)$: fluent d
- Negative interactions:
  - Contradictory effects: $\exists f, f \in Add(a1_{)} \cap Del(a_2)$: fluent e
  - Cross interactions: $\exists f, f \in Del(a_2) \cap Prec(a_1)$: fluent b

a		$+c$			$-b$
	$\boxed{a_1}$	$+d$	d	$\boxed{a_2}$	$+c$
b		$+e$			$-e$

Independent actions

- Two actions a1, a2 are independent (denoted $a_1 \# a_2$) if they have no negative interactions, i.e.:

$$
\begin{array}{lllll}
a_1 & x & \rightarrow & +y & -z \\
\\
a_2 & t & \rightarrow & +u & -v
\end{array}
$$

Independent actions

- Two actions a1, a2 are independent (denoted $a_1 \# a_2$) if they have no negative interactions, i.e.:
  - $Del(a_1) \cap (Prec(a_2) \cup Add(a_2)) = \emptyset$

$$
\begin{array}{ccccc}
a_1 & x & \rightarrow & +y & -z \\
 & & & & \\
a_2 & t & \rightarrow & +u & -v
\end{array}
$$

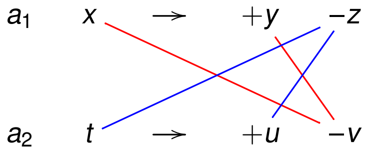
Independent actions

- Two actions a1, a2 are independent (denoted $a_1 \# a_2$) if they have no negative interactions, i.e.:
  - $Del(a_1) \cap (Prec(a_2) \cup Add(a_2)) = \emptyset$ and
  - $Del(a_2) \cap (Prec(a_1) \cup Add(a_1)) = \emptyset$

$$\begin{array}{cccccc} a_1 & x & \rightarrow & +y & -z \\ & & & & \\ a_2 & t & \rightarrow & +u & -v \end{array}$$

Independent actions

- Two actions a1, a2 are independent (denoted $a_1 \# a_2$) if they have no negative interactions, i.e.:
  - $Del(a_1) \cap (Prec(a_2) \cup Add(a_2)) = \emptyset$ and
  - $Del(a_2) \cap (Prec(a_1) \cup Add(a_1)) = \emptyset$



- Set of independent actions:
  - Q is a set of independent actions or independent set iff all the actions a_i which compose it are independent 2 by 2;
- Application of an independent set of actions (forward chaining):
  - an independent set Q is applicable to a state E iff: $\bigcup Prec(a_i) \in E$
  - the resulting state is the set of fluents:

$$E \uparrow Q = (E - \bigcup Del(a_i)) \; (\bigcup Add(a_i))$$

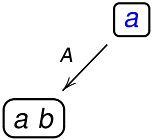
Algorithms for plan synthesis (state-spaces)

$A : a \rightarrow +b$
$B : a \rightarrow +c \; -a$
$C : b \; c \rightarrow +d$

$\boxed{a}$

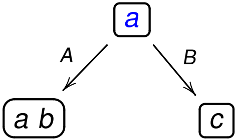
Algorithms for plan synthesis (state-spaces)

$A : a \rightarrow +b$
$B : a \rightarrow +c \; -a$
$C : b \; c \rightarrow +d$



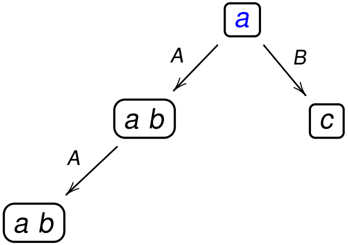
Algorithms for plan synthesis (state-spaces)

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$



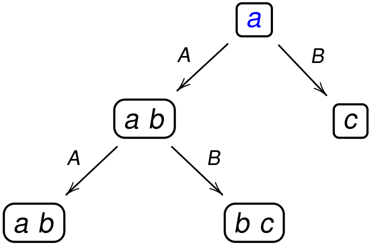
Algorithms for plan synthesis (state-spaces)

$$A : a \rightarrow +b$$
$$B : a \rightarrow +c\ -a$$
$$C : b\ c \rightarrow +d$$



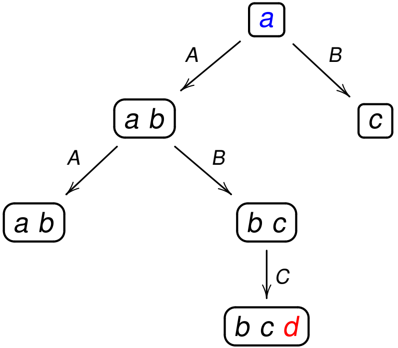
Algorithms for plan synthesis (state-spaces)

$$
\begin{array}{l}
A : a \rightarrow +b \\
B : a \rightarrow +c\, -a \\
C : b\ c \rightarrow +d
\end{array}
$$



Algorithms for plan synthesis (state-spaces)

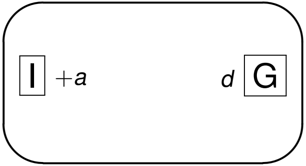
$A : a \to +b$
$B : a \to +c\ -a$
$C : b\ c \to +d$



Solution plan $\langle A, B, C \rangle$

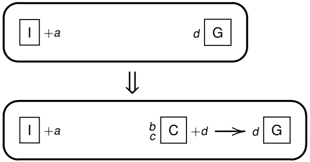
Algorithms for plan synthesis (plan-spaces)

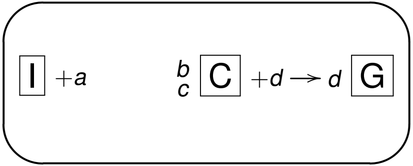
$$A : a \rightarrow +b$$
$$B : a \rightarrow +c\,-a$$
$$C : b\ c \rightarrow +d$$



Algorithms for plan synthesis (plan-spaces)

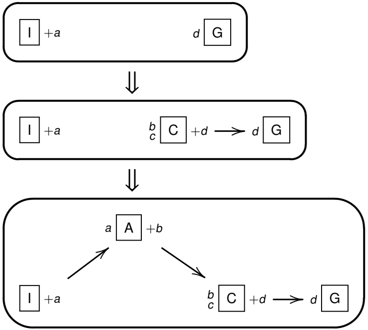
$$
\begin{array}{l}
A : a \rightarrow +b \\
B : a \rightarrow +c\ {-a} \\
C : b\ c \rightarrow +d
\end{array}
$$

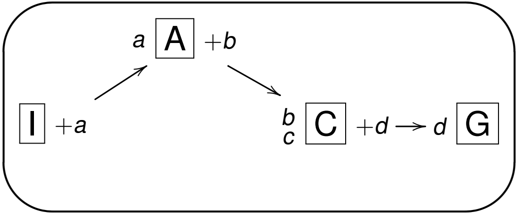




Algorithms for plan synthesis (plan-spaces)

$A : a \rightarrow +b$
$B : a \rightarrow +c \; -a$
$C : b \; c \rightarrow +d$



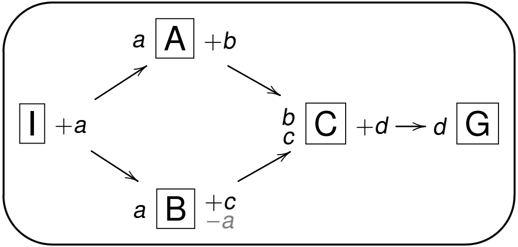


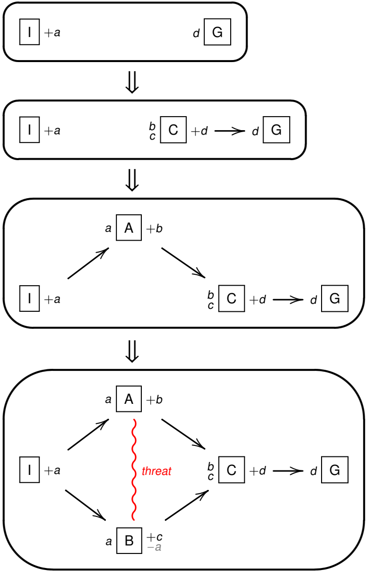
Algorithms for plan synthesis (plan-spaces)

$A : a \rightarrow +b$

$B : a \rightarrow +c \, -a$

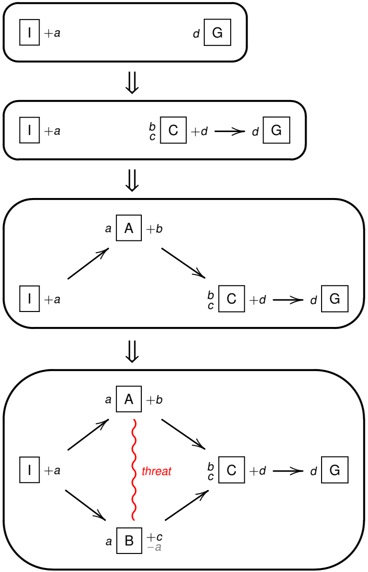
$C : b \; c \rightarrow +d$

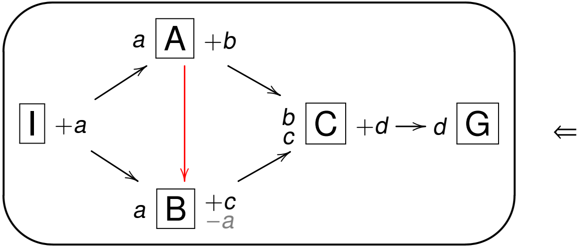




Algorithms for plan synthesis (plan-spaces)

$A : a \rightarrow +b$
$B : a \rightarrow +c \; -a$
$C : b \; c \rightarrow +d$

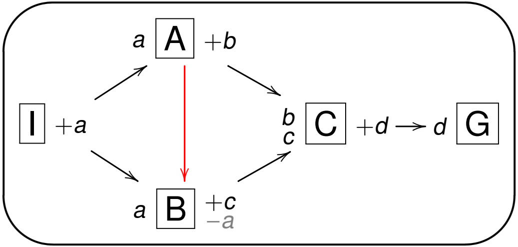




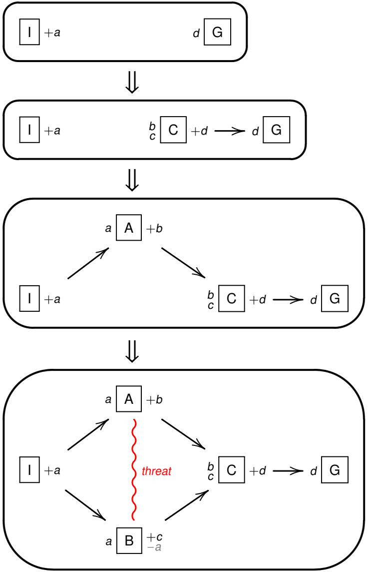
Algorithms for plan synthesis (plan-spaces)

$A : a \rightarrow +b$
$B : a \rightarrow +c\, -a$
$C : b\ c \rightarrow +d$

Solution plan = $\{Actions, Constraints\}$
Actions = $\{A, B, C\}$
Constraints = $\{(A, B), (A, C), (B, C)\}$
post-treated, gives: $\langle A, B, C\rangle$



$\Leftarrow$



Section 4

GRAPHPLAN

Principles of the planner GRAPHPLAN

- GRAPHPLAN *separates planning into two procedures*:
  - construction of the planning graph (polynomial complexity in time and space compared to the size of the problem data);
  - search for a potential solution in the subtree extracted from this graph (NP), which can be carried out by different methods.
- The graph *provides a lot of information* which can be used as *domain-independent heuristics* for classic methods (search in state spaces...), it can also be adapted to take into account resources and time.

Definitions

- In GRAPHPLAN, two actions at the same level in the graph are *mutually exclusive* (mutex) iff:
  - they are not independent or,
  - they have mutex preconditions at the previous level (so they cannot be triggered at the same time): $\exists(p,q) \in Prec(a_1) \times Prec(a_2)$, such that p and q are mutexes.
- Two fluents p and q are mutexes at level i iff all pairs of actions which produce them at this same level are mutexes (there is no pair of non-mutex actions which produce them at this level): $\forall a_1, a_2 / p \in Add(a_1), q \in Add(a_2)$, a_1 and a_2 mutexes.

Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$

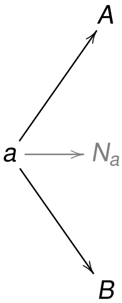
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\, -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$

a

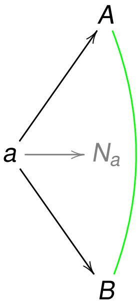
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



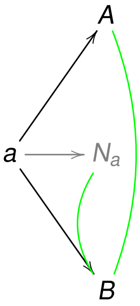
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



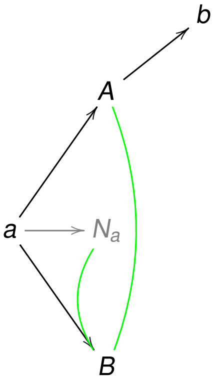
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



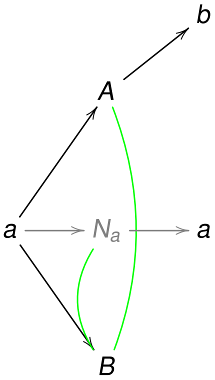
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c \; -a$
$C : b \; c \rightarrow +d$
$NoOps\{a, b, c, d\}$



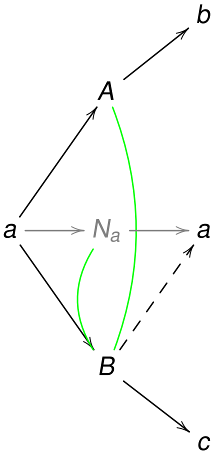
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



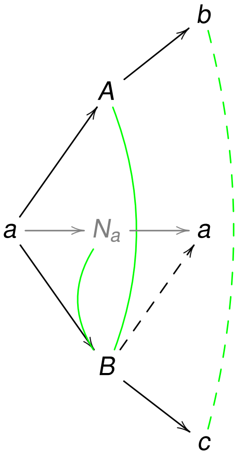
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



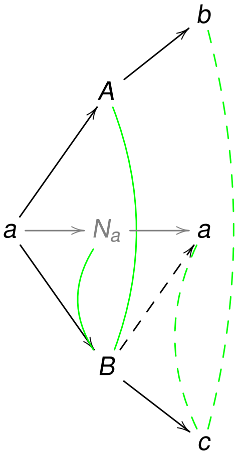
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



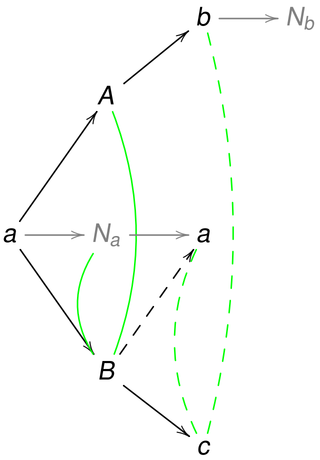
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ {-a}$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



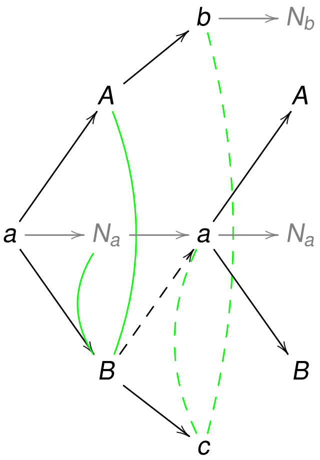
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ {-a}$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



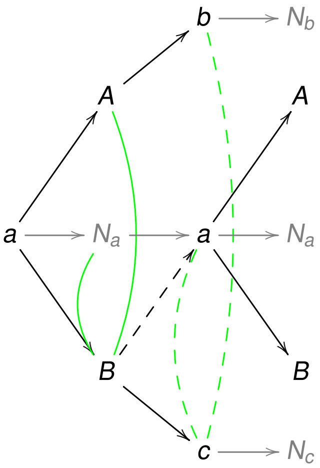
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



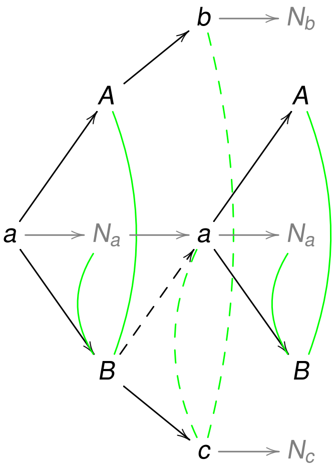
Algorithm of GRAPHPLAN

A : $a \rightarrow +b$
B : $a \rightarrow +c\ -a$
C : $b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



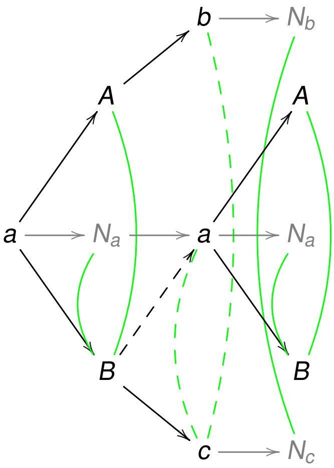
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



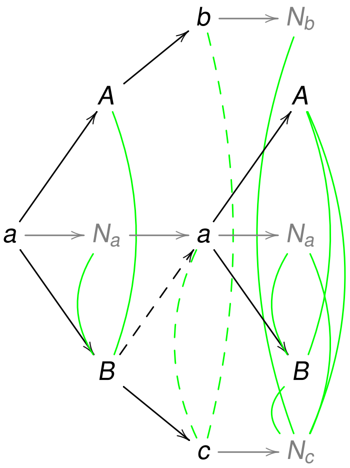
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c \; -a$
$C : b \; c \rightarrow +d$
$NoOps\{a, b, c, d\}$



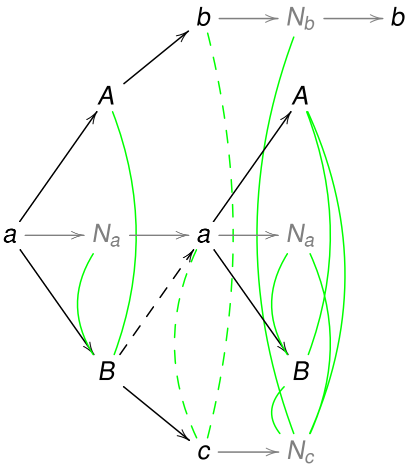
Algorithm of GRAPHPLAN

A : $a \rightarrow +b$
B : $a \rightarrow +c\ -a$
C : $b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



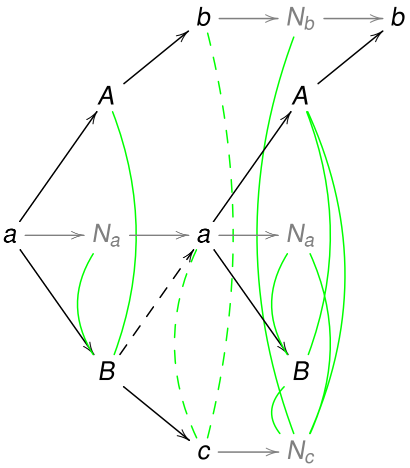
Algorithm of GRAPHPLAN

$A : a \to +b$
$B : a \to +c\ -a$
$C : b\ c \to +d$
$NoOps\{a, b, c, d\}$



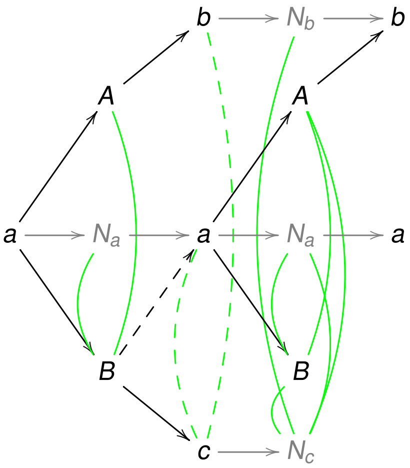
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ {-a}$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



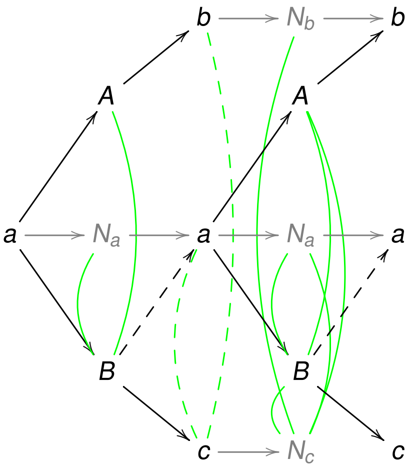
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ {-a}$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



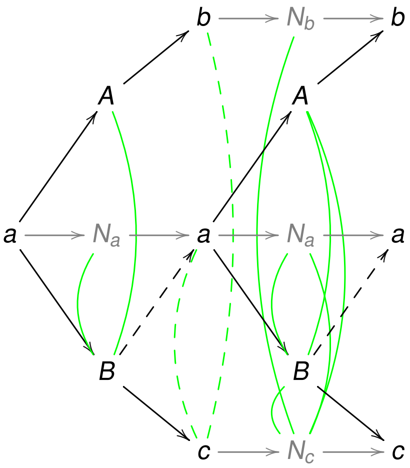
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c \; -a$
$C : b \; c \rightarrow +d$
$NoOps\{a, b, c, d\}$



Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



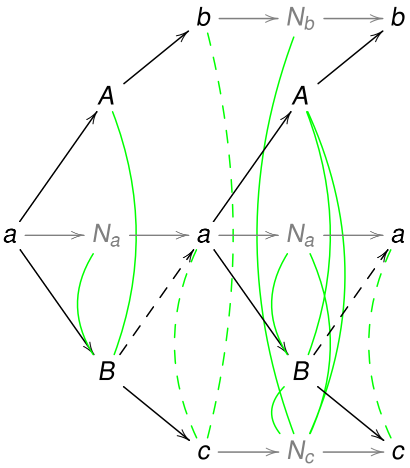
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$

$B : a \rightarrow +c\ -a$

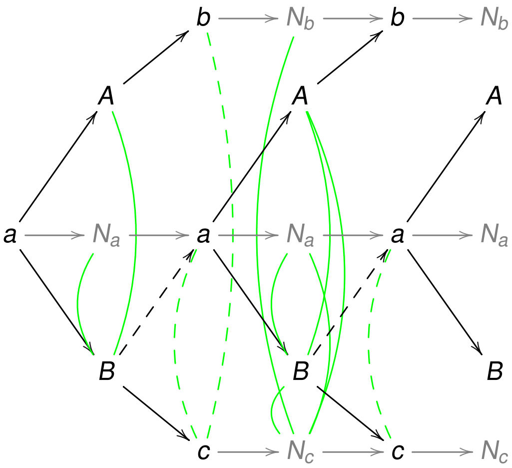
$C : b\ c \rightarrow +d$

$NoOps\{a, b, c, d\}$



Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ -a$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



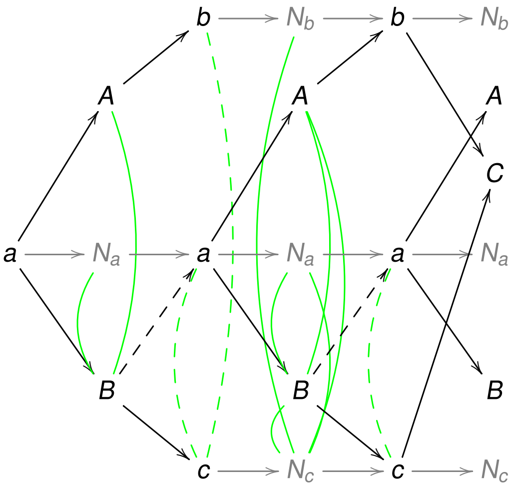
Algorithm of GRAPHPLAN

$A : a \to +b$

$B : a \to +c \, -a$

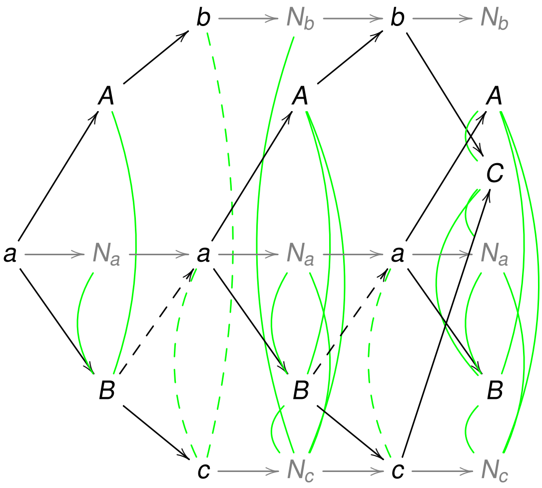
$C : b \ c \to +d$

$NoOps\{a, b, c, d\}$



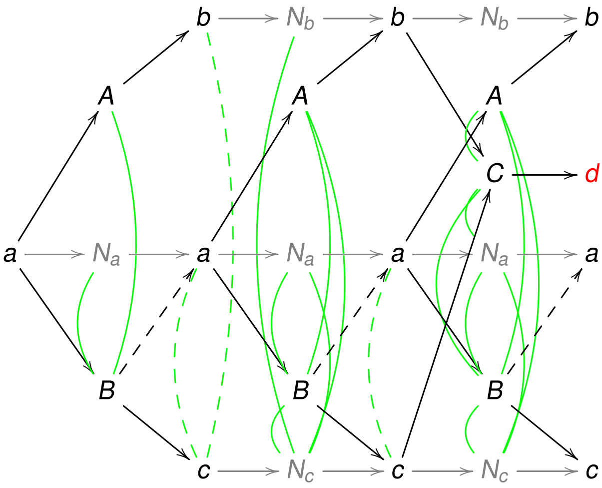
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c \, -a$
$C : b \; c \rightarrow +d$
$NoOps\{a, b, c, d\}$



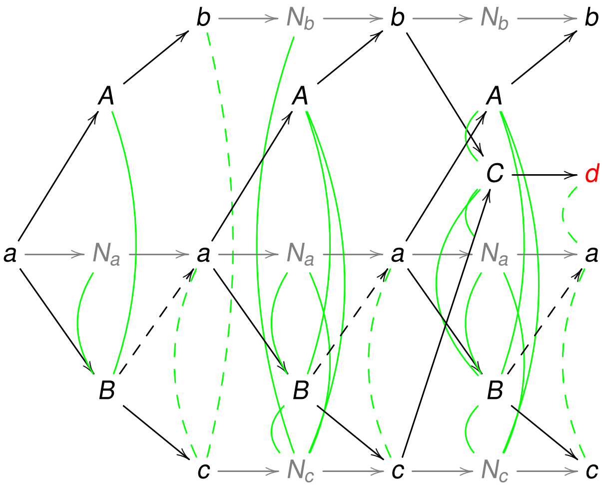
Algorithm of GRAPHPLAN

A : $a \rightarrow +b$
B : $a \rightarrow +c\ {-a}$
C : $b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



Algorithm of GRAPHPLAN

$A : a \rightarrow +b$
$B : a \rightarrow +c\ {-a}$
$C : b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



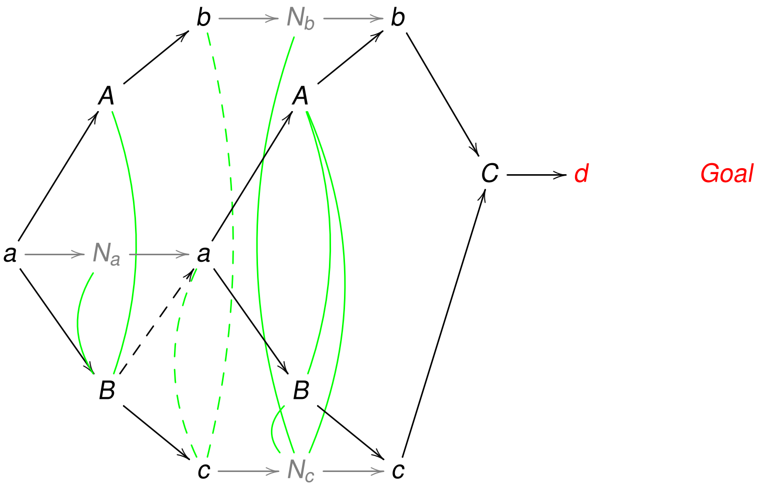
Algorithm of GRAPHPLAN

$A : a \rightarrow +b$

$B : a \rightarrow +c\ {-a}$

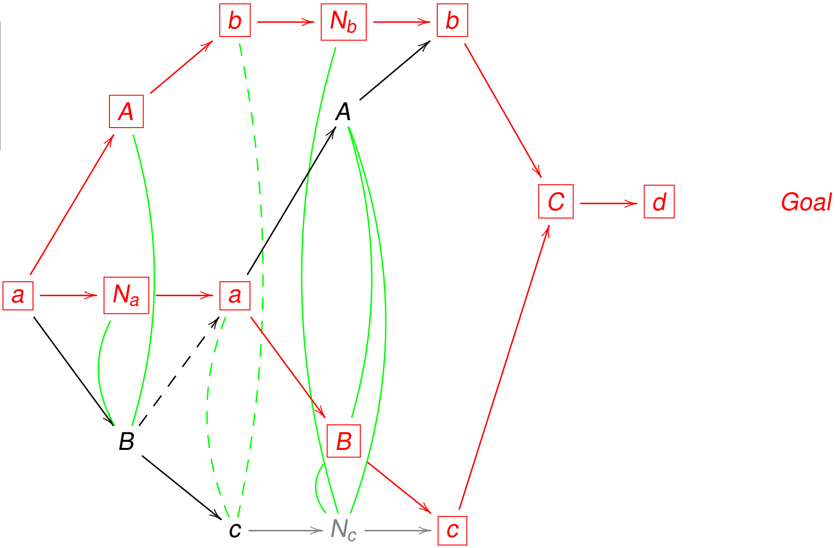
$C : b\ c \rightarrow +d$

$NoOps\{a, b, c, d\}$



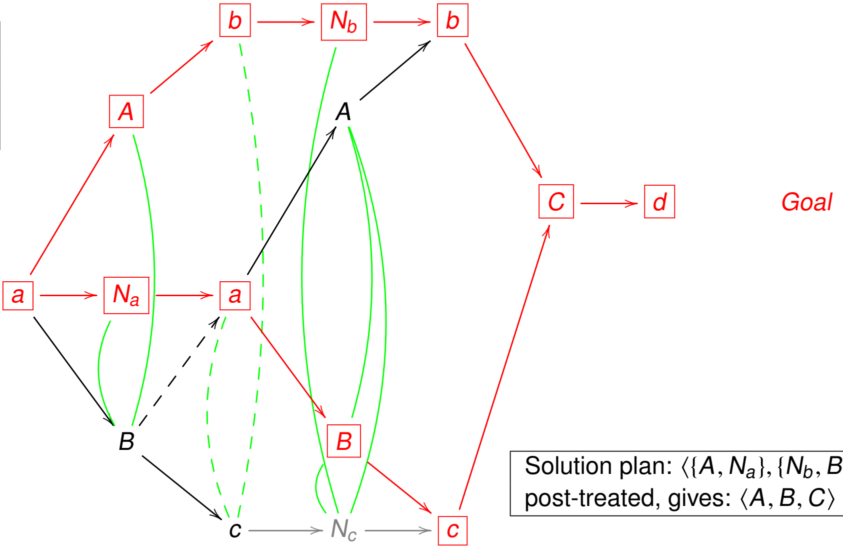
Algorithm of GRAPHPLAN

A : $a \rightarrow +b$
B : $a \rightarrow +c\, -a$
C : $b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



Algorithm of GRAPHPLAN

A : $a \rightarrow +b$
B : $a \rightarrow +c\, -a$
C : $b\ c \rightarrow +d$
$NoOps\{a, b, c, d\}$



Solution plan: $\langle \{A, N_a\}, \{N_b, B\}, \{C\} \rangle$
post-treated, gives: $\langle A, B, C \rangle$

Section 5

SATPLAN

SAT Encodings for Classical Planning

Several different encoding have been proposed:

- **State spaces encodings**
- **Plan spaces encodings**
- **Planning graph encodings**

In the sequel, we present the state spaces encoding with explanatory frame-axioms.

SAT Encodings for Classical Planning

$$S_0(\mathit{Init}) \blacktriangleright \boxed{x_1 \equiv S_1} \blacktriangleright \boxed{x_2 \equiv S_2} \blacktriangleright \boxed{x_3 \equiv S_3} \blacktriangleright \boxed{x_4 \equiv S_4} \blacktriangleright \boxed{x_5 \equiv S_5} \blacktriangleright \boxed{x_6 \equiv S_6} \blacktriangleright \boxed{x_7 \equiv S_7} \blacktriangleright S_8(\mathit{Goal})$$

Figure: Transitions of an 8-step plan in SAT encoding

Each step i is associated with a set of propositional variables $X_i = X_{A,i} \cup X_{F,i}$ where

- $X_{A,i} = \{a_i^1, a_i^2 \ldots, a_i^m\}$ is a set of propositional variables for actions;
- $X_{F,i} = \{f_i^1, f_i^2, \ldots, f_i^n\}$ is a set of propositional variables for fluents

SAT Encoding: Initial State and Goal

Initial state: $$\left(\bigwedge_{\mathbf{f} \in \mathbf{I}} \mathbf{f}_0\right) \wedge \left(\bigwedge_{\mathbf{f} \in \mathbf{F} \setminus \mathbf{I}} \neg \mathbf{f}_0\right)$$

Goal: $$\bigwedge_{\mathbf{f} \in \mathbf{G}} \mathbf{f}_{\mathbf{length}}$$

SAT Encoding: Conditions and Effects of Actions

$$\bigwedge_{i\in[1..length]} \bigwedge_{a\in O} \left(a_i \Rightarrow \left(\left(\bigwedge_{f\in Cond_a} f_{i-1} \right) \wedge \left(\bigwedge_{f\in Add_a} f_i \right) \wedge \left(\bigwedge_{f\in Del_a} (\neg f_i) \right) \right) \right)$$

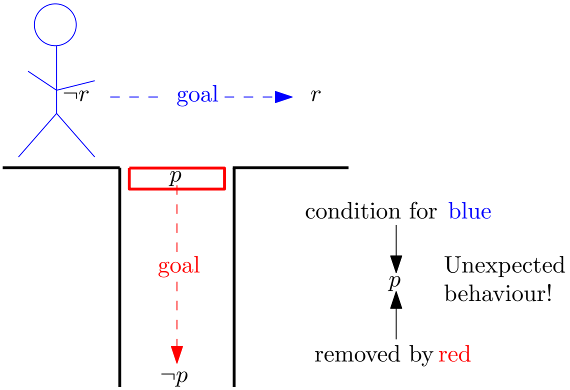
SAT Encoding: Explanatory Frame-Axioms

$$\bigwedge_{i \in [1..length]} \bigwedge_{f \in F} \left((\neg f_{i-1} \wedge f_i) \Rightarrow \left(\bigvee_{\substack{a \in O \\ f \in Add_a}} a_i \right) \right)$$

$$\bigwedge_{i \in [1..length]} \bigwedge_{f \in F} \left((f_{i-1} \wedge \neg f_i) \Rightarrow \left(\bigvee_{\substack{a \in O \\ f \in Del_a}} a_i \right) \right)$$

SAT Encoding: Negative Interactions (Mutex)

$$\bigwedge_{i\in[1..length]} \bigwedge_{a1\in O} \bigwedge_{f\in Cond_{a1}} \bigwedge_{\substack{a2\in O\\(a1\neq a2)\\(f\in Del_{a2})}} (\neg a1_i \vee \neg a2_i)$$



Part 3

Epistemic planning with DEL

Table of contents

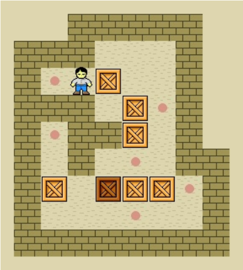
Section 6

Motivation

Overview I

Classical planning:

- One agent.
- Completely known and observable environment.
- Deterministic.
- Example: Sokoban



Carloseow at English Wikipedia, CC BY 3.0, via Wikimedia Commons

Overview II

Epistemic planning:

- Several agents.
- Partially observable environment.
- Coordination sometimes necessary.
- Still deterministic.
- Examples:
  - “Epistemic” blocks world.
  - Cooperative card games.
  - Several robots in a warehouse with walls.

Hanabi

Mannivu, CC BY-SA 4.0, via Wikimedia Commons

A cooperative task

Pico-Hanabi[1] (modified). Three cards of the same color. Two players. No tokens.

Initial state:

- One card for each player + one card on the deck.
- Players cannot see their own cards.
- Each player can see all other player's cards.

Turn-based.

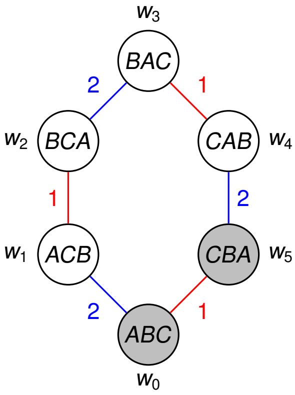
Actions:

- Make an announcement about the partner's cards (only once during the whole game).
- Try to play a card on the table (own card, or from the deck):
  - If the card is on the right order, it's placed on the table and the player gets the other card.
  - Otherwise, the game is over (and lost).

Goal: Place all three cards on the table **on the right order**.

[1] [Engesser et al., 2021]

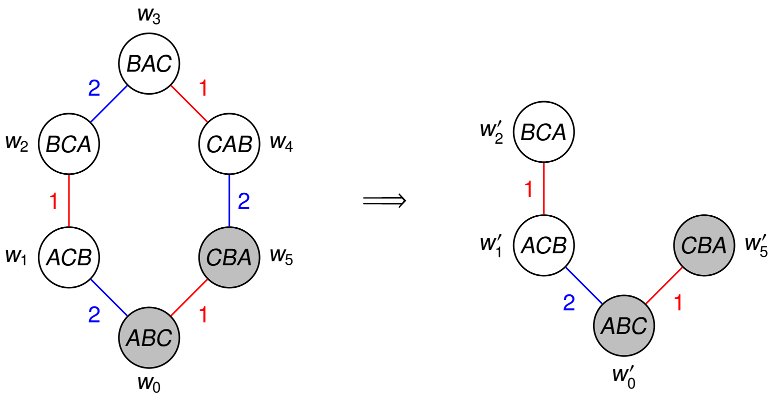
Initial epistemic state



- Agent 1 is the one who plans.
- 1 sees that 2 has card B.
- 1 does not know her hand, nor the deck.
- 1 knows that if she plays a card, they can loose the game.

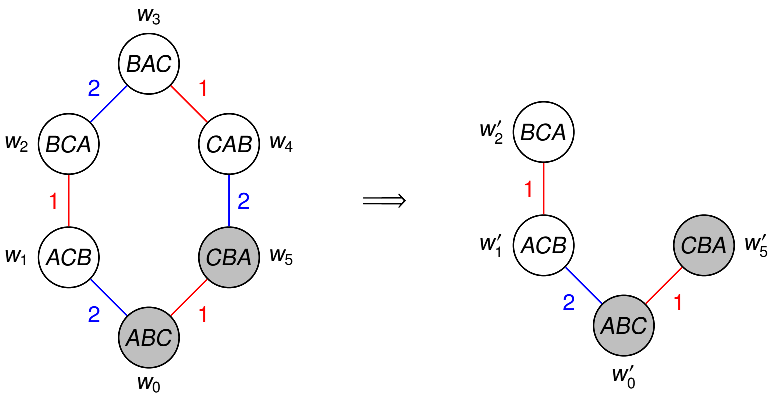
First move

What happens if 1 announces “2 does not have card A”?



First move

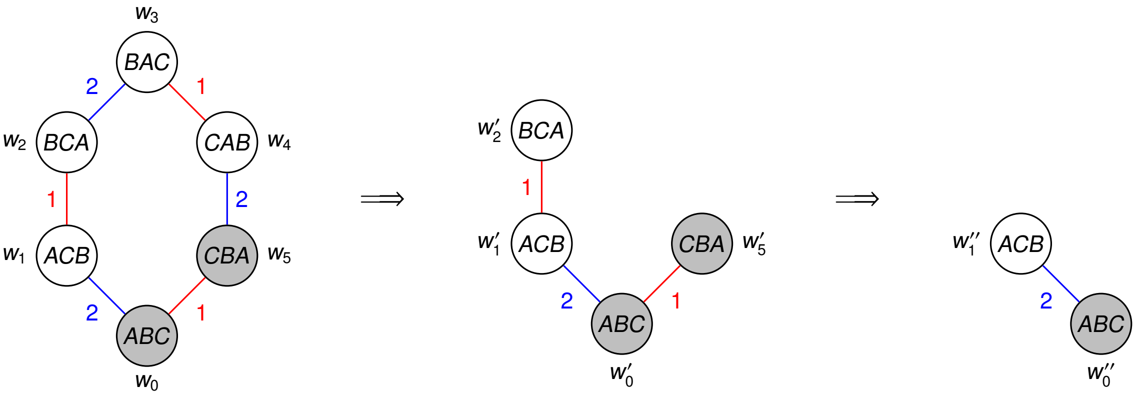
What happens if 1 announces “2 does not have card A”?



- The states where 2 has card A are removed.
- 2 learns that she should not play her card,
- but 2 still does not know her card.

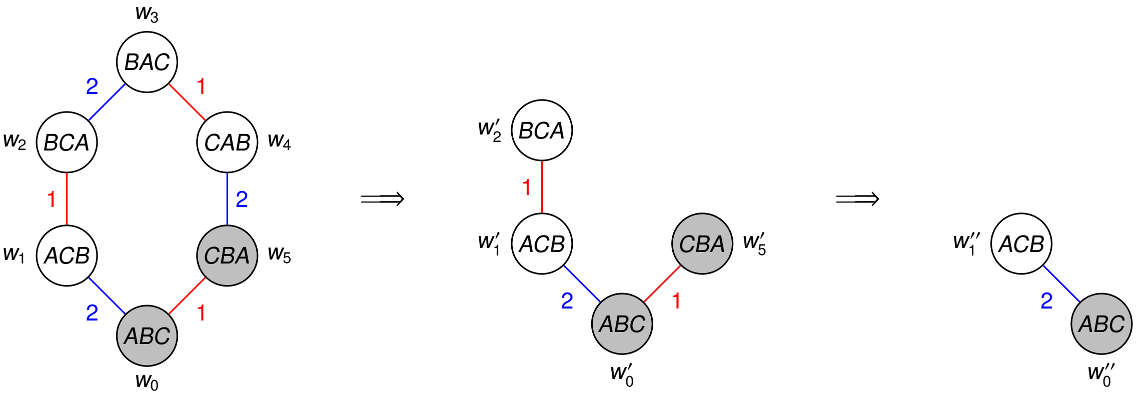
Second move

Then, what happens if 2 announces “1 has card A”?



Second move

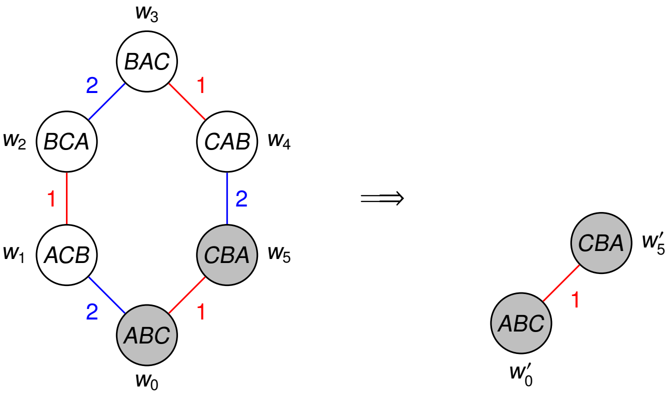
Then, what happens if 2 announces “1 has card A”?



- The states where 1 does not have card A are removed.
- 1 learns that she can play her card,
- but, on the next move, 2 must take a random decision.

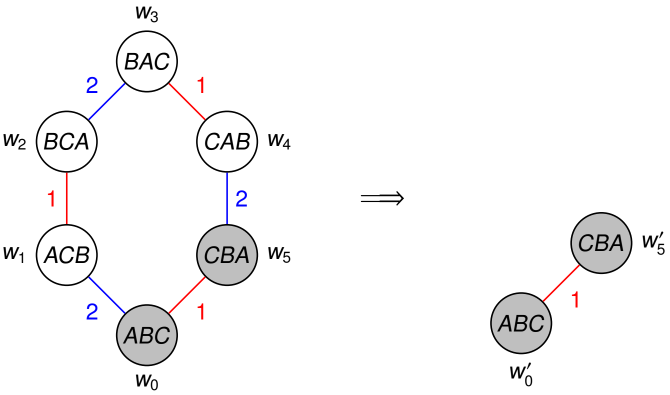
A better first move

What if 1 announces “agent 2 has card B”?



A better first move

What if 1 announces “agent 2 has card B”?



- The states where 2 does not have card B are removed.
- 2 learns her hand.
- Now, if 2 plays well, they can win the game…

Epistemic planning

Epistemic planning = planning + theory of mind (ToM).[2]

Definition (Epistemic planning)

A planning task is a triple $T = \langle s_0, \mathbb{A}, \gamma \rangle$ where:

- s_0: initial epistemic state;
- $\mathbb{A}$: a finite set of epistemic actions;
- γ: an epistemic formula describing the goal.

Definition (Solution)

A solution of a (sequencial) planning task $T = \langle s_0, \mathbb{A}, \gamma \rangle$ is a sequence of actions $\alpha, \ldots, \alpha_n$ of $\mathbb{A}$ such that, for all $1 \leq k \leq n$, α is applicable in $s_0 \otimes \alpha_1 \otimes \ldots \otimes \alpha_{k-1}$ and:

$$s_0 \otimes \alpha_1 \otimes \ldots \otimes \alpha_n \models \gamma$$

[2] [Bolander et al., 2020]

Representation choice

Syntactic approach
States are represented by formulas.

Semantic approach.
States are represented by epistemic models (Kripke structures).

Explicit approach.
The set of states is given (eg.: ATEL[3], CSL[4]).

Implicit approach.
The set of states is induced by the initial state and the set of actions (eg.: STRIPS/PDDL).

Epistemic planning based on DEL uses the semantic and implicit approaches.[5]

[3] [van der Hoek and Wooldridge, 2002]
[4] [Jamroga and Aagotnes, 2007]
[5] [Bolander and Andersen, 2011]

Section 7

Epistemic logic

Syntax

Vocabulary:

- $\mathbb{P}$: a countable non-empty set of propositional variables.
- $\mathbb{A}$: a finite non-empty set of agents.

Language $\mathcal{L}$:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \land \varphi \mid K_i\varphi$$

where $p \in \mathbb{P}$ and $i \in \mathbb{N}$.

Abbreviation:

- $\overline{K}_i\varphi \stackrel{\text{def}}{=} \neg K_i\neg\varphi$

Meanings:

- $K_i\varphi$: agent i knows that φ.
- $\overline{K}_i\varphi$: agent i considers it possible that φ.

Semantics I

Definition (Epistemic model)

A (Kripke) structure $\mathcal{M} = \langle W, R, V \rangle$, where:

- W is a set of possible worlds.
- $R : \mathbb{N} \to (W \times W)$ associates an accessibility relation to each agent.
- $V : \mathbb{P} \to 2^W$ associates a set of states to each propositional variable.

Each accessibility relation is an equivalence class, i.e.:

- Reflexive: $\langle w, w \rangle \in R(i)$.
- Euclidean: $\langle w, w' \rangle, \langle w, w'' \rangle \in R(i)$ implies $\langle w', w'' \rangle \in R(i)$.

Semantics II

Definition (Epistemic state – internal approach)

A pair $s = \langle \mathcal{M}, W_d \rangle$, where:

- $\mathcal{M}$: an epistemic model.
- $W_d \subseteq W$: a set of possible worlds called 'designated world'.

The set of designated worlds:

- Corresponds to the world considered possible by the planning agent.
- Contains the actual world.
- In the initial state, it coincides with the set of accessible worlds from the actual world for the planning agent.

Semantics III

Definition (Satisfaction relation)

$$\mathcal{M}, W_d \models \varphi \quad \text{iff} \quad \mathcal{M}, w \models \varphi, \text{ for all } w \in W_d$$

$$\begin{array}{lll}
\mathcal{M}, w \models \top & & \\
\mathcal{M}, w \models p & \text{iff} & w \in V(p) \\
\mathcal{M}, w \models \neg\varphi & \text{iff} & \mathcal{M}, w \not\models \varphi \\
\mathcal{M}, w \models \varphi_1 \wedge \varphi_2 & \text{iff} & \mathcal{M}, w \models \varphi_1 \text{ and } \mathcal{M}, w \models \varphi_2 \\
\mathcal{M}, w \models K_i\varphi & \text{iff} & \mathcal{M}, w' \models \varphi, \text{ for all } w' \in W \text{ s.t. } \langle w, w' \rangle \in R(i)
\end{array}$$

Meanings:

- $\mathcal{M}, W_d \models \varphi$: the planning agent knows that φ at planning time.
- $\mathcal{M}, w \models K_i\varphi$: agent i knows that φ at execution time.

Example: Pico-Hanabi

- Agents: 1 and 2
- Propositional variables:
  - $p_{A,1}$: "1 has card A"

    ...
  - $p_{C,e}$: "C is in the deck"

Example: Pico-Hanabi

- Agents: 1 and 2
- Propositional variables:
  - $p_{A,1}$: "1 has card A"

    ...
  - $p_{C,e}$: "C is in the deck"
- Abbreviations:
  - $A_1 \overset{\text{def}}{=} (p_{A,1} \wedge \neg p_{A,2} \wedge \neg p_{A,e})$: "A is only with player 1"

    ...
  - $C_e \overset{\text{def}}{=} (p_{C,e} \wedge \neg p_{C,1} \wedge \neg p_{C,2})$: "C is only on the deck"
  - $ABC \overset{\text{def}}{=} A_1 \wedge B_2 \wedge C_e$

    ...
  - $CBA \overset{\text{def}}{=} C_1 \wedge B_2 \wedge A_e$

Example: Pico-Hanabi

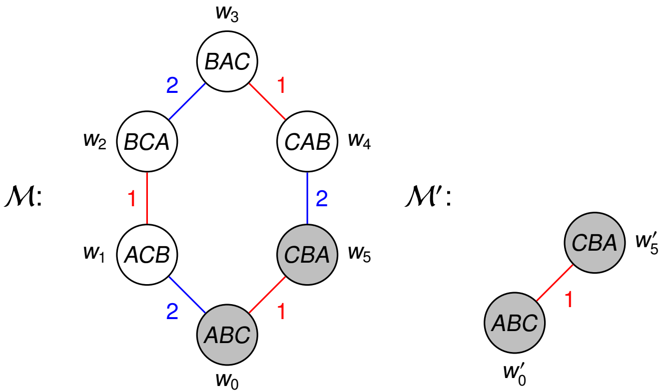
- Agents: 1 and 2
- Propositional variables:
  - $p_{A,1}$: "1 has card A"

    ...
  - $p_{C,e}$: "C is in the deck"
- Abbreviations:
  - $A_1 \stackrel{\text{def}}{=} (p_{A,1} \wedge \neg p_{A,2} \wedge \neg p_{A,e})$: "A is only with player 1"

    ...
  - $C_e \stackrel{\text{def}}{=} (p_{C,e} \wedge \neg p_{C,1} \wedge \neg p_{C,2})$: "C is only on the deck"
  - $ABC \stackrel{\text{def}}{=} A_1 \wedge B_2 \wedge C_e$

    ...
  - $CBA \stackrel{\text{def}}{=} C_1 \wedge B_2 \wedge A_e$
- A desirable state (some kind of "intermediate goal"):
  - $H_1 \stackrel{\text{def}}{=} K_1 A_1 \vee K_1 B_1 \vee K_1 C_1$: "1 knows her own hand"
  - $H_2 \stackrel{\text{def}}{=} K_2 A_2 \vee K_2 B_2 \vee K_2 C_2$: "2 knows her own hand"

Example: Pico-Hanabi



$$(\mathcal{M}, w_0) \models \overline{K}_1 A_1 \wedge \overline{K}_1 C_1$$
$$(\mathcal{M}, w_0) \models \overline{K}_2 B_2 \wedge \overline{K}_2 C_2$$
$$(\mathcal{M}, \{w_0, w_5\}) \models \neg H_1 \wedge \neg H_2$$

$$(\mathcal{M}', \{w'_0, w'_5\}) \models \overline{K}_1 A_1 \wedge \overline{K}_1 C_1$$
$$(\mathcal{M}', w'_0) \models K_2 ABC$$
$$(\mathcal{M}', w'_5 \models K_2 CBA$$
$$(\mathcal{M}', \{w'_0, w'_5\}) \models \neg H_1 \wedge H_2$$

Remark

Epistemic logic permits the verification of the epistemic states of the system.

However, the execution of an action in an epistemic state is not always evident.

For example, what is the effect of the following STRIPS action in the initial state of Pico-Hanabi?

$$\begin{aligned} \text{PRE} &: K_1 A_1 \\ \text{ADD} &: \emptyset \\ \text{DEL} &: \emptyset \end{aligned}$$

Remark

Epistemic logic permits the verification of the epistemic states of the system.

However, the execution of an action in an epistemic state is not always evident.

For example, what is the effect of the following STRIPS action in the initial state of Pico-Hanabi?

$$\begin{aligned} \text{PRE} &: K_1 A_1 \\ \text{ADD} &: \emptyset \\ \text{DEL} &: \emptyset \end{aligned}$$

This action does not seem useful (because there is no physical effect).

However, this a communication action!

Remark

Epistemic logic permits the verification of the epistemic states of the system.

However, the execution of an action in an epistemic state is not always evident.

For example, what is the effect of the following STRIPS action in the initial state of Pico-Hanabi?

$$\text{PRE} : K_1 A_1$$
$$\text{ADD} : \emptyset$$
$$\text{DEL} : \emptyset$$

This action does not seem useful (because there is no physical effect).

However, this a communication action!

In addition, we want to be able to encode partially observable actions.

Section 8

Dynamic epistemic logics

Public announcements

Language $\mathcal{L}$:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid \langle !\varphi\rangle\varphi$$

where $p \in \mathbb{P}$ and $i \in \mathbb{N}$.

Abbreviation:

- $[!\psi]\varphi \stackrel{\text{def}}{=} \neg\langle !\psi\rangle\neg\varphi$

Meanings:

- $\langle !\psi\rangle\varphi$: ψ is true and φ is true after the announcement of ψ.
- $[!\psi]\varphi$: if ψ is true, then φ is true after the announcement of ψ.

Example:

- $\langle !p\rangle K_i p$: p is true and i knows that p after the announcement of p.

Semantics

Update: $(\mathcal{M}, W_d) \otimes !\varphi = (\mathcal{M}', W_d')$, where:[6]

- $\mathcal{M}' = \langle W', R', V' \rangle$
- $W' = \{w \mid (\mathcal{M}, w) \models \varphi\}$
- $R'(i) = R(i) \cap (W' \times W')$
- $V'(p) = V(p) \cap W'$
- $W_d' = W_d \cap W'$

That is, remove the worlds where φ is false.

[6] [Plaza, 1989, Plaza, 2007]

Semantics

Update: $(\mathcal{M}, W_d) \otimes !\varphi = (\mathcal{M}', W'_d)$, where:[6]

- $\mathcal{M}' = \langle W', R', V' \rangle$
- $W' = \{w \mid (\mathcal{M}, w) \models \varphi\}$
- $R'(i) = R(i) \cap (W' \times W')$
- $V'(p) = V(p) \cap W'$
- $W'_d = W_d \cap W'$

That is, remove the worlds where φ is false.

Satisfaction relation:

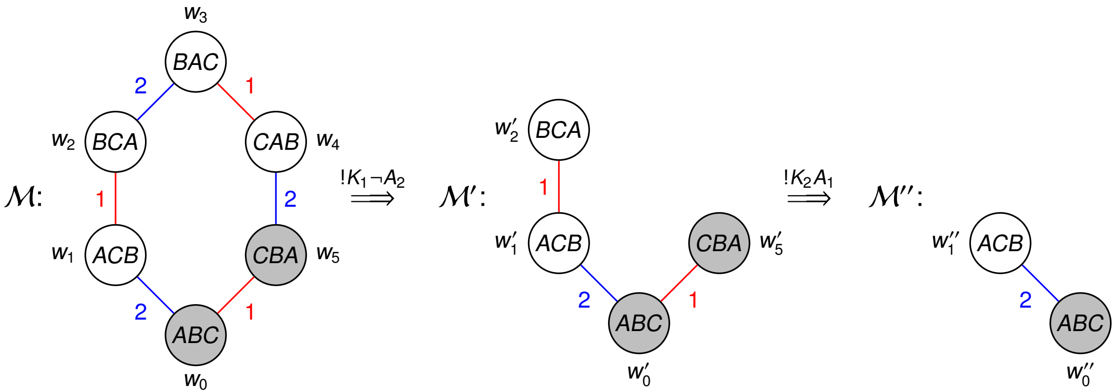
$$
\begin{array}{lcl}
(\mathcal{M}, w) \models \langle !\psi \rangle \varphi & \text{iff} & (\mathcal{M}, w) \models \psi \text{ and } (\mathcal{M}, w) \otimes !\psi \models \varphi \\
(\mathcal{M}, w) \models [!\psi] \varphi & \text{iff} & (\mathcal{M}, w) \models \psi \text{ implies } (\mathcal{M}, w) \otimes !\psi \models \varphi
\end{array}
$$

[6] [Plaza, 1989, Plaza, 2007]

Example: Pico-Hanabi

1 announces “2 does not have card A” (the bad move)

2 announces “1 has card A”



$(\mathcal{M}, \{w_0, w_5\}) \models [!K_1\neg A_2][!K_2A_1](H_1 \wedge \neg H_2)$

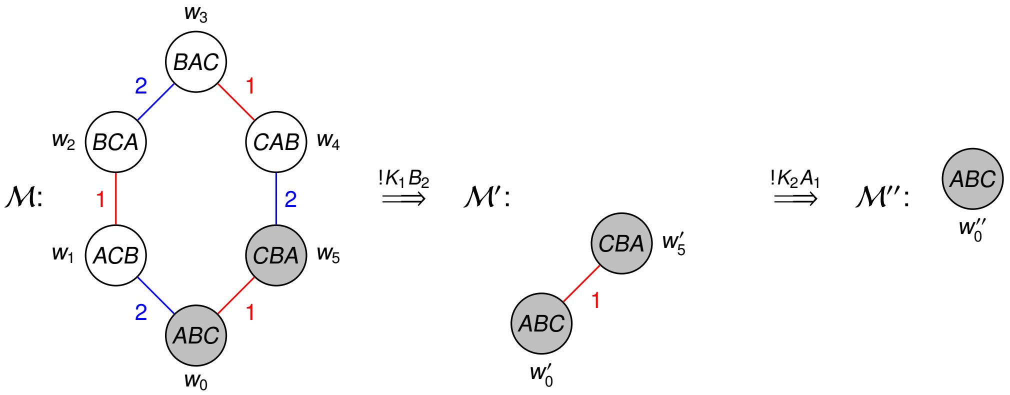
$(\mathcal{M}', \{w'_0\}) \models [!K_2A_1](H_1 \wedge \neg H_2)$

$(\mathcal{M}'', \{w''_0\}) \models H_1 \wedge \neg H_2$

$(\mathcal{M}'', \{w''_0\}) \models K_1A_1 \wedge \neg K_2B_2$

Example: Pico-Hanabi

1 announces “2 has card B” (the good move)
2 announces “1 has card A”.



$(\mathcal{M}, \{w_0, w_5\}) \models [!K_1B_2][!K_2A_1](H_1 \wedge H_2)$ $\quad (\mathcal{M}'', \{w''_0, w''_5\}) \models H_1 \wedge H_2$

$(\mathcal{M}', \{w'_0, w'_5\}) \models [!K_2A_1](H_1 \wedge H_2)$ $\quad (\mathcal{M}'', \{w''_0\}) \models K_1A_1 \wedge K_2B_2$

Reasoning methods

- Reduction axioms (sub-optimal):

$$\langle !\psi\rangle p \leftrightarrow (\psi \wedge p)$$
$$\langle !\psi\rangle \neg\varphi \leftrightarrow (\psi \wedge \neg\langle !\psi\rangle\varphi)$$
$$\langle !\psi\rangle(\varphi_1 \vee \varphi_2) \leftrightarrow (\langle !\psi\rangle\varphi_1 \vee \langle !\psi\rangle\varphi_2)$$
$$\langle !\psi\rangle \hat{K}_i\varphi \leftrightarrow (\psi \wedge \hat{K}_i\langle !\psi\rangle\varphi)$$

- Optimal reduction[7]
- Tableaux[8]

[7] [Lutz, 2006]
[8] [Balbiani et al., 2010]

Assignments

Addition of assignments to the language:[9]

- $\langle \sigma \rangle \varphi$: φ is true after the assignment σ.

where:

$$\sigma : \mathbb{P} \to \mathcal{L}$$

[9] [van Ditmarsch et al., 2005]

Assignments

Addition of assignments to the language:[9]

- $\langle\sigma\rangle\varphi$: φ is true after the assignment σ.

where:

$$\sigma : \mathbb{P} \to \mathcal{L}$$

Update: $(\mathcal{M}, W_d) \otimes \sigma = (\mathcal{M}', W'_d)$, where:

- $\mathcal{M}' = \langle W', R', V' \rangle$
- $W' = W$
- $R'(i) = R(i)$
- $V'(p) = \{w \mid \mathcal{M}, w \models \sigma(p)\}$
- $W'_d = W_d \cap W'$

[9] [van Ditmarsch et al., 2005]

Reasoning methods

- Reduction axioms (sub-optimal):

$$\langle\sigma\rangle p \leftrightarrow (p)\sigma$$
$$\langle\sigma\rangle\neg\varphi \leftrightarrow \neg\langle\sigma\rangle\varphi$$
$$\langle\sigma\rangle(\varphi_1 \vee \varphi_2) \leftrightarrow (\langle\sigma\rangle\varphi_1 \vee \langle\sigma\rangle\varphi_2)$$
$$\langle\sigma\rangle K_i\varphi \leftrightarrow K_i\langle\sigma\rangle\varphi$$

- Optimal reduction[10]

[10] [van Ditmarsch et al., 2012]

Events

It is possible to encode STRIPS actions with public announcements and assignments.

However, this complicates the task for the user.

It is simpler to create actions that have both announcements and assignments together.

Events

An event is a structure $e = \langle \mathrm{pre}(e), \mathrm{eff}(e) \rangle$, where:

- $\mathrm{pre}(e) \in \mathcal{L}$: the event pre-condition.
- $\mathrm{eff}(e) \in (\mathbb{P} \to \mathcal{L})$: the event effects.

Update: $(\mathcal{M}, W_d) \otimes e = (\mathcal{M}', W_d')$, where:

- $\mathcal{M}' = \langle W', R', V' \rangle$
- $W' = \{w \mid \mathcal{M}, w \models \mathrm{pre}(e)\}$
- $R'(i) = R(i) \cap (W' \times W')$
- $V'(p) = \{w \mid \mathcal{M}, w \models \sigma(p)\} \cap W'$
- $W_d' = W_d \cap W'$

Therefore, we now have public announcements and assignments together.

Applicability and coordination

Definition (Applicability)

An action α is applicable for agent *i* in a state *s* iff for each designated world *w* there is a designated event *e* such that $w \models \mathrm{pre}(e)$.

Definition (Implicit coordination)

Each action of the event must be applicable for the acting agent.

STRIPS actions

Events permit the encoding of STRIPS actions.

Action:

$$\begin{aligned} \text{PRE} &: \varphi \\ \text{ADD} &: p \\ \text{DEL} &: q \end{aligned}$$

Encoding:

$$\begin{aligned} e &= \langle \text{pre}(e), \text{eff}(e) \rangle \\ \text{pre}(e) &= \varphi \\ \text{eff}(e) &= p \leftarrow \top, q \leftarrow \bot \end{aligned}$$

Therefore, an action without physical effect is a public announcement!

Partially observable actions

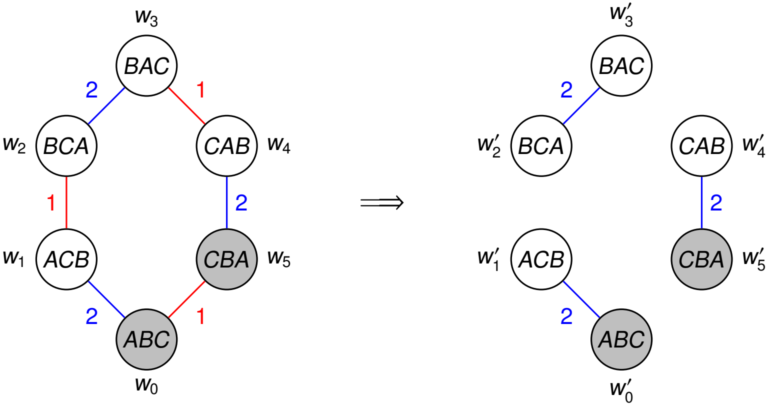
How to encode (semi-) private actions?

E.g.: 1 peeks.

Partially observable actions

How to encode (semi-) private actions?

E.g.: 1 peeks.



- At planning time:
  - 1 and 2 do not know their hands, nor the deck.
- At execution time:

Event models

Definition (Event models)

A (Kripke) structure $\mathcal{E} = \langle E, Q, \text{pre}, \text{eff} \rangle$, where:[11]

- E: set of events.
- $Q : \mathbb{N} \to (E \times E)$: associates a accessibility relation to each agent.
- $\text{pre} : E \to \mathcal{L}$: associates a formula to each event (pre-condition).
- $\text{eff} : E \to (\mathbb{P} \to \mathcal{L})$: associates an assignment to each event (effects).

As before, each accessibility relation is an equivalence relation.

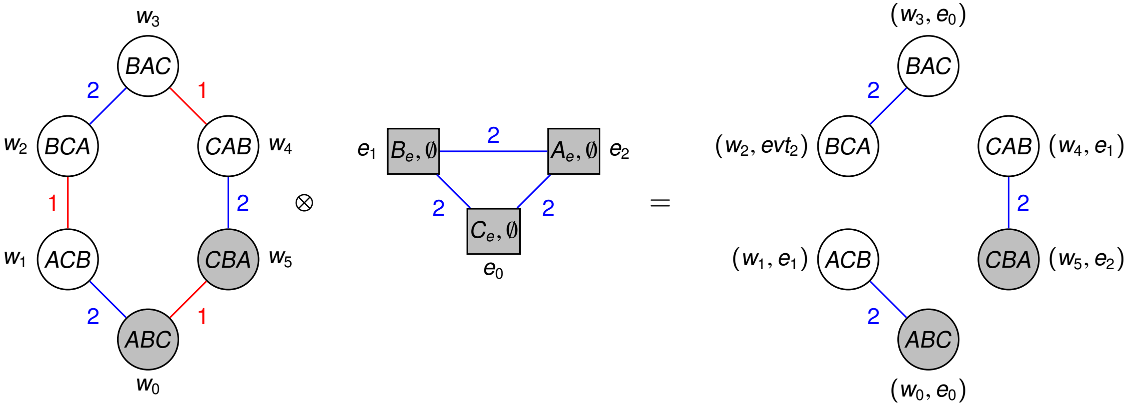
Update: $(\mathcal{M}, W_d) \otimes (\mathcal{E}, E_d) = (\mathcal{M}', W'_d)$, where:

- $W' = \{(w, e) \mid \mathcal{M}, w \models \text{pre}(e)\}$
- $R'(i) = \{\langle (w, e), (w', e') \rangle \mid \langle w, w' \rangle \in R(i) \text{ and } \langle e, e' \rangle \in Q(i)\}$
- $V'(i) = \{(w, e) \mid \mathcal{M}, w \models \text{eff}(e)(p)\} \cap W'$
- $W'_d = \{(w, e) \in W_d \times E_d\} \cap W'$

[11] [Baltag et al., 1998, Baltag and Moss, 2004, van Ditmarsch et al., 2007]

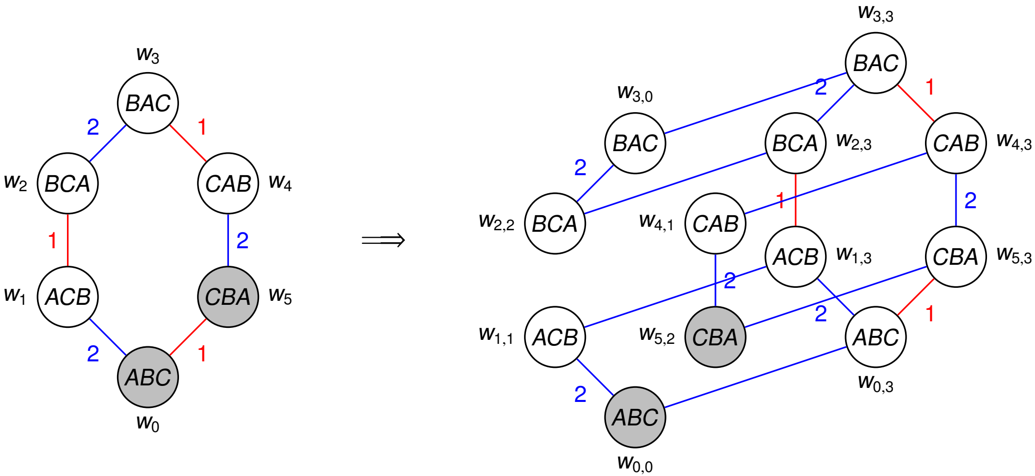
Partially observable action

Agent 1 peeks (to see the card on the deck).



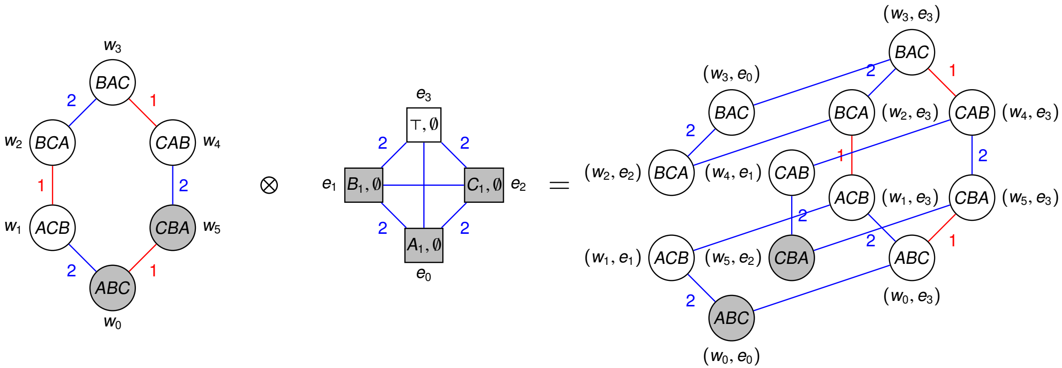
Private action

2 quits the room. During that period, 1 sees her own hand, but agent 2 suspects that 1 did that.[12]



[12] Agent 2 must suspects of the result, otherwise we get out from the logic of “knowledge”.

Private action



This kind of action can duplicate the size of the model.

This is why computational complexity of epistemic planning is high, when it is decidable.

Reasoning methods

- Reduction[13]
- Tableaux[14]
- Symbolic model checking[15]

[13] [van Benthem et al., 2006]
[14] [Aucher and Schwarzentruber, 2013]
[15] [van Benthem et al., 2018, Gamblin et al., 2022]

Section 9

Some open questions

Decidability

Epistemic planning is undecidable in K_n, KT_n, $K4_n$, $K45_n$, $KT4_n$ et $KT5_n$.[16]

Recently, several fragments have been studied:[17]

	without eff	with eff
$d = 0$	PSPACE-complete	decidable
$d \leq 1$	?	undecidable
$d \leq 2$	undecidable	undecidable
not bound	undecidable	undecidable

[16] [Aucher and Bolander, 2013]
[17] [Charrier et al., 2016]

Some open questions

- Circumvent undecidability. [18]
- Find compact representations for models. [19]
- Find representation languages for actions. [20]
- Model belief (instead of knowledge). [21]
- Propose heuristics for epistemic planning.

---

[18] [Bolander et al., 2020, Cooper et al., 2021]
[19] [Charrier and Schwarzentruber, 2017, van Benthem et al., 2018, Gamblin et al., 2022]
[20] [Baral et al., 2022]
[21] [Balbiani et al., 2012, Caridroit et al., 2016]

Section 10

References

References I

Aucher, G. and Bolander, T. (2013).
Undecidability in epistemic planning.
In *Proc. of IJCAI*.

Aucher, G. and Schwarzentruber, F. (2013).
On the complexity of dynamic epistemic logic.
In *In Proc. of TARK*, pages 19–28.

Balbiani, P., van Ditamarsch, H., Herzig, A., and de Lima, T. (2012).
Some truths are best left unsaid.
In *Proc. of AiML 9*, pages 36–54.

Balbiani, P., van Ditmarsch, H., Herzig, A., and de Lima, T. (2010).
Tableaux for public announcement logic.
Journal of Logic and Computation, 20(1):55–76.

Baltag, A. and Moss, L. S. (2004).
Logics for epistemic programs.
Synthese, 139:165–224.

Baltag, A., Moss, L. S., and Solecki, S. (1998).
The logic of public announcements, common knowledge, and private suspicions.
In *Proc. of TARK*, pages 43–56.

References II

Baral, C., Gelfond, G., Pontelli, E., and Son, T. C. (2022).
An action language for multi-agent domains.
Artificial Intelligence, 302:103601.

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., Charrier, T., Pinchinat, S., and Schwarzentruber, F. (2020).
Del-based epistemic planning: Decidability and complexity.
Artificial Intelligence, 287:1–34.

Caridroit, T., Konieczny, S., de Lima, T., and Marquis, P. (2016).
On distances between kd45n kripke models and their use for belief revision.
In *Proc. of ECAI*, pages 1053–1061.

Charrier, T., Maubert, B., and Schwarzentruber, F. (2016).
On the impact of modal depth in epistemic planning.
In *Proc. of IJCAI*.

Charrier, T. and Schwarzentruber, F. (2017).
A succinct language for dynamic epistemic logic.
In *Proc. of AAMAS*, pages 123–131.

References III

Cooper, M. C., Herzig, A., Maffre, F., Maris, F., Perrotin, E., and Régnier, P. (2021).
A lightweight epistemic logic and its application to planning.
Artificial Intelligence, 298:103437.

Engesser, T., Mattmüller, R., Nebel, B., and Thielsher, M. (2021).
Game description language and dynamic epistemic logic compared.
Artificial Intelligence, 292:103433.

Gamblin, S., Niveau, A., and Bouzid, M. (2022).
A symbolic representation for probabilistic dynamic epistemic logic.
In *Proc. of AAMAS*, pages 445–453.

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

Lutz, C. (2006).
Complexity and succintness of public announcement logic.
In *Proc. AAMAS*, pages 137–144.

Plaza, J. (1989).
Logics of public communications.
In *Proc. of ISMIS*, pages 201–216.

References IV

Plaza, J. (2007).
Logics of public communications.
Synthese, 158(2):165–179.

van Benthem, J., van Eijck, J., Gattinger, M., and Su, K. (2018).
Symbolic model checking for dynamic epistemic logic – s5 and beyond.

van Benthem, J., van Eijck, J., and Kooi, B. (2006).
Logics of communication and change.
Information and Computation, 204(11):1620–1662.

van der Hoek, W. and Wooldridge, M. (2002).
Tractable multiagent planning for epistemic goals.
In *Proc. of AAMAS*, pages 1167–1174. ACM Press.

van Ditmarsch, H., Herzig, A., and de Lima, T. (2012).
Public announcements, public assignments and the complexity of their logic.
Journal of Applied Non-Classical Logics, 22(3):249–273.

van Ditmarsch, H., van der Hoek, W., and Kooi, B. (2005).
Dynamic epistemic logic with assignment.
In *Proc. AAMAS*, pages 141–148.

References V

van Ditmarsch, H., van der Hoek, W., and Kooi, B. (2007).
Dynamic Epistemic Logic.
Springer.

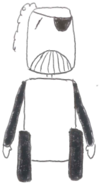
Table of contents

Part 4

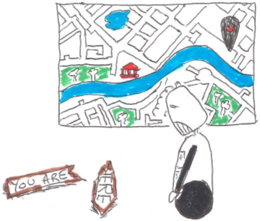
Contingent Planning with Belief States

Setting

Nondeterministic actions



Partial observability



Uncertain state

Contents

Section 11

One Agent, No Probabilities

Minesweeper Instance

?	?	?
?	1	?
?	2	?
?	?	?

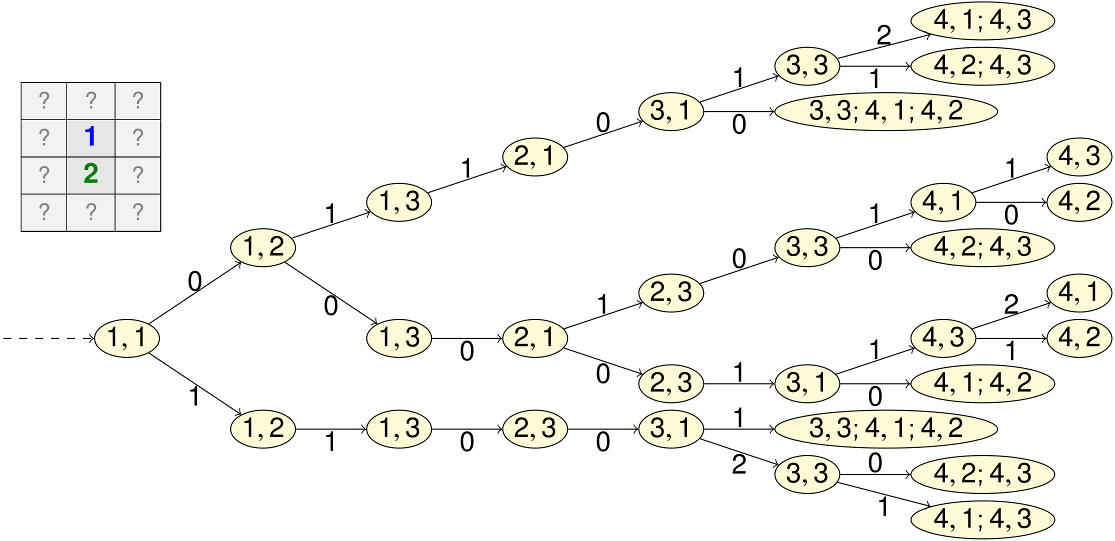
Instance:

- states = all possible grids with 2 mines
- actions = $\{\textsc{click}(i, j) \mid i, j\}$
- observations = $\{0, 1, 2, \ldots, 8\} \cup \{$$\}$
- initial belief state = all states consistent with numbers revealed

Note: adversarial/robust version

Example Winning Policy

?	?	?
?	1	?
?	2	?
?	?	?



Formal Setting

Contingent planning instance:

- sets S (states), A (actions), Ω (observations)
- transition function $T : S \times A \to \mathcal{P}(S)$
- goal states $G \subseteq S$
- observation function: $O : S \times A \times S \to \mathcal{P}(\Omega)$
- initial belief: $B_0 \subseteq \mathcal{P}(S)$

Strong cyclic policy:

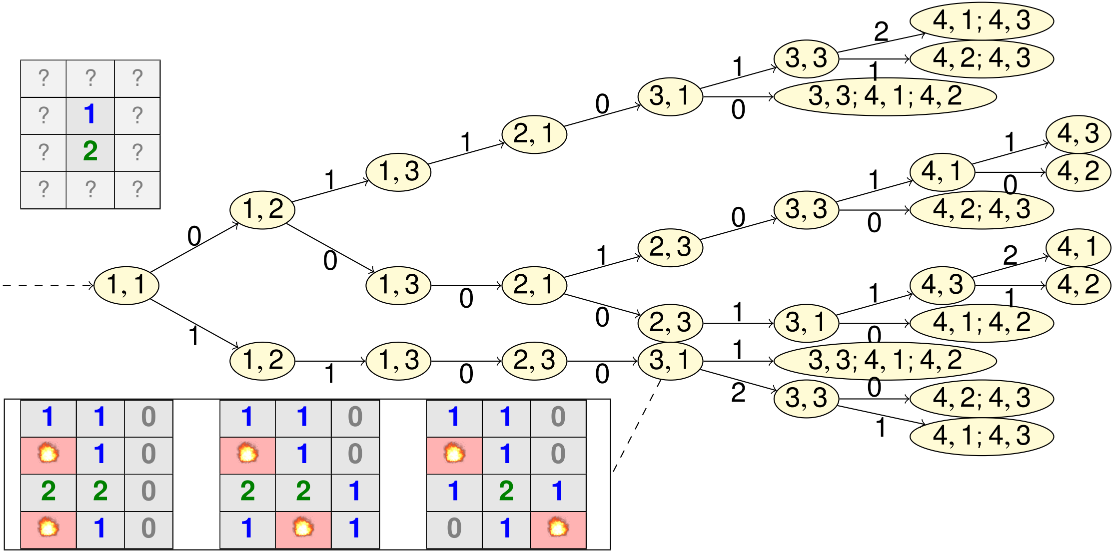
- mapping $\pi : \Omega^* \to A$
- value: 1 (winning) if $\forall \omega_1, \omega_2, \ldots,$ policy π reaches goal , else 0
- note: winning policy existence decidable (finite space)

And/Or Search

Finding strong policy for contingent planning = and/or search:

- root = B_0
- or-nodes = possible actions
- and-node = possible observations
- leaves = goal states
- policy = strategy in And/Or graph

Belief States



Progression in Belief Space

In histories:

1	1	0
?	1	0
?	2	?
?	?	?

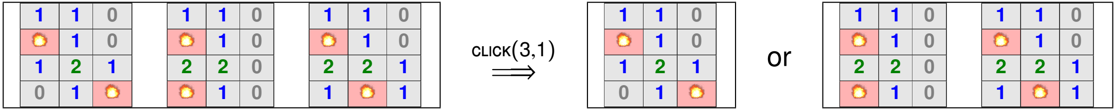
CLICK(3,1) $\Longrightarrow$

1	1	0
?	1	0
1	2	?
?	?	?

or

1	1	0
?	1	0
2	2	?
?	?	?

In belief space:



Belief Space Fully Observable Problem

Progression: $\mathrm{prog}(B, a, \omega) := \{s' \in S \mid \exists s \in B : s' \in T(s, a), \omega \in O(s, a, s')\}$

Belief Space Fully Observable Problem

Progression: $\text{prog}(B, a, \omega) := \{s' \in S \mid \exists s \in B : s' \in T(s, a), \omega \in O(s, a, s')\}$

Belief space transformation $\cdot^{\mathcal{B}}$ for contingent instance $I = (S, A, T, R, \Omega, O, B_0)$:

- $S^{\mathcal{B}} := \mathcal{P}(S)$
- $A^{\mathcal{B}} := A$
- $T^{\mathcal{B}}(B, a) := \{\text{prog}(B, a, \omega) \mid \exists s' \in T(s, a) : \omega \in O(s, a, s')\}$
- $R^{\mathcal{B}}(B) := \min_{s \in B} R(s)$
- belief state fully observed: $\Omega := S^{\mathcal{B}}, O(B, a, B') := \{B'\}$
- policy for $I^{\mathcal{B}} \equiv$ policy for I

Fully observable nondeterministic planning

Planning in the Belief Space

Direct approaches:

- CMBP [Cimatti and Roveri, 2000]: conformant planning (no sensing), regression-based
- AO*: contingent planning [Bonet and Geffner, 2000]
- belief states are huge $\rightarrow$ symbolic representations using BDDs
- other representations: DNF, CNF, Prime Implicates [To et al., 2017]

Known literals [Palacios and Geffner, 2009]:

- conformant planning
- store only $K\ell$ for relevant known literals in current B
- avoids storing B

Section 12

One Agent, Probabilities

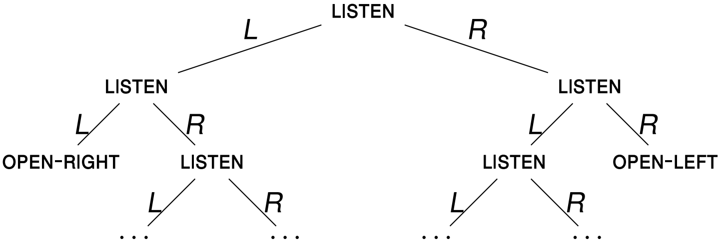
Tiger Example

Problem:

- two doors, one with tiger, one with gold
- ontic actions: open left/right door (+10 or -100)
- sensing action: listen roar , yields good/bad clue .9/.1
- initial belief: tiger left/right .5/.5
- timestep costs 1

Intuitively: listen enough to have strong belief where tiger is

Tiger Policy



POMDPs: Formal Setting

Partially Observable Markov Decision Problem:

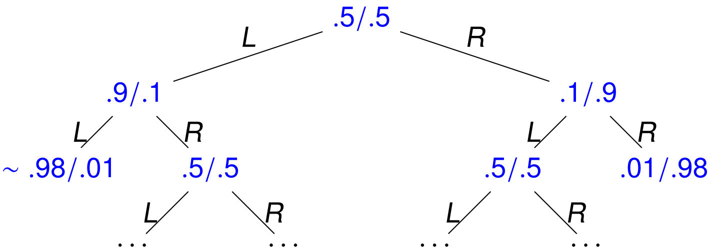
- sets S (states), A (actions), Ω (observations)
- transition function: $T : S \times A \to \Delta(S)$
- reward function: $R : S \to \mathbb{R}$
- observation function: $O : S \times A \times S \to \Delta(\Omega)$
- initial belief: $B_0 \in \Delta(S)$

Solution/policy:

- again depends on whole history: mapping $\pi : \omega \in \Omega^* \to A$
- value: expectation of cumulated reward
- note: undecidable at indefinite horizon

Belief Based are Again Here

Recall: B_0 is left/right .5/.5, listen gives clue .9/.1, reward +10/-100



Maintained by Bayes rule:

$$B(s') \leftarrow \eta\Big(\sum_{s} B(s)T(s' \mid s,a)O(\omega \mid s,a,s')\Big)$$

Regression Approach

Recall: B_0 is left/right .5/.5, listen gives clue .9/.1, reward +10/-100

One action remaining:

- open left gives $-100B(\mathrm{l}) + 10B(\mathrm{r}) - 1$
- open right gives $10B(\mathrm{l}) - 100B(\mathrm{r}) - 1$
- listen gives $0B(\mathrm{l}) + 0B(\mathrm{r}) - 1$

$$\Rightarrow \alpha\text{-vectors: } \left\{ \begin{array}{rcll} v^1(\textsc{open-left}) & = & (-100, 10, -1) & , \\ v^1(\textsc{open-right}) & = & (10, -100, -1) & , \\ v^1(\textsc{listen}) & = & (0, 0, -1) & \end{array} \right\}$$

Execution: maintain $B = (B(\mathrm{l}), B(\mathrm{r}), 1)$ and choose $\mathrm{argmax}_a \left(B \cdot v^1(a)\right)$

Regression Approach, 2 Actions Left

Open left, open right: still $v^2(\textsc{open-left}) = (-100, 10, -1)$, $v^2(\textsc{open-right}) = (10, -100, -1)$

Regression Approach, 2 Actions Left

Open left, open right: still $v^2(\textsc{open-left}) = (-100, 10, -1)$, $v^2(\textsc{open-right}) = (10, -100, -1)$

Listen; may yield observation L or R:

- open right on observation L and open left on R

$$B(\mathrm{l}) \times \left(.9v^1(\textsc{open-right}) + .1v^1(\textsc{open-left})\right) + B(\mathrm{r})\left(.1v^1(\textsc{open-right}) + .9v^1(\textsc{open-left})\right)$$

Regression Approach, 2 Actions Left

Open left, open right: still $v^2(\textsc{open-left}) = (-100, 10, -1)$, $v^2(\textsc{open-right}) = (10, -100, -1)$

Listen; may yield observation L or R:

- open right on observation L and open left on R

$$B(\mathrm{l}) \times \left(.9v^1(\textsc{open-right}) + .1v^1(\textsc{open-left})\right) + B(\mathrm{r})\left(.1v^1(\textsc{open-right}) + .9v^1(\textsc{open-left})\right)$$

$$= B(\mathrm{l}) \times (.9 \times (10, -100, -1) + .1 \times (-100, 10, -1)) + B(\mathrm{r}) \times (.1 \times (10, -100, -1) + .9 \times (-100, 10, -1))$$

Regression Approach, 2 Actions Left

Open left, open right: still $v^2(\textsc{open-left}) = (-100, 10, -1)$, $v^2(\textsc{open-right}) = (10, -100, -1)$

Listen; may yield observation L or R:

- open right on observation L and open left on R

$$B(\mathrm{l}) \times \left(.9v^1(\textsc{open-right}) + .1v^1(\textsc{open-left})\right) + B(\mathrm{r})\left(.1v^1(\textsc{open-right}) + .9v^1(\textsc{open-left})\right)$$

$$= B(\mathrm{l}) \times (.9 \times (10, -100, -1) + .1 \times (-100, 10, -1)) + B(\mathrm{r}) \times (.1 \times (10, -100, -1) + .9 \times (-100, 10, -1))$$

$$\Rightarrow v_1^2(\textsc{listen}) = aB(\mathrm{l}) + bB(\mathrm{r}) + c$$

Regression Approach, 2 Actions Left

Open left, open right: still $v^2(\textsc{open-left}) = (-100, 10, -1)$, $v^2(\textsc{open-right}) = (10, -100, -1)$

Listen; may yield observation L or R:

- open right on observation L and open left on R

$$B(\mathrm{l}) \times \left(.9v^1(\textsc{open-right}) + .1v^1(\textsc{open-left})\right) + B(\mathrm{r})\left(.1v^1(\textsc{open-right}) + .9v^1(\textsc{open-left})\right)$$

$$= B(\mathrm{l}) \times (.9 \times (10, -100, -1) + .1 \times (-100, 10, -1)) + B(\mathrm{r}) \times (.1 \times (10, -100, -1) + .9 \times (-100, 10, -1))$$

$$\Rightarrow v_1^2(\textsc{listen}) = aB(\mathrm{l}) + bB(\mathrm{r}) + c$$

- listen left on observation L and open right on $R \Rightarrow v_2^2(\textsc{listen}) = dB(\mathrm{l}) + eB(\mathrm{r}) + f$
- ...

Regression Approach, 2 Actions Left

Open left, open right: still $v^2(\textsc{open-left}) = (-100, 10, -1)$, $v^2(\textsc{open-right}) = (10, -100, -1)$

Listen; may yield observation L or R:

- open right on observation L and open left on R

  $$B(\mathrm{l}) \times \left(.9v^1(\textsc{open-right}) + .1v^1(\textsc{open-left})\right) + B(\mathrm{r})\left(.1v^1(\textsc{open-right}) + .9v^1(\textsc{open-left})\right)$$

  $$= B(\mathrm{l}) \times (.9 \times (10, -100, -1) + .1 \times (-100, 10, -1)) + B(\mathrm{r}) \times (.1 \times (10, -100, -1) + .9 \times (-100, 10, -1))$$

  $$\Rightarrow v_1^2(\textsc{listen}) = aB(\mathrm{l}) + bB(\mathrm{r}) + c$$

- listen left on observation L and open right on $R \Rightarrow v_2^2(\textsc{listen}) = dB(\mathrm{l}) + eB(\mathrm{r}) + f$
- ...

Execution: again, maintain $B = (B(\mathrm{l}), B(\mathrm{r}), 1)$ and choose $\operatorname{argmax}_a \left(\operatorname{argmax}_i \left(B \cdot v_i^2(a)\right)\right)$

Wrap-up: Regression

Planning time; compute α-vectors:

- set $v^0(\_) := \{R\}$
- for $t = 1, 2, \ldots$: set $v^t(a) := \{\omega_1 : v_1, \ldots, \omega_k : v_k \mid v_1, \ldots, v_k \in v^{t-1}\}$
- until ε-convergence/stopping criterion

Execution time, given α-vectors $\forall a, v(a)$:

- set $B := B_0$
- perform $a := \text{argmax}_a B \cdot v(a)$
- observe ω
- update B using a, ω and Bayes rule
- iterate

Section 13

Knowledge-Based Policies

Knowledge-Based Policies

Intuition:

- recall: α-vectors $v_{i,j}(\text{OPEN-LEFT}), v_{i,j}(\text{OPEN-RIGHT}), v_{i,j}(\text{LISTEN})$
- $(B(\mathrm{l}), B(\mathrm{r}), 1) \cdot v(\text{OPEN-LEFT}) > (B(\mathrm{l}), B(\mathrm{r}), 1) \cdot \text{OPEN-RIGHT}, (B(\mathrm{l}), B(\mathrm{r}), 1) \cdot \text{LISTEN}$
  $\rightarrow$ compact representation of set of belief states

Let's generalize to a Knowledge-Based Policy [Z. et al., 2020]:

---

while $\neg K(goal)$ **do**

  if $K\neg\text{mine}(1,1)$ **then** $\text{CLICK}(1,1)$ **else** ε **fi;**

  if $K\neg\text{mine}(1,2)$ **then** $\text{CLICK}(1,2)$ **else** ε **fi;**

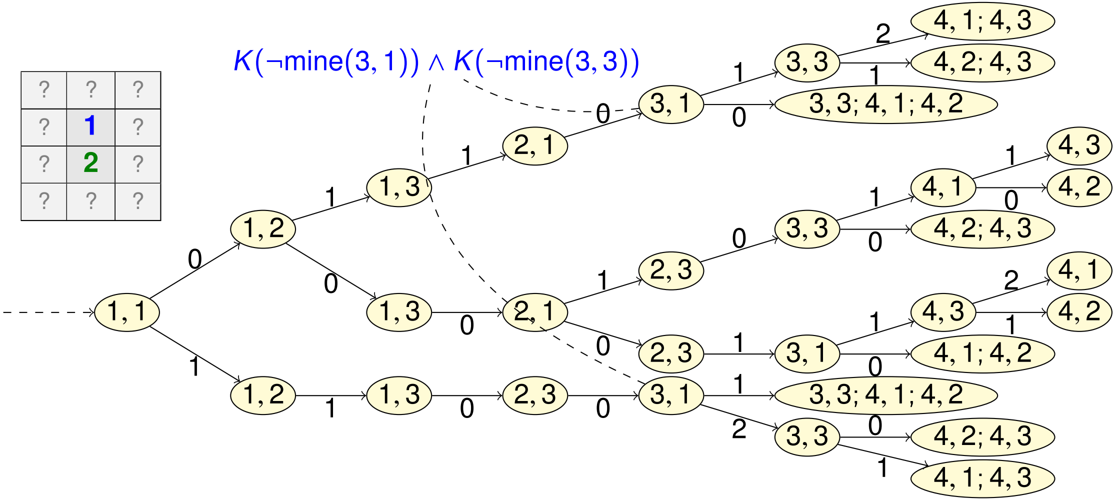
  ...

  if $K\neg\text{mine}(4,3)$ **then** $\text{CLICK}(4,3)$ **else** ε **fi**

od

KBPs: Succinctness

Intuition: several histories lead to same sufficient knowledge



Complexity Issues

Executing a KBP:

- maintain knowledge
- decide branching conditions
- this is (single-agent) epistemic logic!

Technical questions:

- Proved: KBP always as succinct as reactive policy; possibly exponentially more
- KBP explainable
- no free lunch: execution is Θ_2^P-complete
- computing plans mostly open

Other Approaches to Planning

Many other approaches for POMDPs/contingent:

- dedicated algorithms
- forward, backward, heuristic, complete. . .
- machine learning. . .

Section 14

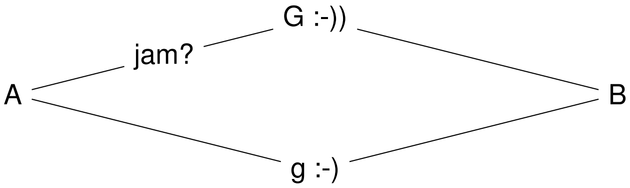
Several Agents, Probabilities

Decentralized Planning Tasks

Setting:

- multi-agent, collaborative
- offline planning, centralized
- online execution, decentralized, no explicit communication

Example:



radio but no cell phone

Decentralized POMDPs

Decentralized POMDP:

- sets of agents I, states S, actions A, observations Ω
- transition function $T : S \times A^I \to \Delta(S)$
- reward function $R : S \to \mathbb{R}$
- observation function $O : S \times A^I \times S \to \Delta(\Omega^I)$
- initial common belief state $B_0 \in \Delta(S)$

Decentralized POMDPs

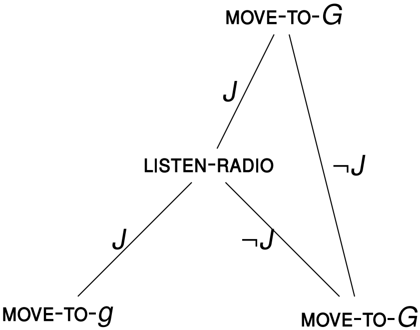
Decentralized POMDP:

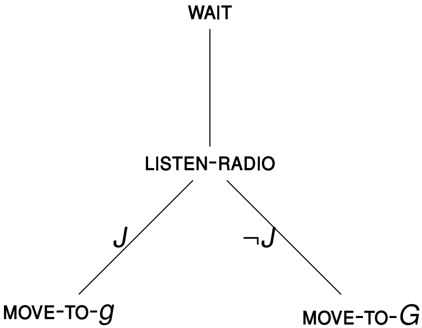
- sets of agents I, states S, actions A, observations Ω
- transition function $T : S \times A^I \to \Delta(S)$
- reward function $R : S \to \mathbb{R}$
- observation function $O : S \times A^I \times S \to \Delta(\Omega^I)$
- initial common belief state $B_0 \in \Delta(S)$

Joint policy:

- policy π for each agent
- policy of A = function from observation history of A
- value = expected reward of joint policy

Example Policy





Belief Space for Decentralized POMDPs

Natural generalization of single-agent case:

- maintain belief over state: $B \in \Delta(S)$
- not sufficient!
- should distinguish:
  - there is a traffic jam and B knows this
  - there is a traffic jam and B does not know

Belief Space for Decentralized POMDPs

Natural generalization of single-agent case:

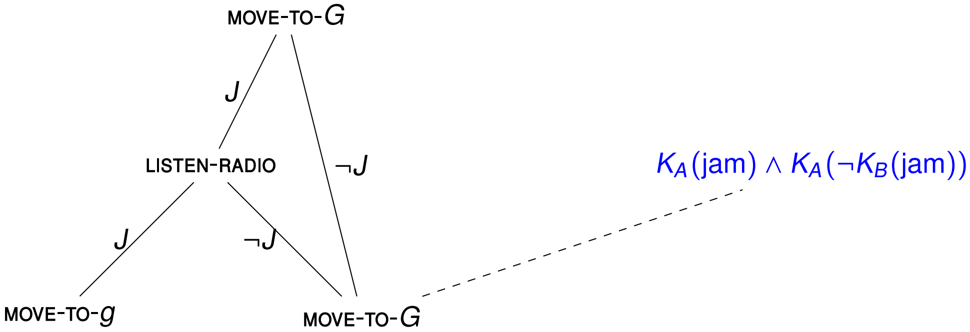
- maintain belief over state: $B \in \Delta(S)$
- not sufficient!
- should distinguish:
  - there is a traffic jam and B knows this
  - there is a traffic jam and B does not know

Each agent must maintain multi-agent knowledge!

- up to any depth
- this is reasoning in DEL

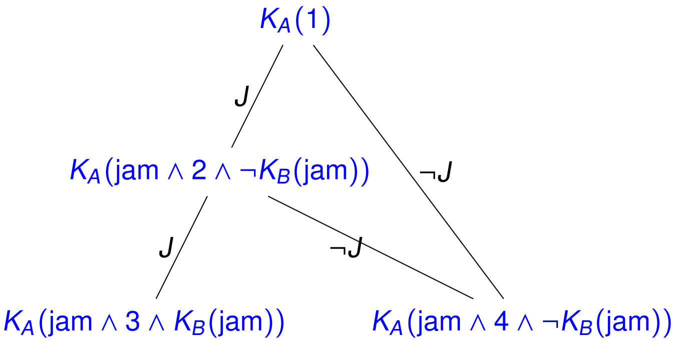
Maintaining Multi-Agent Knowledge in Practice

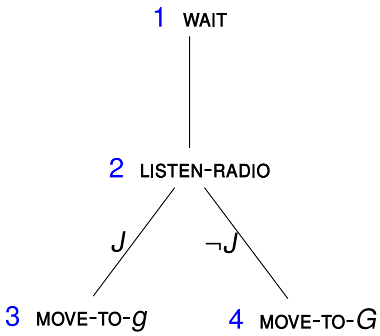
Implicit anyway:



Making Knowledge Explicit

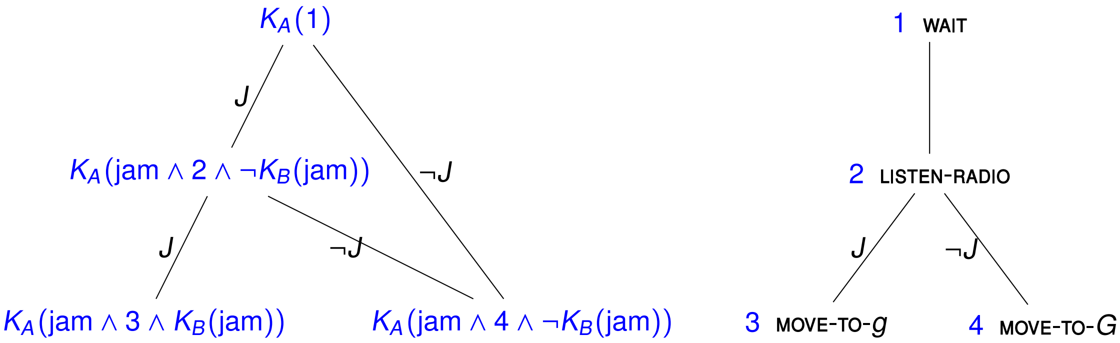
Maintain knowledge about state + other agents' "program counters"





Making Knowledge Explicit

Maintain knowledge about state + other agents' "program counters"



Notes:

- centralized planning is crucial
- knowledge about B's program counters may be imprecise, like $K_A(1 \vee 3)$

Multi-Agent KBPs

Multi-Agent KBP [Saffidine et al., 2018] for A:

---

while $\top$ **do**

  if $K_A(\neg\text{jam}) \vee \left(\neg K_A(\text{jam}) \wedge \neg K_A(\neg\text{jam})\right)$ **then** MOVE-TO-G

  else if $K_A(\text{jam}) \wedge \neg K_A(K_B(\text{jam})) \wedge \neg K_A(\neg K_B(\text{jam}))$ **then** LISTEN-RADIO

  else if $K_A(\text{jam}) \wedge K_A(K_B(\text{jam}))$ **then** MOVE-TO-g

  else if $K_A(\text{jam}) \wedge K_A(\neg K_B(\text{jam}))$ **then** MOVE-TO-G

od

---

and similar for B

Multi-Agent KBPs

Multi-Agent KBP [Saffidine et al., 2018] for A:

---

while $\top$ **do**

  if $K_A(\neg\text{jam}) \vee \big(\neg K_A(\text{jam}) \wedge \neg K_A(\neg\text{jam})\big)$ **then** MOVE-TO-G

  else if $K_A(\text{jam}) \wedge \neg K_A(K_B(\text{jam})) \wedge \neg K_A(\neg K_B(\text{jam}))$ **then** LISTEN-RADIO

  else if $K_A(\text{jam}) \wedge K_A(K_B(\text{jam}))$ **then** MOVE-TO-g

  else if $K_A(\text{jam}) \wedge K_A(\neg K_B(\text{jam}))$ **then** MOVE-TO-G

od

---

and similar for B

As succinct and possibly exponentially more than reactive policies

Section 15

References

References I

Bonet, B. and Geffner, H. (2000).
Planning with incomplete information as heuristic search in belief space.
In *Proc. AIPS 2000*.

Cimatti, A. and Roveri, M. (2000).
Conformant planning via symbolic model checking.
Journal of Artificial Intelligence Research, 13:305–338.

Palacios, H. and Geffner, H. (2009).
Compiling uncertainty away in conformant planning problems with bounded width.
Journal of Artificial Intelligence Research.

References II

Saffidine, A., Schwarzentruber, F., and Zanuttini, B. (2018).
Knowledge-based policies for qualitative decentralized pomdps.
In McIlraith, S. A. and Weinberger, K. Q., editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 6270–6277. AAAI Press.

Son Thanh To, Tran Cao Son, E. P. (2017).
A generic approach to planning in the presence of incomplete information: Theory and implementation (extended abstract).
In *Proc. IJCAI 2017*.

Zanuttini, B., Lang, J., Saffidine, A., and Schwarzentruber, F. (2020).
Knowledge-based programs as succinct policies for partially observable domains.
Artif. Intell., 288:103365.

Part 5

Temporal, dynamic and flexible planning

Table of contents

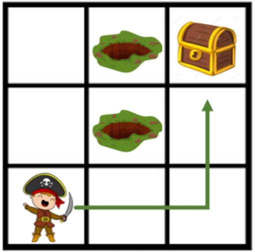
Section 16

Basics of Temporal Planning

Classical Planning

Sequence of actions from an initial state to a final state

- Initial State: pirate position
- Action: left, right, down, top
- Goal: reach the treasure



Sequence: right → right → top → top

Temporal planning

Added potential expressiveness:

- durations of the actions
- preconditions / effects should be true at the beginning, at the end, or during the actions
- temporal relationships between actions
- parallelism / concurrency
- synchronization / interruption

Temporal planning: a brief history

Some history

STRIPS (FIKES et al., 1970, Artificial Intelligence):

- First state-based search planner
- Implicit representation of time through succession of states
- Use relative time labels specifying after what an action can be executed

GraphPlan (Blum et al., 1995, IJCAI):

- Builds a state graph + transitions = all possible actions
- Allows parallelism and adds mutex

Temporal planning: a brief history

First-intention "temporal" classical planners:

- First produce a task plan and then assign timestamps to the actions starting points
- Implicit representation of time
- Greedily repairs the plan in case of flaws
- Solves only temporally simple problems

MetricFF (Hoffmann et al., 2003, AIR) unofficially wins the IPC-2008 time channel

YAHSP (V. Vidal et al., 2011 & 2014, IPC) wins IPC-2011 and 2014

Temporal planning: towards expicit time

Deviser (Vere et al., 1983, IEEE):

- First planner to make time information explicit
- Parallel planner with time and duration constraints
- Deterministic durations
- Ad-hoc representation = not based on any known theoretical model

Temporal planning: towards expicit time

O-Plan (Currie et al., 1991, AI):

- First planner to use time point concepts and metric constraints between time points
- Extends the literal formulation of DEVISER

Temporal Constraint Networks (Dechter et al., 1991, AI):

- First theoretical model of time constraints (TCSP)
- Based on time graph representation (STN, DTN)
- First filtering algorithms and time verification (AC, PC, ...)

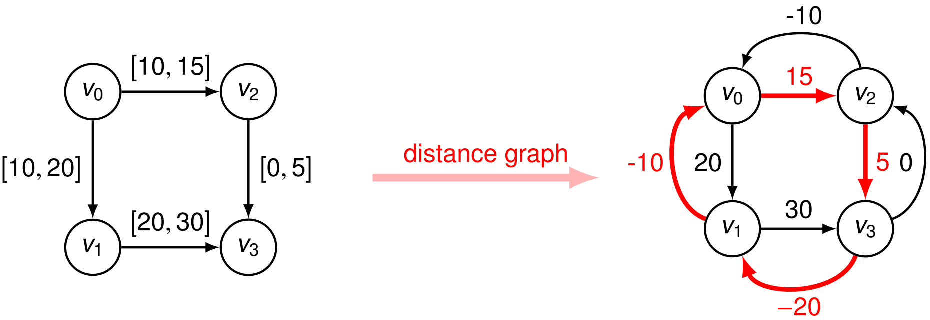
Section 17

Dealing with Time and Uncertainty in Planning

A CSP-based Dedicated Time Management

Simple Temporal Network (STN, Dechter et al., 1991, AI)

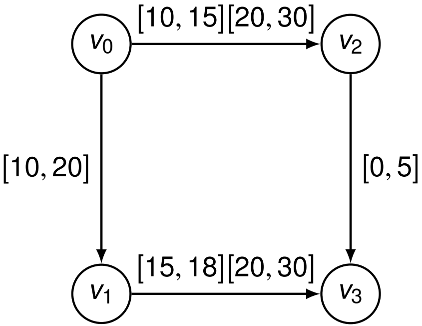
- Nodes = time-points and edges = durations (intervals)
- is **consistent** if there is an assignment of values to instants satisfying all time constraints.
- consistency is checked through polynomial-time propagation algorithms (O(n^3)): Path consistency or Floyd-Warshall



A CSP-based Dedicated Time Management

Disjunctive Temporal Network (DTN, Studer et al., 1998, DKE)

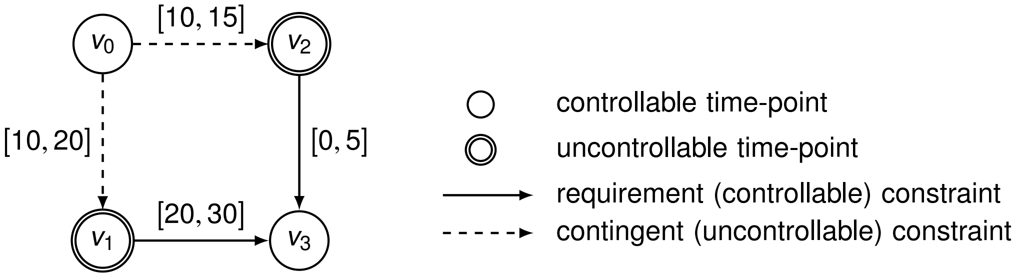
- Nodes = time-points and edges = sets of duration intervals
- checking consistency is NP-hard



How to manage uncertain durations in Temporal Networks

Simple Temporal Network with Uncertainty (STNU, Vidal et al., 1999)

- Nodes = time-points and edges = **controllable** and **uncontrollable** duration (interval)



Consistency redefined as Controllability

An STNU is **controllable** if an assignment of the controllable time-points exists such that all the requirement constraints are satisfied, whatever values taken by the contingent durations.

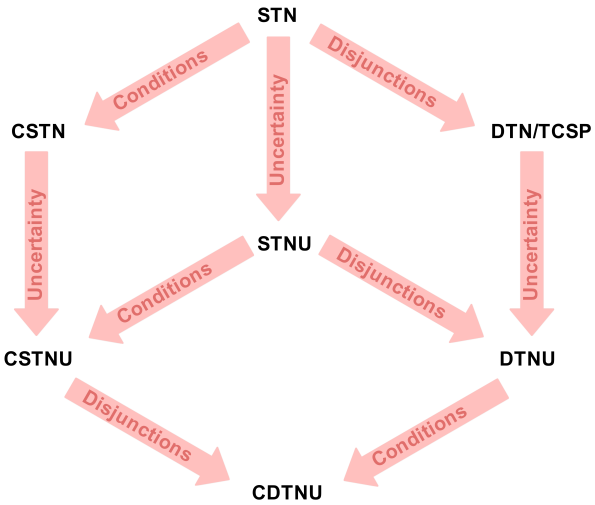
Three situations depending on when and how effective durations are observed:

- **Weak controllability (WC)** assumes contingents are observed just before execution.
- **Dynamic controllability (DC)** assumes contingents are observed during execution
- **Strong controllability (SC)** assumes contingents are never known/observed.

Complexity:

- WC is co-NP-complete
- DC is polynomial
- SC is polynomial

Going further: adding conditional branches



Adding explicit time: more history

IxTeT (Laborie et al., 1995, IEEE):

- First temporal planner incorporating plan generation and a temporal constraint (and resource) solver
- Use STNs for consistency

IxTeT-eXec (Lemai et al., 2004, ICAPS):

- Regularly updates the plan during execution
- Reactive plan repair in the event of failure
- Incremental replanning when new targets are set
- Consider DTNs and STNUs with dynamic controllability

Adding explicit time: more history

State search approach + temporal reasoning

PDDL2.1 (Fox and Long, 2003, JAIR):

- Extension of PDDL (Planning Domain Description Language) to PDDL2.1 to include temporal aspects

CRIKEY (Hashley et al., 2004, ECAI):

- Able to reason with coordinated actions
- Divides sustainable actions into start and end actions
- Uses STNs
- **CRIKEY3 (Coles et al., 2008, AAAI)**: temporal coordination problems such as deadlines

TLP-GP (Maris et al., 2008, Time) & LPGP (Long et al., 2003, ICAPS):

- GraphPlan-based with SAT or DTN solver

Adding explicit time: more history

Other approaches

Prottle (Little et al., 2005, AAAI):

- Extends PDDL2.1 to consider probabilistic effects
- Uses AND/OR graphs for state search

Tempastic (Younes et al., 2004, ICAPS):

- Limited to deterministic problems because STNs are used
- Policy generation, debugging and repair for continuous planning with concurrency

Adding explicit time: more history

Beaudry et al., 2010, ICAPS:

- Bayesian approach extending the forward approach
- Represents uncertainty continuously and randomly (numerical value)
- Manages concurrency under time uncertainty

ITSAT (Rankooh et al., 2015, JAIR):

- A satisfiability-based planner using a SAT solver

FAPE (Bit-Monnot et al., 2019, CoRR):

- Considers hierarchical and time-based planning

Bernardini et al., 2017, Autonomous Robots:

- Temporal planning + probabilistic reasoning for autonomous vehicles on surveillance missions.

Section 18

Dynamic planning and execution

Planning, Scheduling, Resource Allocation

Task Planning

- choose and order the actions that will enable the agent to achieve a given goal

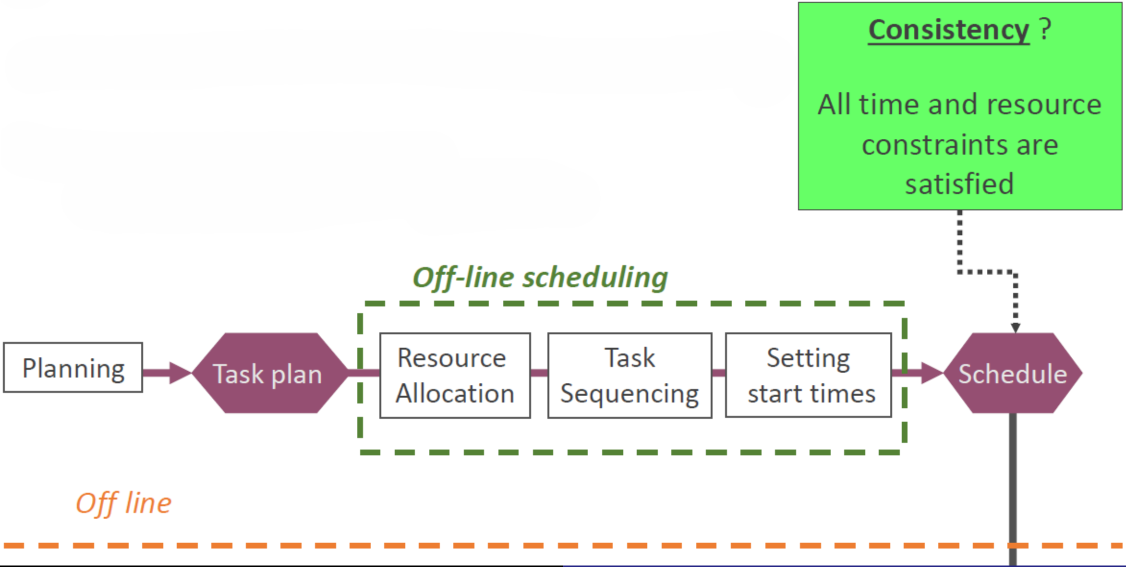
Scheduling

- place in time a set of known operations to be performed by the agent

Ressource allocation

- assign a resource to each operation required for its execution (e.g., machine, operator, tool, etc.)

General framework of planning/scheduling without uncertainty



Off-line/Online reasoning

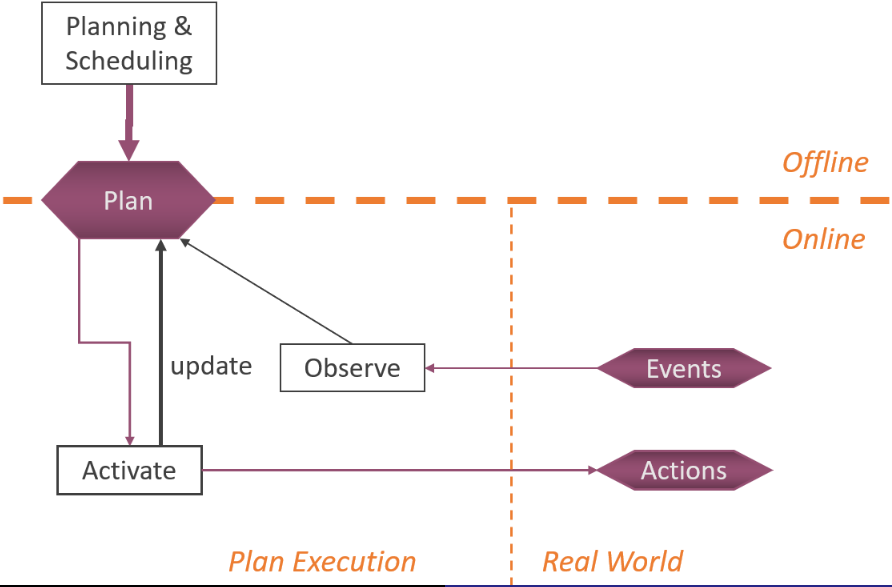
Off-line reasoning: predictive planning/scheduling

- Generally static
- Never questioned by the execution manager

Online reasoning: simultaneous with execution

- Dynamic by nature
- Reactive to observations
- Meets real-time needs

Plan execution in the ideal world



Execution under uncertainty ?

The planned schedule is not always adapted to the current situation

- Adapt online through replanning/rescheduling?
- Making the predictive plan/schedule more robust?
- Compromise between those two options?

Flexibility, Stability, Robustness

Flexible plan/schedule = alternatives are left open, with online arbitration

- Time flexibility
- Order flexibility
- Flexibility on assignments
- Flexibility on actions/action sequences

Stable plan/schedule = minimize the discrepancy between the predicted plan and the actually executed one

Robust plan/schedule = minimize at execution time the loss of “quality” from the optimal predicted plan

Possible sources of disruption/uncertainties

Goals

- new needs (e.g., redo a failed task, new order, etc.)

Events:

- unforeseen (e.g., machine breakdown) or with unknown date of occurrence
- observability: partially / not observable

Actions:

- variable/uncertain durations
- undesirable effects / disregarded preconditions: *to move, the battery must not be empty!*

Possible sources of disruption/uncertainties

Uncertainties may be on:

- time / resources / state of the world

Uncertain events may be:

- synchronous (end of a task of uncertain duration, events expected at an uncertain date)
- asynchronous (might occur at any time)

Plan/schedule generation can be:

- monotonous: additions, but no change in the current plan
- non-monotonic: (emergency or opportunistic) revisions of the current plan

Models of uncertainty

Simple and basic:

- sets of possible values

Probabilities:

- Bayesian networks
- Markov Decision Processes

Possibilities:

- fuzzy sets

Planning and execution: reactive, proactive or progressive

Different studies exist to differentiate the different ways to interleave planning and execution: predictive or proactive vs reactive and sometimes continuous or progressive (Van de Vonder, E.Demeulemeester and W.Herroelen, 2007) (M.Davari and E.Demeulemeester, 2019) (Bidot et al., 2009). We have chosen to focus on the last one = summary of tutorials given at AAAI'02 and ICAPS'03.

Reactive approach:

- Plan predicted offline, but revised online = asynchronous events - non-monotonic

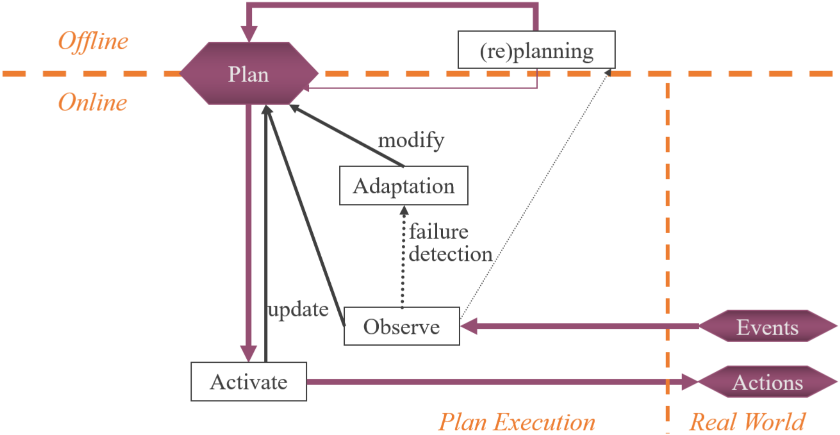
Progressive approach: Prediction/Execution on a sliding horizon:

- Short-term online planning, resuming as the exec removes uncertainties = monotonous

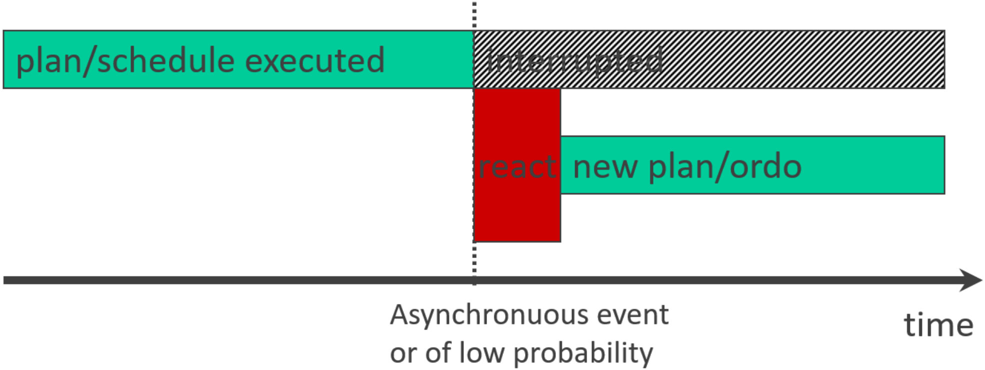
Proactive approach:

- Plan built offline, incorporating knowledge of uncertainties = synchronous events

Planning and execution: reactive approach

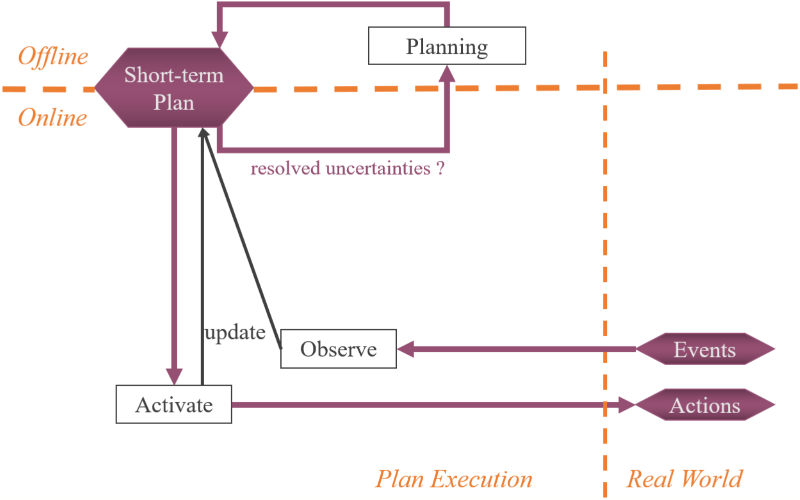


Planning and execution: reactive approach

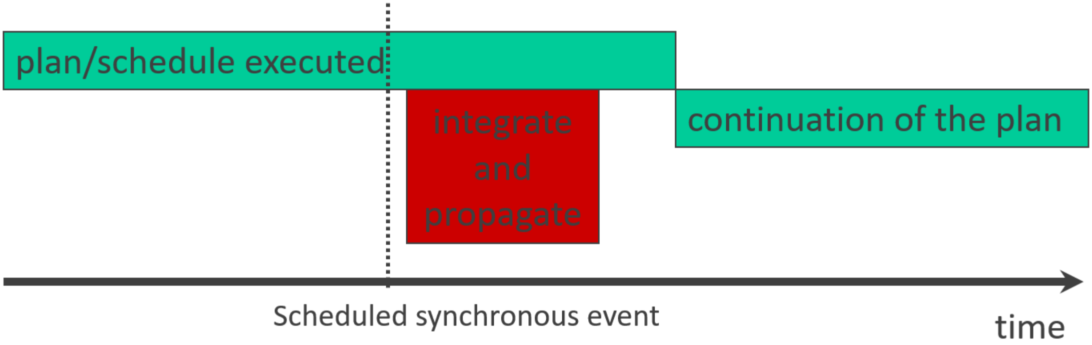


- Need to make decisions very quickly = generally suboptimal solutions
- Low memory requirements

Planning and execution: progressive approach



Planning and execution: progressive approach



- More time to decide = can be optimal
- Must not be too frequent
- Low memory requirements

Proactive approaches : 3 subfamilies

Complete methods:

- computation of a predictive rigid plan/ordo covering the largest number of cases
- stability goal + proba or fuzzy modeling

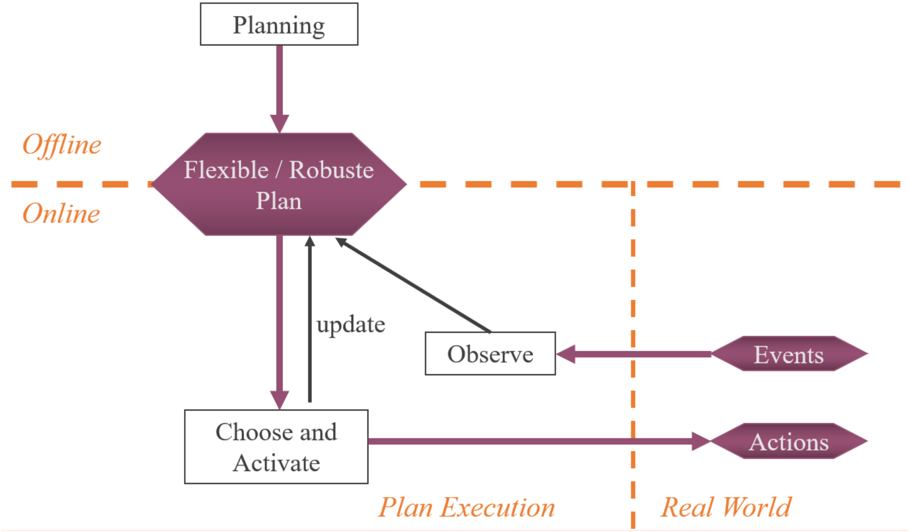
Flexible methods:

- added flexibility on times, orders, and/or assignments
- plans/schedules containing indetermination

Conditional methods:

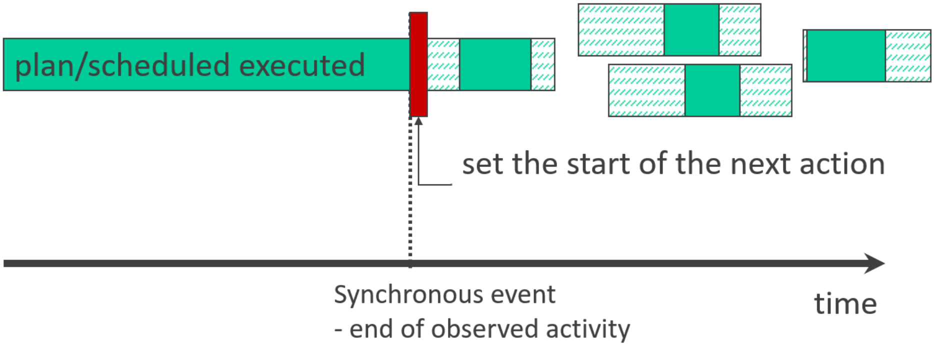
- added flexibility on possible actions/action sequences
- plans/schedules containing conditional branches

Proactive approaches : 3 subfamilies



Proactive approaches : 3 subfamilies

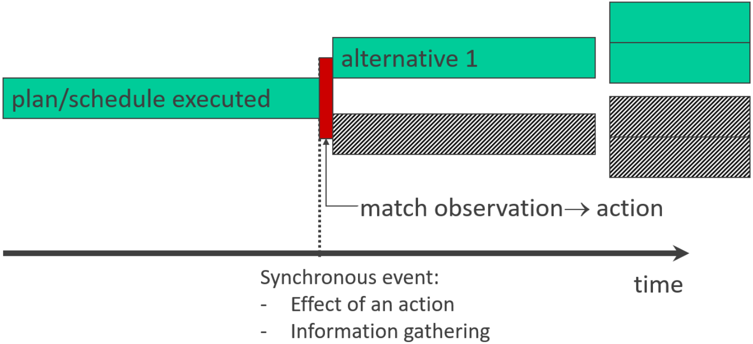
Proactive: time flexibility



- Quick decisions + at pre-determined times
- Low memory requirements

Proactive approaches : 3 subfamilies

Proactive: conditional branches



- Quick decisions + at pre-determined times
- Optimal
- High memory requirements

Section 19

References

References I

- Richard Fikes and Nils J. Nilsson, 1971
  STRIPS: A new approach to the application of Theorem Proving Problem Solving
  Artificial Intelligence
- Avrim Blum and Merrick L. Furst, 1995
  Fast Planning Through Planning Graph Analysis
  Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)
- Jörg Hoffmann, 2003
  The Metric-FF Planning System: Translating“Ignoring Delete Lists”to Numeric State Variables
  Journal of Artificial Intelligence Research
- Vincent Vidal, 2011
  YAHSP2: Keep it simple, stupid
  Proceedings of the 7th International Planning Competition (IPC-2011)
- Vincent Vidal, 2014
  YAHSP3 and YAHSP3-MT in the 8th international planning competition
  Proceedings of the 7th International Planning Competition (IPC-2011)
- Steven A. Vere, 1983
  Planning in Time: Windows and Durations for Activities and Goals
  IEEE Trans. Pattern Anal. Mach. Intell.

References II

Ken Currie and Austin Tate, 1991
O-Plan: The open Planning Architecture
Artificial Intelligence

Rina Dechter and Itay Meiri and Judea Pearl, 1991
Temporal Constraint Networks
Artificial Intelligence

Rudi Studer and V. Richard Benjamins and Dieter Fensel, 1998
Knowledge Engineering: Principles and Methods
Data Knowledge Engineering

Thierry Vidal and and Hélène Fargier, 1999
Handling contingency in temporal constraint networks: from consistency
Journal of Experimental & Theoretical Artificial Intelligence

Laborie Philippe and Ghallab Malik, 1995
IxTeT: an integrated approach for plan generation and scheduling
Symposium on Emerging Technologies and Factory Automation. (ETFA'95)

Solange Lemai and Félix Ingrand, 2004
Interleaving temporal planning and execution: IxTeT-eXeC
Proceedings of the ICAPS Workshop on Plan Execution

References III

Maria Fox and Derek Long, 2003
PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains
Journal of Artificial Intelligence Research

Keith Halsey and Derek Long and Maria Fox, 2004
Multiple Relaxations in Temporal Planning
Proceedings of the 16th Eureopean Conference on Artificial Intelligence

Andrew Coles and Maria Fox and Derek Long and Amanda Smith, 2008
Planning with Problems Requiring Temporal Coordination
Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence

Frederic Maris and Pierre Régnier, 2008
TLP-GP: Solving Temporally-Expressive Planning Problems
15th International Symposium on Temporal Representation and Reasoning

Derek Long and Maria Fox, 2003
Exploiting a Graphplan Framework in Temporal Planning
Proceedings of the Thirteenth International Conference on Automated

Iain Little and Douglas Aberdeen and Sylvie Thiébaux, 2005
Prottle: A Probabilistic Temporal Planner
Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence

References IV

Håkan L. S. Younes and Reid G. Simmons, 2004
Policy Generation for Continuous-time Stochastic Domains with Concurrency
Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS)

Eric Beaudry and Froduald Kabanza and François Michaud, 2010
Planning for Concurrent Action Executions Under Action Duration Uncertainty
Proceedings of the 20th International Conference on Automated Planning and Scheduling, ICAPS

Masood Feyzbakhsh Rankooh and Gholamreza Ghassem-Sani, 2015
ITSAT: An Efficient SAT-Based Temporal Planner
Journal of Artificial Intelligence Research

Arthur Bit-Monnot and Malik Ghallab and Félix Ingrand and David E. Smith, 2019
FAPE: a Constraint-based Planner for Generative and Hierarchical Temporal Planning
CoRR

Sara Bernardini and Maria Fox and Derek LongSara Bernardini and Maria Fox and Derek Long, 2017
Combining temporal planning with probabilistic reasoning for autonomous surveillance missions
Autonomous Robots

S.Van de Vonder and E.Demeulemeester and W.Herroelen, 2007
A classification of predictive-reactive project scheduling procedures
Journal of Scheduling, 10(3)

References V

M.Davari and E.Demeulemeester, 2019
The proactive and reactive resource-constrained project scheduling problem
Journal of Scheduling, 22(2)

Part 6

Multi-agent planning

Table of contents

Section 20

Basics of multi-agent planning

A MAP system

The agents:

- *Physical* distinctive entities acting on the world
- *Homogeneous* or *heterogeneous* (sensors/actuators, actions, knowledge model, reasoning capabilities)
- May have different levels of authority

Overall supervision system:

- Centralized or decentralized/distributed
- Mixed: e.g., centralized planning but distributed execution monitoring

Communication:

- Global or partial (neighbouring reachability)
- Instantaneous or with delays
- Reliable or delivery failures

Collaboration, cooperation, coordination?

Different ways of taking part in distributed problem solving (Sioutis et al., 2006)(Roschelle et al., 1995, CSCL)

Collaboration:

- a mutual engagement of participants to solve the problem together = *interactions during a necessarily distributed planning process*

Cooperation:

- a common task divided among participants, where each agent is responsible for a portion of the problem = *goals are distributed ("task" allocation) then local planning*

Coordination:

- a mutual commitment to synchronize the tasks at some points = *a global common plan has been generated, or on the contrary agents have their own private plans*

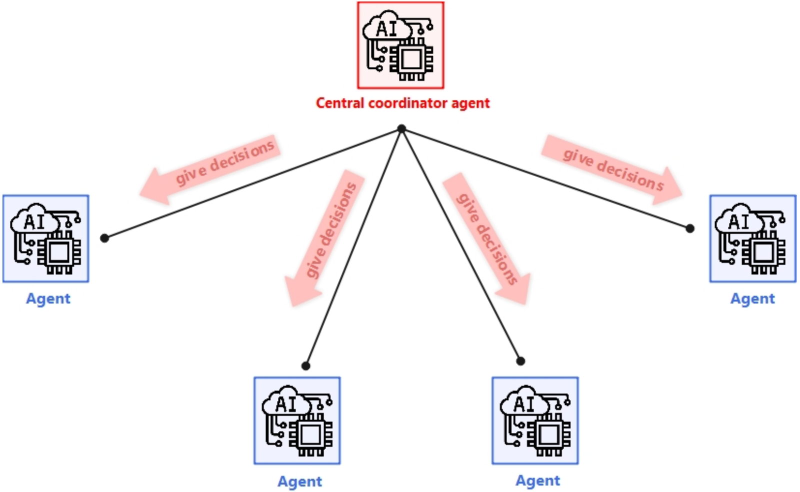
Centralized planning

- A common goal to satisfy
- A global plan
- Existence of a specific single agent with planning capability (others are executing agents)
- Classical planning systems can be used

Drawbacks:

- Not scalable: exponential blow-up in the action space (Jonsson et al., 2011, AAAI)
- No privacy among the agents (Nissim et al., 2012, AAMAS)

Centralized planning



Decentralized planning

It might refer to different paradigms:

- *Cooperative* agents with common goals ("tasks"), which are distributed among them
  - by a central agent
  - through negotiation

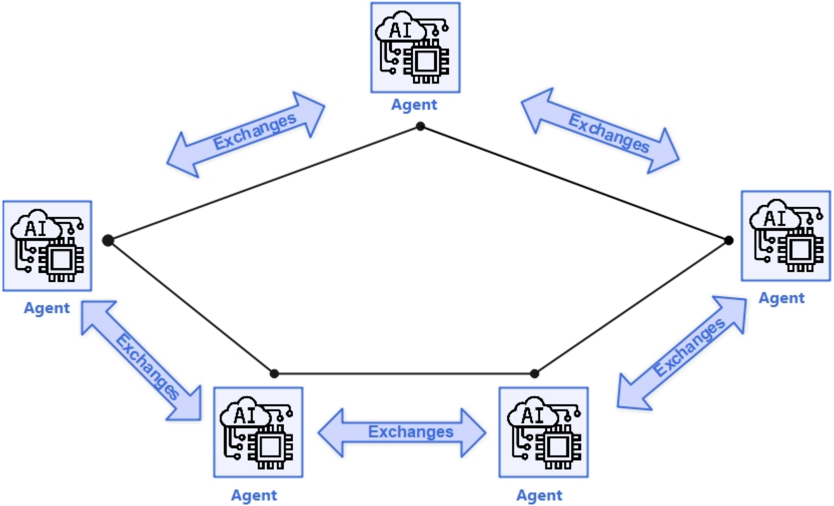
  + coordination at some points

- *Collaborative* agents where each
  - has its own goal(s) and builds its own plan but negotiate with others to improve their plan and/or help improve other agents' plans
  - takes part in the achievement of the common goal(s) by iteratively proposing (possibly mutual) actions

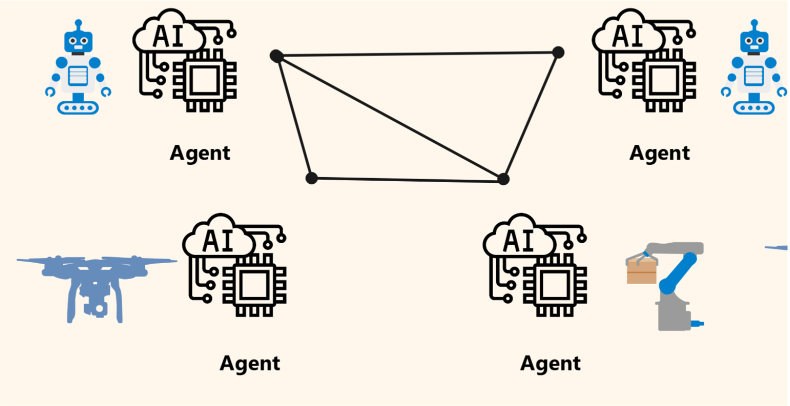
  + coordination at some points

- *Non-Collaborative* agents that selfishly aim to achieve their goals at others' expense

Homogeneous decentralized architectures



Heterogeneous decentralized architectures



Section 21

Task allocation: a quick survey

The task allocation problem

Aim

- Finding the best assignment of tasks among agents

Motivation

- (homogeneous) efficiency: closest agent / parallelism / needed cooperation
- (heterogeneous) tasks fit agent capabilities

A quick survey

5 main methods (Skaltis et al., 2021, ICUAS)

Auction-based methods:

- use negotiation protocol to bid on tasks based on local perception
- centralized (Contract Net Protocol) or distributed (Consensus-Based Bundle Algorithm)

Game-theoretical methods:

- agents are players and have some strategy
- aim to reach a global solution that is the best outcome for all the agents (Nash Equilibria)

Optimized-based methods:

- aims to maximize the profit or minimize the cost of a global function
- use deterministic, stochastic, or metaheuristic methods

A quick survey (continued)

5 main methods (Skaltis et al., 2021, ICUAS)

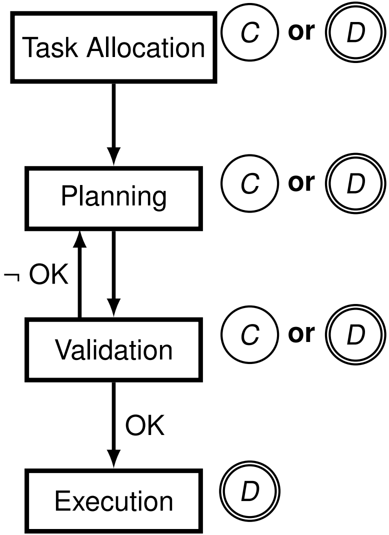
Learning based methods:

- provides learning capability to agents and trains them
- trains agents to confront potential disturbances depending on past decisions
- enables agents to react to future disturbances

Hybrid based methods:

- combines some of the previous methods
- provides more robust and complete solutions

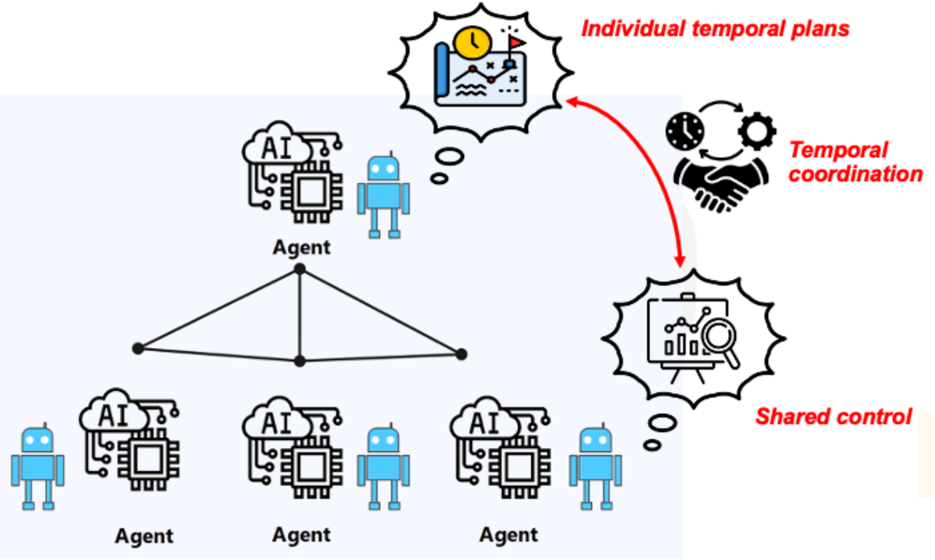
Getting the whole picture



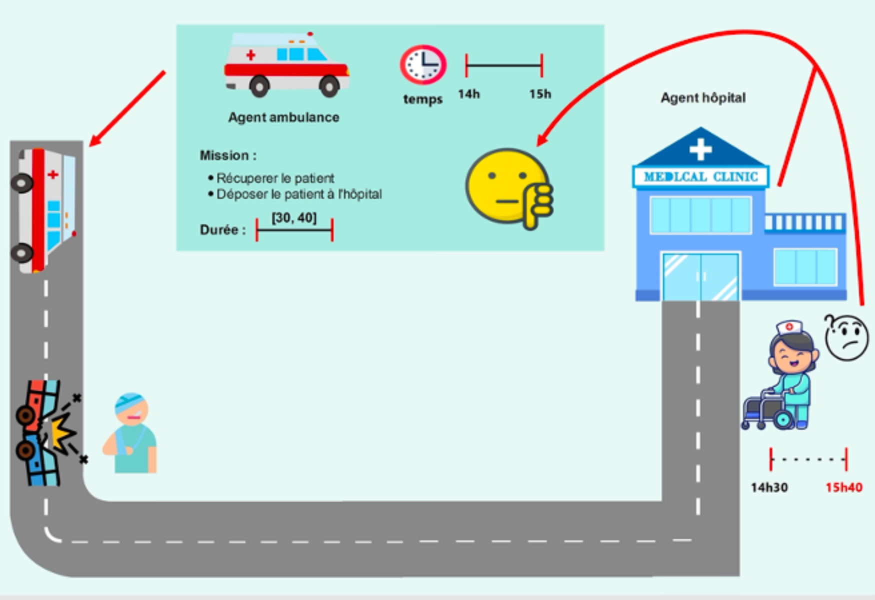
Section 22

Shared Control of Interdependent Plans

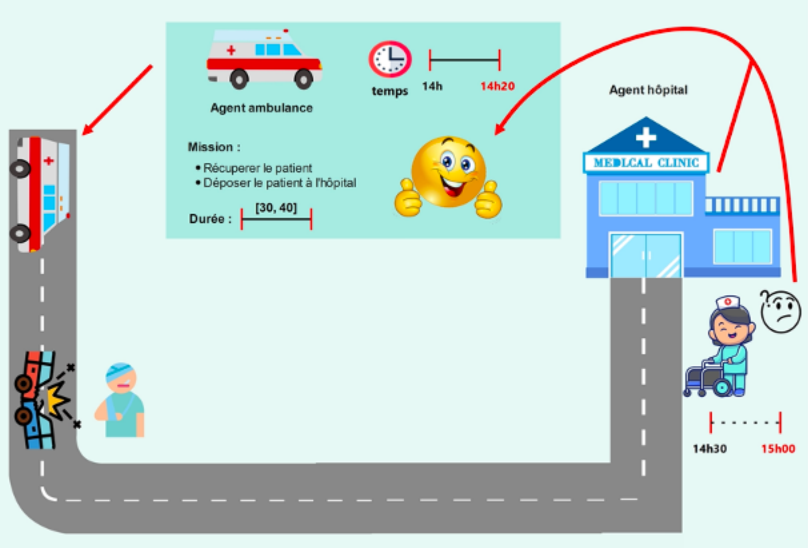
Coordination of Temporal Plans



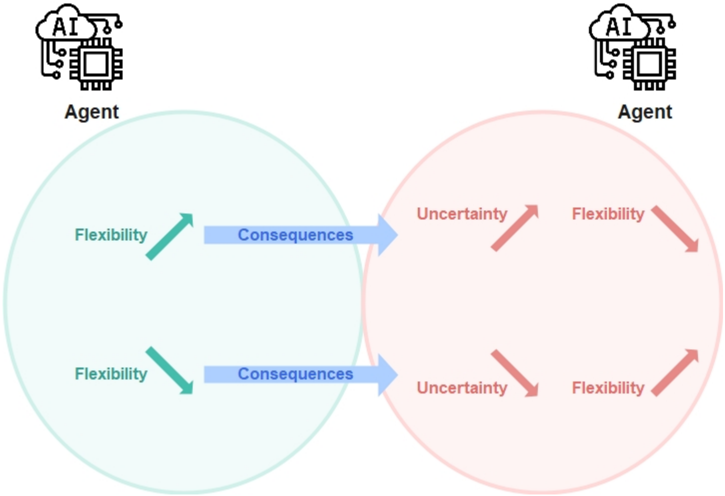
Illustrative Example



Illustrative Example



Flexibility sharing

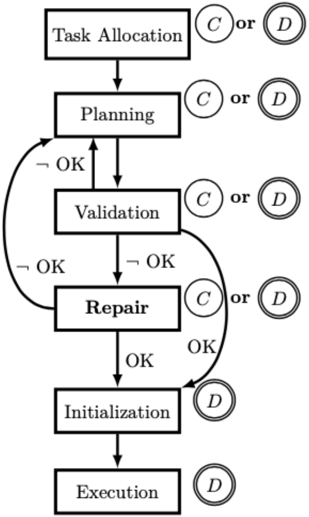


Recent / on-going work on this topic

Multiple Interdependent STNUs (A.Sumic and T.Vidal, 2024)

- Some activity durations (*contracts*) are controlled by some agent but observed by other agents that depend on them.
- Global controllability of a STNU = local controllabilities
- In case of local non-controllability due to such external contracts, better to *repair* through negotiation than to replan.

Revisiting / extending the whole picture



References I

Jeremy Roschelle and Stephanie D. Teasle, 1995
The construction of shared knowledge in collaborative problem solving
Computer supported collaborative learning

Christos Sioutis and Jeffrey Tweedale, 2006
Agent Cooperation and Collaboration
Knowledge-Based Intelligent Information and Engineering Systems (KES 2006)

Anders Jonsson and Michael Rovatsos, 2011
Scaling Up Multiagent Planning: A Best-Response Approach
International Conference on Automated Planning and Scheduling (ICAPS)

Raz Nissim and Ronen I. Brafman, 2012
Multi-agent A* for parallel and distributed systems
International Conference on Autonomous Agents and Multiagent Systems, AAMAS

George Marios Skaltsis, Hyo-Sang Shin, and Antonios Tsourdos, 2021
A survey of task allocation techniques in MAS
International Conference on Unmanned Aircraft Systems (ICUAS)

A.Sumic, T.Vidal, A.Micheli and A.Cimatti, 2024,
Introducing Interdependent Simple Temporal Networks with Uncertainty for Multi-agent Temporal Planning
31st Int'l Symposium on Temporal Representation and Reasoning (TIME'24)