# Reasoning about Knowledge for Planning

Bruno Zanuttini

Journée « Défis de l'IA », PFIA 2021, Tuesday, June 29, 2021

Partially Observable Planning Problems:

- states $s$ including variables $\mathsf{a}, \mathsf{b}$ (notation: $s \models a - b$)
- observations $\mathsf{fr}, \mathsf{ch}$
- obs. $\mathsf{fr}$ means $\mathsf{a} \to \mathsf{a} + 1$, $\mathsf{ch}$ means $\mathsf{b} \to \mathsf{b} + 1$

Sequences of observations:

- starting from $s \models 0 - 0$
- observing $(\mathsf{ch}, \mathsf{fr}, \mathsf{fr}, \mathsf{fr})$ means $s \models 3 - 1$
- but also $(\mathsf{fr}, \mathsf{ch}, \mathsf{fr}, \mathsf{fr})$, $(\mathsf{fr}, \mathsf{fr}, \mathsf{ch}, \mathsf{fr})$, $(\mathsf{fr}, \mathsf{fr}, \mathsf{fr}, \mathsf{ch})$

# In Two Slides

## Single-agent policies:

- assume single agent, $A = \{\text{didier}\}$
- assume action att is optimal when $s \models 1 - 0$
- assume action def is optimal when $s \models 3 - 1$
- standard policy representation (from $s \models 0 - 0$):
  - $\pi(\text{ch}, \text{fr}, \text{fr}, \text{fr}) = \text{def}$, $\pi(\text{fr}, \text{ch}, \text{fr}, \text{fr}) = \text{def}$, $\ldots$, $\pi(\text{ch}) = \text{att}$
- knowledge-based representation: $\mathsf{K}(3 - 1) \mapsto \text{def}$, $\mathsf{K}(0 - 1) \mapsto \text{att}$

## Multiagent setting:

- collaborative agents, (*e.g.*, $A = \{\text{karim}, \text{kylian}, \text{n'golo}, \ldots, \text{hugo}\}$
- reason on own knowledge
- but also on others': $\mathsf{K}(\neg \mathsf{K}_{\text{benjamin}}(\text{hugo-alone})) \mapsto \text{def}$

# Outline

Some Planning Settings

Reasoning with Knowledge: The Single-Agent Case

Reasoning with Knowledge in QDec-POMDPs

More Challenges

# Outline

Some Planning Settings

Reasoning with Knowledge: The Single-Agent Case

Reasoning with Knowledge in QDec-POMDPs

More Challenges

# Single-Agent Contingent Planning

Planning actions for reaching a goal contingent on observations:

- state space, actions, goal states
- known successor function, nondeterministic
- known observation function

Example:

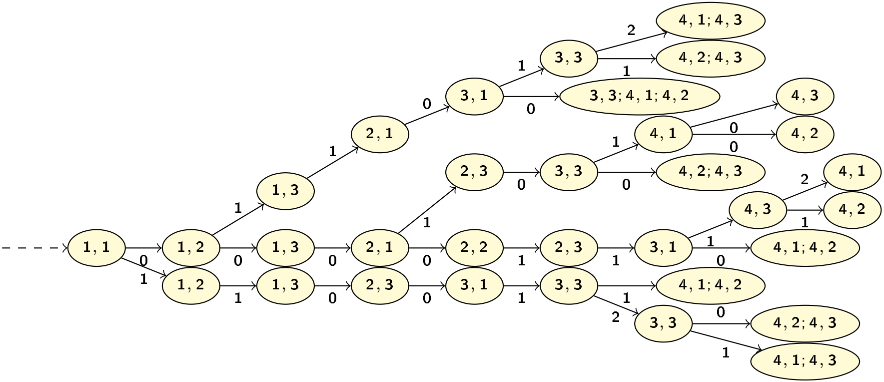
| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

goal state = any with all non-bombs clicked

Stochastic setting:
Partially Observable Markov Decision Processes (POMDPs)

# Contingent Policies

Mapping from histories of observations to actions




Execution: going from node to node using observations received

# Multiagent Planning

This presentation: common rewards/collaborative planning

Centralized setting:

- at each timestep, CA dictates each agent the action to take
- agent tell their observations to CA
- examples: sensor networks, patrolling

Distributed setting:

- no plan/policy agreed upon
- each agent plans, executes, observes on its own
- communication modelled by actions, if any
- example: crisis management (firemen, police. . . )

# Decentralized Exec., Centralized Planning

Still collaborative

Setting:

- each agent acts and observes on its own
- communication modelled by actions, if any
- but strategy/policy can be agreed upon beforehand

Examples:

- soccer
- exploration
- rescueing
- card games (Hanabi, Bridge. . . ). . .

# Outline

Some Planning Settings

Reasoning with Knowledge: The Single-Agent Case

Reasoning with Knowledge in QDec-POMDPs

More Challenges

# Minesweeper

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

# Minesweeper

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

# Minesweeper

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

# Minesweeper

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | 0 |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | 0 |
| ? | 2 | ? |
| ? | ? | ? |

---

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | 1 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | 1 |
| 0 | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

# Minesweeper

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | 0 |
| ? | 2 | ? |
| ? | ? | ? |

---

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | 1 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | 1 |
| 0 | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

---

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

# Minesweeper

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 1 | 1 | 0 |
| ? | 1 | 0 |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | 1 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 1 | 1 |
| 0 | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| ? | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | ? | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

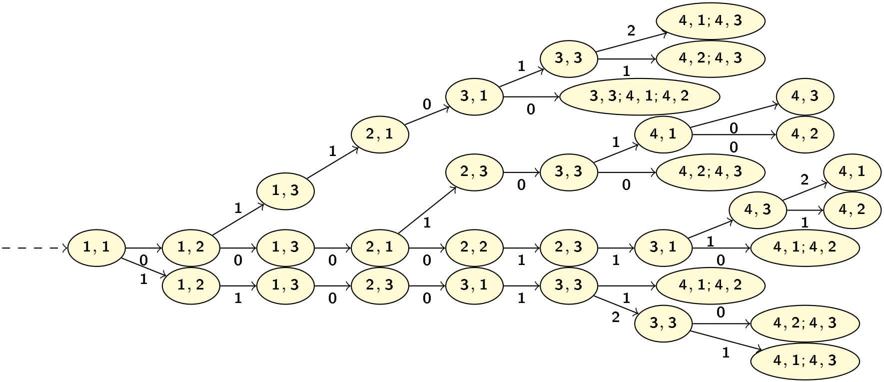
| | | |
|---|---|---|
| 0 | 0 | ? |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| ? | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

# Observation-Based Policy

Obvious, tree representation:




→ huge policies, huge search space

# Single-Agent Knowledge

Knowledge state: all states of the problem consistent with observations

Observations:

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | ? |
| ? | 2 | ? |
| ? | ? | ? |

Knowledge state: {

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| * | 2 | 0 |
| * | 2 | 0 |

,

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| * | 2 | 0 |
| 2 | * | 1 |

,

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| * | 2 | 0 |
| 1 | 2 | * |

}

# Knowledge for Planning: Many Uses

- MBP (Bertoli *et al.*, 2003):
  represent knowledge states compactly as OBDDs
  planning by forward search
- Contingent-FF (Hoffmann & Brafman, 2005):
  check equivalence of histories at planning time
- Complexity of belief tracking = reasoning about histories
  Bonet & Geffner, 2014; Brafman & Shani, 2016

- Finite-state controllers: policy as transducer observations/actions
- $\alpha$-vectors for POMDPs:
  - assume reward of a is 10 in $s_1$, $-1$ in $s_2$, dual for b
  - compact representation of policies by $\vec{\alpha}(a) = (10, -1)$, $\vec{\alpha}(b) = (-1, 10)$
  - execute $\mathrm{argmax}(\vec{\alpha}(a) \cdot \vec{bs})$

# Some Features of Known Approaches

Pros:

- towards efficient planning algorithms
- ensure efficient execution

Cons:

- do not solve pb of succinct policies
- essentially reason belief-state-wise
- not readable

# Proposal: Use Knowledge-Based Programs

KBPs: specifying/verifying protocols (Fagin, Halpern, '90's)
Use for representing policies (with Lang, Saffidine, Schwarzentruber):

<u>while</u> $\neg \mathbf{K} \bigwedge_{i,j}(c_{i,j} \oplus m_{i,j})$ <u>do</u>
    <u>if</u> $\mathbf{K}\neg m_{1,1}$ <u>then</u> $\text{click}_{1,1}$ <u>fi</u>
    <u>if</u> $\mathbf{K}\neg m_{1,2}$ <u>then</u> $\text{click}_{1,2}$ <u>fi</u>
    ...
    <u>if</u> $\mathbf{K}\neg m_{H,W}$ <u>then</u> $\text{click}_{H,W}$ <u>fi</u>
<u>od</u>

Branching condition = compact representation of sets of KSs
Execution:

- maintain one's knowledge by progression through observations
- evaluate branching conditions in current knowledge

# Some Features of KBPs as Plans

Drawback:

- computing what action to take: $\Theta_P^2$-hard ($= P^{NP[\log n]}$)
- tradeoff succinctness/execution
- suitable in not-so-real-time settings

Succinctness:

- always as succinct as standard representations
- possibly exponentially more so
- for some problems, than any reactive representation

Interpretability:

- readable, understandable
- “writeable:” let experts write policies, check auto.
- easier to write generic policies

# Algorithmic Challenges

Efficient execution:

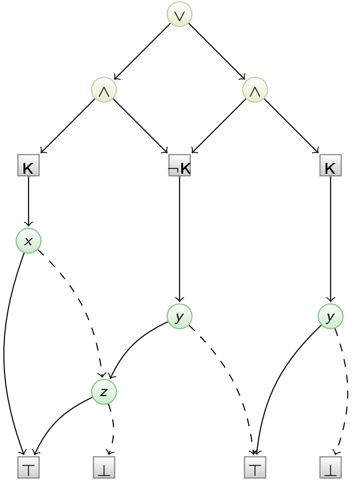
- maintain agent's knowledge
  *explicit: OBDDs, implicit: SAT approaches*
- evaluate branching conditions
- knowledge compilation questions

Efficient planning:

- very challenging
- forward search not so useful
- idea: start from policy tree and generalize
  online regression (Rintanen, 2004)
- regression algorithms: need efficient languages
  sets of sets of prop. assignments/sets of Boolean functions

# Some Languages

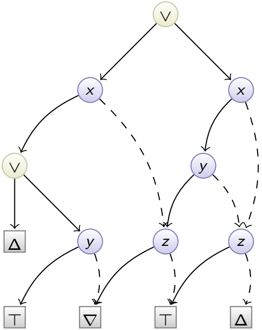
Bienvenu, Fargier, Marquis 2010:




S5-DNF$_{\text{DNF,CNF}}$ (BFM 2010):

$$\bigvee \begin{cases} \mathbf{K}(ab \vee cd) \wedge \neg\mathbf{K}((x \vee y) \wedge (\bar{a})) \\ \mathbf{K}(x) \wedge \neg\mathbf{K}(z \vee \bar{x}) \wedge \neg\mathbf{K}(a \vee b) \\ \neg\mathbf{K}(t) \end{cases}$$

ESD (Niveau & Z. 2016):



# Outline


Some Planning Settings

Reasoning with Knowledge: The Single-Agent Case

Reasoning with Knowledge in QDec-POMDPs

More Challenges

# Setting of Decentralized Execution

Reminder:

- execution is decentralized
- planning can be centralized
- policies: per agent, history $\mapsto$ action

# Toy Example

*Alice in Toulouse, wants to join Bob in Paris. Can fly, but if pilot strike, will take train. Bob wants to pick her up at Paris airport or station.*

Possible histories:

1. strike, web_avail, web_informs . . .
2. strike, ¬web_avail, ¬radio_avail . . .
3. strike, ¬web_informs, ¬radio_informs
4. ¬strike . . .
5. etc.

# Using Knowledge

Knowledge about the situation is not enough

- A knows strike, web_avail, web_informs
- must know that B browses the web
- agents must know each other's plan

# Some Uses of Knowledge

Occupancy states (Dibangoye *et al.*, 2016):

- occupancy state = state + probability of joint histories
- sufficient planning-time statistics

Epistemic Planning (Bolander, Aucher. . . ):

- actions with explicit epistemic effects
- typical example: broadcast/gossip
- essentially

Distributed setting (Engesser *et al.*, 2017):

- no centralized planning (crisis management)
- reason on what plans other can compute
- essentially studied in sequential setting

(With Lang, Saffidine, Schwarz)

(A)

1. if $\neg$strike then fly else take train
2. if train then
   - 2.1 if web_avail then browse web
   - 2.2 if radio_avail then listen to radio
3. if $\mathbf{K}_A\mathbf{K}_B$strike then go to station else go to airport

(B)

1. wait
2. do
   - 2.1 browse web
   - 2.2 listen to radio
3. if strike $\wedge$ $\mathbf{K}_B\mathbf{K}_A\mathbf{K}_B$strike then go to station else go to airport

# Execution

Requires each agent to evaluate epistemic formulas

Essential assumption: each one knows other agents' plans

Possible histories:

1. strike, web_avail, web_informs ...: $K_A K_B$strike
2. strike, ¬web_avail, ¬radio_avail ...: $\neg K_A K_B$strike
3. strike, ¬web_informs, ¬radio_informs: $K_B \neg K_A K_B$strike
4. ¬strike ...: $K_A \neg K_B$strike
5. etc.

# Key Features and Challenges

Pros and cons:

- essentially same as single-agent setting
- hard execution (PSPACE-complete)
- exponential gains on succinctness

Representation languages for knowledge:

- same knowlege compilation questions
- space of multi-agent KS is huge/infinite
- use of compact languages:
  through “programs”: Charrier
  BDDs: Gattinger, Gamblin & Niveau

# Outline

Some Planning Settings

Reasoning with Knowledge: The Single-Agent Case

Reasoning with Knowledge in QDec-POMDPs

More Challenges

# Going Further

Approximate reasoning:

- bound epistemic depth
- makes sense with bounded reasoners
- approx. valid plan...
- *vs* valid with approx. execution
- *vs* valid with approx. knowledge
- logics of belief

Learning:

- reinforcement learning (unknown model)
- build on PSRs (Littman)
- embedding into spaces of epistemic features