Knowledge-Based Programs as Explainable Policies for Contingent Planning

J. Lang, A. Saffidine, F. Schwarzentruber, B. Zanuttini

MAFTEC, April 1, 2019
Planning Problems

Let’s design an agent for solving problems!

Maybe even let the agent compute its policy by itself.
Before we send the agent to the mine field... 
Let’s just check how it is planning to behave
Standard Policies

Before we send the agent to the mine field... Let's just check how it is planning to behave.

Wouldn't this lack of a little readability, verifiability... explainability?
Knowledge-Based Programs

What about this behaviour?

```plaintext
while not sure that all positions except mines have been cleared do
  if sure that there is no mine at (1, 1) then click_{1,1} fi
  if sure that there is no mine at (1, 2) then click_{1,2} fi
  ... 
  if sure that there is no mine at (H, W) then click_{H,W} fi
od
```

Wouldn't this be perfectly readable, verifiable... explainable?
Outline

Contingent Planning Problems

Standard Representations

Knowledge-Based Programs

The Bright Side of KBPs as Policies

The Dark Side of KBPs as Policies

Conclusion

Multi-Agent KBPs

Synthesis of KBPs

More Succinctness?
Outline

Contingent Planning Problems

Standard Representations

Knowledge-Based Programs

The Bright Side of KBPs as Policies

The Dark Side of KBPs as Policies

Conclusion

Multi-Agent KBPs

Synthesis of KBPs

More Succinctness?
Partially Observable Domains

- $X = \{x_1, \ldots, x_n\}$: propositional variables $\rightarrow 2^n$ states
- $A = \{a_1, \ldots, a_k\}$: actions
- $O = \{o_1, \ldots, o_p\}$: observations
- $\varphi^\delta$: transition function

States are not directly observable

Minesweeper $H \times W$:
- variables $X$: $m_{i,j}, c_{i,j} \ (\forall i,j)$ $\rightarrow$ space of $2^{2HW}$ states
- actions $A$: click$_{i,j} \ (\forall i,j)$
- observations $O$: $o_0, \ldots, o_8 + o_{\text{lost}}$
Actions

Actions:
- **ontic effects**: change current state (nondeterministic)
- **epistemic effects**: yield observation (nondeterministic, ambiguous)

Description for Minesweeper:

\[
\varphi^\delta = \bigwedge_{i,j} (\text{click}_{i,j} \rightarrow \varphi^\delta_{i,j})
\]

with

\[
\varphi^\delta_{i,j} = c'_{i,j} \land (m_{i,j} \rightarrow o_{\text{lost}}) \land (\neg m_{i,j} \rightarrow \bigwedge_{n=0,...,8} (\varphi_{n,i,j} \leftrightarrow o_n)) \land \bigwedge_{x \neq c_{i,j}} (x' \leftrightarrow x)
\]
Planning Problems

- Domain + initial belief state + goal states
- Same as POMDPs except for proba.

Minesweeper:
- initial belief state:
  \[ \bigwedge_{i,j} (\neg c_{i,j}) \land \bigvee (m_{i,j} \land m'_{i',j'}) \land \bigwedge_{i,j} (\neg (m_{i,j} \land m'_{i',j'} \land m''_{i'',j''})) \]
- goals: \[ \bigwedge_{i,j} (c_{i,j} \oplus m_{i,j}) \]
Policies

- Prescribe the agent what action to take
- Cannot be as a function of current state
- Function from histories actions/observations to actions
- Abstract notion

Examples for Minesweeper:

- let \((p_t)_t = \langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 1, 3 \rangle \ldots\)
- \(\pi\) def. by \(\pi := \text{click}_p\)
- \(\pi'\) def. by
  \[
  \begin{aligned}
  \pi'(\epsilon) &= \text{click}_{p_0} \\
  \pi'(h) &= \text{click}_{p_{t(h)+1}} \quad \text{if } o_{|h|^{-1}(h)} = o_0 \\
  \pi'(h) &= \text{click}_{p_{t(h)+2}} \quad \text{otherwise}
  \end{aligned}
  \]
Valid policies

Execution model: at each $t = 0, 1, \ldots$

- current state $s^t$ (nonobservable)
- current history $h^t = a^0 o^0 a^1 o^1 \ldots a^t o^t$
- action $a^t = \pi(h^t)$ executed (or “stop”)
- observation $o^t +$ new state $s^{t+1}$ chosen nondet. wrt $\varphi^\delta$
- $o^t$ given to agent
- $s^{t+1} =$ new current state
- new current history $h^{t+1} = h^t a^t o^t$

Valid policy: $\forall s^0 \models \varphi^I$, terminate in finite time $t$ and $s^t \models \varphi^G$
Example policy
Example policy
Example policy
Example policy
Example policy
Outline

Contingent Planning Problems

Standard Representations

Knowledge-Based Programs

The Bright Side of KBPs as Policies

The Dark Side of KBPs as Policies

Conclusion

Multi-Agent KBPs

Synthesis of KBPs

More Succinctness?
Policy Trees

The natural representation:
- Node = action, edge = observation
- One history = one branch
- No child: stop

Usage:
- typical output of planners
- policy typically found as a tree
- DAGs when equivalent situations detected
- very verbose
Finite-State Controllers

Natural compaction of trees:

Usage:
- direct search in policy space
- representation of infinite policies
Common Properties

Note:

- there are other representations
- implicit representation as DAG (Brafman & Hoffmann 2005)
- with “pseudo-epistemic” literals (Albore, Geffner & Palacios 2009)

Common properties:

- branching on observations
- “reactive”: execution is instantaneous at each timestep
- unreadable
Outline

Contingent Planning Problems

Standard Representations

Knowledge-Based Programs

The Bright Side of KBPs as Policies

The Dark Side of KBPs as Policies

Conclusion

Multi-Agent KBPs

Synthesis of KBPs

More Succinctness?
Syntax

Essentially defined by Fagin et al., 1990’s:

\[
\kappa ::= \varepsilon \mid a \mid \kappa ; \kappa \mid \text{if } \Theta \text{ then } \kappa \text{ else } \kappa \text{ fi} \mid \text{while } \Theta \text{ do } \kappa \text{ od}
\]

with \( \Theta \) either

- subjective epistemic formula over variables \( X \)
- \( \text{jo}(o) \) for some observation \( o \)

Note: no auxiliary variable
Example KBP

Minesweeper:

\[
\text{while } \neg K \land \bigwedge_{i,j} (c_{i,j} \oplus m_{i,j}) \text{ do } \\
\text{ if } K \neg m_{1,1} \text{ then } \text{click}_{1,1} \text{ fi } \\
\text{ if } K \neg m_{1,2} \text{ then } \text{click}_{1,2} \text{ fi } \\
\ldots \\
\text{ if } K \neg m_{H,W} \text{ then } \text{click}_{H,W} \text{ fi } \\
\text{ od }
\]
Epistemic Logic

Subjective formulas over $X$:

$$\Phi ::= K\varphi \mid \hat{K}\varphi \mid \neg\Phi \mid \Phi \vee \Phi \mid \Phi \wedge \Phi$$

with $\varphi$ propositional over $X$

Minesweeper:

$$Km_{2,1} \wedge K(m_{4,1} \vee m_{4,2} \vee m_{4,3}) \wedge \hat{K}m_{4,1} \wedge \hat{K}m_{4,2} \wedge \neg\hat{K}(m_{4,1} \wedge m_{4,2})$$
Semantics of Subjective Formulas

- **S5 semantics**
- **Belief state** = set of possible states $B$
- $B \models K\varphi$: \( \forall s \in B, s \models \varphi \)
- $B \models \hat{K}\varphi$: \( \exists s \in B, s \models \varphi \)

\[\begin{array}{ccc}
1 & 1 & 0 \\
* & 1 & 0 \\
2 & 2 & 1 \\
* & 1 & 1 \\
\end{array}\]  \[\begin{array}{ccc}
1 & 1 & 0 \\
* & 1 & 0 \\
1 & 2 & 1 \\
0 & 1 & 1 \\
\end{array}\]  \[\begin{array}{ccc}
1 & 1 & 0 \\
2 & 2 & 0 \\
* & 1 & 1 \\
\end{array}\]

\[
K m_{2,1} \\
\land \ K(m_{4,1} \lor m_{4,2} \lor m_{4,3}) \\
\land \ \hat{K}m_{4,1} \land \hat{K}m_{4,2} \\
\land \ \neg \hat{K}(m_{4,1} \land m_{4,2})
\]
Operational Semantics of KBPs

- Agent maintains current belief state $B^t$
  $\rightarrow [\text{if } \Phi \text{ then } \ldots \text{ else } \ldots \text{ fi}]: \text{ decide } B^t \models \Phi$

- Agent maintains last observation received $o^{t-1}$
  $\rightarrow [\text{if } j_o(o) \text{ then } \ldots \text{ else } \ldots \text{ fi}]: \text{ decide } o^{t-1} = o$

- Same for $[\text{while } \Phi \ldots ]$ and $[\text{while } j_o(o) \ldots ]$

- A KBP represents a policy given $B^0$ and $o^{-1}$

Minesweeper: same policy given $B^0 =$

\[
\begin{array}{ccc}
? & ? & ? \\
? & 1 & ? \\
? & 2 & ? \\
? & ? & ? \\
\end{array}
\]

and $o^{-1} = \bot$
Outline

Contingent Planning Problems

Standard Representations

Knowledge-Based Programs

**The Bright Side of KBPs as Policies**

The Dark Side of KBPs as Policies

Conclusion

Multi-Agent KBPs

Synthesis of KBPs

More Succinctness?
Succinctness

Reactive representation of policies: with execution in polytime at each step

Families of contingent planning problems \((\Pi_n)_n\):

- with description polysize in \(n\)
- with valid KBPs \((\kappa_n)_n\) polysize in \(n\)
- with no valid reactive policy \((\pi_n)_n\) polysize in \(n\) \hspace{1cm} (if NP \not\subseteq P/poly)

Notes:

- for any reactive representation
- more succinct than FSCs in the sense of KC (modulo repr. of KBPs)
- proof based on 3SAT but could have been on Minesweeper
A Very Small Clock/A Very Slow KBP

Recall:
- no auxiliary variable in KBP $\rightarrow$ polysize memory
- $+ \text{ polysize } \pi \rightarrow$ termination in at most exponential time, or loop
- valid reactive $\pi$ cannot execute for more than $O(|\pi|2^n)$ steps

With knowledge:
- no auxiliary variable but $2^{2^n}$ belief states
- there is a polysize KBP going through all of them and terminating
- can be used as a $2^{2^n}$-step clock
- a very very small clock... or a very very slow KBP!
Readability and Succinctness

Succinctness:

- embed policy in low-memory device: robot, satellite...
- transmit policy quickly or over low bandwidth: other planet...
- enhances readability/explainability by humans

Readable/explainable policies:

- succinct + clear and simple semantics
- without jo: abstracts away from sensor model

Who cares about readable/explainable policies?

- missions with high cost, high risk, high stake
- policy reviewed/written by experts
- autonomous agents in society: airport surveillance...
Outline

Contingent Planning Problems
Standard Representations
Knowledge-Based Programs
The Bright Side of KBPs as Policies

The Dark Side of KBPs as Policies
Conclusion
Multi-Agent KBPs
Synthesis of KBPs
More Succinctness?
Slow Execution

All representations in the literature:

- reactive representations
- execution in low polynomial time at each step

KBPs:

- deciding next action to take: \( \text{complete for } \Theta^2_P = P^{\parallel NP} \)
- however belief state is maintained

Still:

- \( P^{\parallel NP} \): use SAT solvers
- factored belief tracking (Brafman & Shani, Geffner)
Slow Automated Verification

Verification:

- given: contingent planning problem $\Pi$
- given: some policy $\pi$ (e.g., written by experts)
- decide whether $\pi$ is valid for $\Pi$

Complexity:

- reactive representations: in PSPACE
- KBPs with while: EXPSPACE-hard
- without while: $\Pi^2_P$-hard (vs in NP)
Taking the Best of Both Worlds

Use (knowledge) compilation:
- let experts write KBPs
- automatically verify
- keep/store/transmit succinct representation
- if needed by application, unroll (parts) into reactive representation
- example use case: next plans for robot on Mars

As a specification language:
- initially intended to be so
- literature taking this view (Van Der Meyden)
- automatic refinement of specifications through operational semantics
Outline

Contingent Planning Problems
Standard Representations
Knowledge-Based Programs
The Bright Side of KBPs as Policies
The Dark Side of KBPs as Policies
Conclusion
Multi-Agent KBPs
Synthesis of KBPs
More Succinctness?
Summary

KBPs are

▶ succinct (provably more than FSCs)
▶ more high-level than standard policies
▶ abstract away from sensor model
▶ but harder to execute and verify

Possible usage:

▶ as readable specification language for policies
▶ for low memory or low bandwidth
▶ explainable policies for robots in society
Outline

Contingent Planning Problems
Standard Representations
Knowledge-Based Programs
The Bright Side of KBPs as Policies
The Dark Side of KBPs as Policies
Conclusion

Multi-Agent KBPs
Synthesis of KBPs
More Succinctness?
Planning:

- Allô Bob, tu viens me chercher à l’aéroport demain?
- OK, et à la gare si la grève aérienne se confirme.
- J’ai cassé mon téléphone, on se prévient comment?
- J’écouterai la radio pour savoir s’il y a grève…

Execution by Alice (with strike):

- Alice prend le train, écoute la radio
- Donc Alice sait que Bob sait qu’il y a une grève aérienne
- Donc Alice sait que Bob ira la chercher à la gare
while $\neg K_M(clothesBought)$ do
  if $K_M(DadAtShopA)$ then
    oneStepToShopB
  else if $K_M(DadAtShopB)$ then
    oneStepToShopA
  else
    observeTraffic
Outline

Contingent Planning Problems
Standard Representations
Knowledge-Based Programs
The Bright Side of KBPs as Policies
The Dark Side of KBPs as Policies
Conclusion
Multi-Agent KBPs
Synthesis of KBPs
More Succinctness?
From the Problem Specification

Complexity:
- same as standard policies
- \textit{2-EXPTIME-complete} in the general case
- better complexity for some subclasses

In practice:
- regression from goals
- need for efficient data structures (BDD like)
- work in progress
- with Anaëlle Wilczynski, Alexandre Niveau, Sébastien Gamblin...
From a Standard Policy?

Idea: use power of standard planners...

... and summarize as KBP

Model-agnostic explanations vs explainable by design
Outline

Contingent Planning Problems
Standard Representations
Knowledge-Based Programs
The Bright Side of KBPs as Policies
The Dark Side of KBPs as Policies
Conclusion
Multi-Agent KBPs
Synthesis of KBPs
More Succinctness?
Reasoning about the Past/Future?

Universal plan:

```plaintext
while true do
    for all actions a do
        if K (after this action there will be a plan for G then
            do a
        fi
    od
od
```

Common in logic, but not as specification language!