Flexible and Scalable Partially Observable Planner with Linear Translations

(or Solving and Scaling Up in Wumpus with a Domain-Independent Planner)

Blai Bonet and Hector Geffner
Universidad Simón Bolívar, ICREA & Universitat Pompeu Fabra

Introduction

Planning with sensing = tracking belief states + action selection

Contributions:

– Develop efficient and sound on-line partially observable planner (LW1)
– LW1 is complete for width-1 problems (deterministic actions and sensing)
– Belief tracking done via linear translation as state tracking
– Actions selected by classical planner from linear translation too

Consequences:

– LW1 can be applied on any problem (it is a sound planner)
– LW1 offers guarantees on width-1 problems
– Its scope can be broadened by merging variables

LW1 builds on CLG (Albore et al., 2009) and K-replanner (B and G, 2011)

Model: Multivalued-Variable Formulation of Planning with Sensing

Problem is tuple $P = \langle V, I, G, A, W \rangle$ where:

– V is set of variables X, each with finite domain D_X
– I is set of X-literals defining initial situation
– G is set of X-literals defining goal
– A is set of actions with preconditions and conditional effects
– W is sensing component

Sensing component W comprises:

– Observable variables Y with domain D_Y
– Precondition $Pre(Y)$ that tells when Y is observed
– State formulas $W(Y = y)$ for each Y and $y \in D_Y$
– Formulas $W(Y = y)$ for diffent $y \in D_Y$ must be mutually exclusive
– $Pre(Y)$ and formulas $W(Y = y)$ are in positive DNF

Belief Tracking: $X(P)$

Use untagged knowledge literals KL for each literal L in problem P. Denote with Kx and $K\overline{x}$ the literals $K(X = x)$ and $K(X \neq x)$

1. Basic Translation $X_0(P)$ for problem $P = \langle V, I, G, A, W \rangle$ is classical problem $P' = \langle F', I', G', A' \rangle$ with axioms D':

– $F' = \{KL : L \in \{X = x, X \neq x\}, X \in V, x \in D_X\}$
– $I' = \{KL : L \in I\}$
– $G' = \{KL : L \in G\}$
– $A' = A$ but with each precondition L replaced by KL, and each effect $C \to X = x$ replaced by $KC \to Kx$ and $\neg K\neg C \to \neg K\overline{x}$
– $D' = \{Kx \Rightarrow \bigwedge_{x':x' \neq x} K\overline{x}',\ \bigwedge_{x':x' \neq x} K\overline{x}' \Rightarrow Kx\}$ for all $x \in D_X$ and $X \in V$

2. Action Compilation: Let $C, x \to x'$ be an effect for action a. The compiled effects associated to this effect are all rules $KC, K\neg L_1, \ldots, K\neg L_m \to K\overline{x}$ where $C_i \to x$ are all rules for a that lead to x and L_i is a literal in C_i. The compiled effects for a are all compiled effects for its original effects.

3. Translation $X(P)$ is translation $X_0(P)$ with compiled effects (Palacios and G, 2006).

4. Action Progression: the state s_a that results from s and a in $X(P)$ is the state s_a obtained from s and a in classical problem P' closed with the axioms in D'

5. Adding Observations: Let s_a be the state following the execution of action a in state s in $X(P)$. The state s_a^o that results from obtaining the observation o is:

$$s_a^o = \text{Unit-Literals}(\text{Unit-Resolution-Closure}(s_a \cup D' \cup K_o))$$

where K_o is codification of observation o; i.e., for each term $C_i \wedge L_i$ in $W(Y = y)$ such that o is inconsistent with $Y = y$, K_o contains implication $KC_i \Rightarrow K\neg L_i$. If empty clause obtained, $s_a^o \doteq \bot$

Theorem (Soundness and Completeness for Width-1 Problems) Let P be a partial observable problem of width 1. An execution is possible and achieves the goal G in P iff the same execution is possible and achieves the goal KG in $X(P)$.

Action Selection: $H(P)$

$X(P)$ provides effective way for tracking beliefs through classical progression and unit resolution

However, classical problem P' cannot be used for action selection because sensing and deduction are carried out outside P'

Following K-replanner, we bring sensing and deduction into P' by adding suitable actions:

– actions for making assumptions about observations (optimism in the face of uncertainty)
– actions for capturing the deductions (OR constraints) in D'

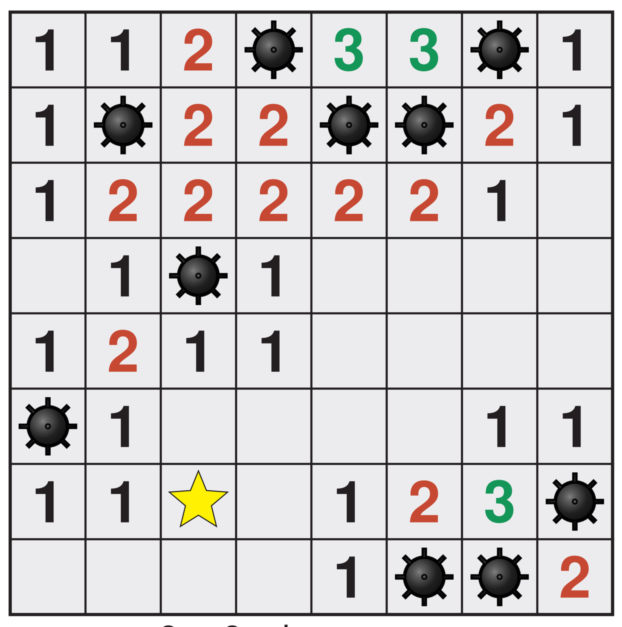
The result is $H(P)$ translation that is a classical problem and can be used for action selection

LW1 = belief tracking done with $X(P)$ + action selection done with $H(P)$

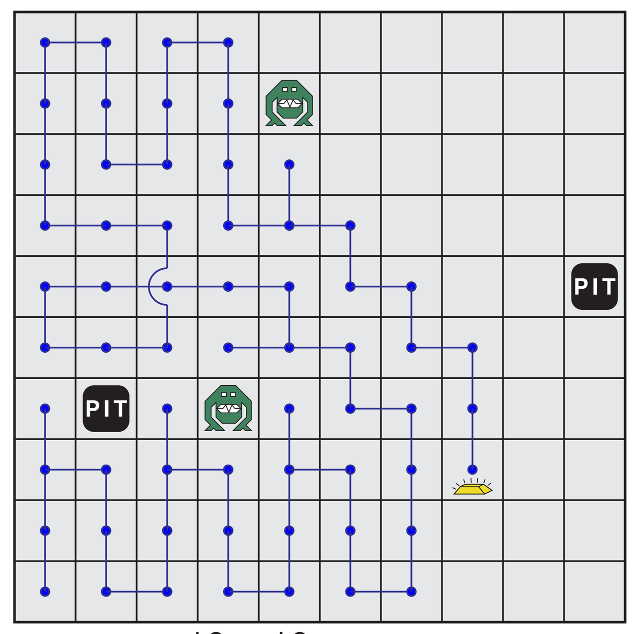
Future Work

– Probabilistic beliefs (POMDPs)
– Optimistic action selection vs. worst-case vs. expected cost

Experimental Results I



8 × 8 minesweeper



10 × 10 wumpus

domain	problem	#sim	solved	average calls	average length	avg. time in seconds total	avg. time in seconds prep	avg. time in seconds exec
mines	4x4	100	35	5.1	18.0	11.3	10.7	0.6
mines	6x6	100	37	9.6	38.0	522.4	506.6	15.8
mines	8x8	100	43	13.1	66.0	3488.2	3365.4	122.7
wumpus	5x5	100	100	12.2	15.2	1.4	0.9	0.4
wumpus	10x10	100	100	54.1	60.5	182.5	173.2	9.2
wumpus	15x15	100	100	109.7	121.0	3210.3	3140.3	70.0

LW1 on Minesweeper and (full) Wumpus

Example: Wumpus in PDDL-like Syntax

```
(define (domain wumpus)
    (:types pos)
    (:predicates (adj ?p ?q - pos) (need-start) (at ?p - pos)
        (wumpus-at ?p - pos) (pit-at ?p - pos) (gold-at ?p - pos) (got-the-treasure)
        (stench ?p - pos) (breeze ?p - pos) (glitter ?p - pos) (alive)
    )

    (:variable agent-pos (forall (?p - pos) (at ?p)))
    (:variable gold-pos (got-the-treasure) (forall (?p - pos) (gold-at ?p)))
    (:variable (wumpus-at-cell ?p - pos) (wumpus-at ?p))
    (:variable (pit-at-cell ?p - pos) (pit-at ?p))
    (:obs-variable (stench-var ?p - pos) (stench ?p))
    (:obs-variable (breeze-var ?p - pos) (breeze ?p))
    (:obs-variable (glitter-var ?p - pos) (glitter ?p))

    (:action start
        :parameters (?j - pos)
        :precondition (and (need-start) (alive) (at ?j))
        :effect (not (need-start))
        :sensing-model %% CUT (SAME AS :sensing-model IN move ACTION)
    )
    (:action move
        :parameters (?i ?j - pos)
        :precondition (and (adj ?i ?j) (at ?i) (alive) (not (need-start)))
        :effect (and (not (at ?i)) (at ?j)
                     (when (wumpus-at ?j) (not (alive)))
                     (when (pit-at ?j) (not (alive))))
        :sensing-model
          (and (forall (?p - pos) (when (and (adj ?j ?p) (wumpus-at ?p)) (stench ?j)))
               (when (forall (?p - pos) (or (not (adj ?j ?p)) (not (wumpus-at ?p)))) (not (stench ?j)))
               (forall (?p - pos) (when (and (adj ?j ?p) (pit-at ?p)) (breeze ?j)))
               (when (forall (?p - pos) (or (not (adj ?j ?p)) (not (pit-at ?p)))) (not (breeze ?j)))
               (when (gold-at ?j) (glitter ?j))
               (when (not (gold-at ?j)) (not (glitter ?j))))
    )
    (:action grab
        :parameters (?i - pos)
        :precondition (and (at ?i) (alive) (gold-at ?i))
        :effect (and (got-the-treasure)); (not (gold-at ?i)))
    )
)

(define (problem p10x10)
    (:domain wumpus)
    (:objects p1-1 p1-2 p1-3 p1-4 ...  p10-7 p10-8 p10-9 p10-10 - pos)
    (:init
      (adj p1-1 p1-2) (adj p1-2 p1-1) (adj p1-1 p2-1) (adj p2-1 p1-1) ...
      (need-start) (not (wumpus-at p1-1)) (not (pit-at p1-1)) (at p1-1) (alive)
    )
    (:goal (got-the-treasure))
    (:hidden (gold-at p8-3) (pit-at p2-4) (wumpus-at p4-4) (pit-at p10-6) (wumpus-at p5-9))
)
```

Experimental Results II

			LW1						K-Replanner with Front End						HCP	
				average		avg. time in seconds				average		avg. time in seconds				
domain	problem	#sim	solved	calls	length	total	prep	exec	solved	calls	length	total	prep	exec	length	time
colorballs	9-5	1000	1000	65.6	126.8	468.2	454.0	14.2	1000	210.4	481.2	725.0	687.9	37.0	320	57.7
colorballs	9-7	1000	1000	69.8	146.1	632.7	615.5	17.1	1000	292.4	613.3	1719.0	1645.9	73.1	425	161.5
doors	17	1000	1000	54.2	114.1	495.3	490.1	5.1	1000	65.0	213.6	88.3	77.1	11.2	143	17.7
doors	19	1000	1000	67.2	140.1	928.2	920.5	7.6	1000	82.7	269.2	143.5	128.5	14.9	184	46.1
localize	15	134	134	9.3	15.2	21.8	5.5	16.3	—	—	—	—	—	—	—	—
localize	17	169	169	10.7	17.2	69.9	20.1	49.7	—	—	—	—	—	—	—	—
medpks	150	151	151	2.0	2.0	10.9	10.0	0.9	151	2.0	2.0	1.3	1.2	0.0	nd	nd
medpks	199	200	200	2.0	2.0	26.0	23.5	2.4	200	2.0	2.0	3.2	3.1	0.1	nd	nd
rocksample	8-12	1000	1000	6.9	191.5	124.2	1.4	122.7	—	—	—	—	—	—	115	0.5
rocksample	8-14	1000	1000	10.2	272.3	22.5	2.7	19.7	—	—	—	—	—	—	135	0.6
unix	3	28	28	17.0	46.5	1.9	1.4	0.4	28	17.0	46.5	1.2	1.0	0.2	42.0	0.6
unix	4	60	60	33.0	93.7	23.0	21.6	1.4	60	33.0	93.7	16.4	15.3	1.1	76.5	7.2
wumpus	20d	1000	1000	5.3	57.2	162.6	160.5	2.0	1000	5.8	69.2	28.9	25.8	3.0	90	5.1
wumpus	25d	1000	1000	5.4	67.3	729.7	724.5	5.1	1000	6.1	80.9	73.5	68.4	5.1	nd	nd

Dash (—) means that the planner cannot solve a domain, and 'nd' means that no data is reported for the instance. Key columns are highlighted in gray.

Width

Width refers to max # of uncertain state variables that interact in a problem, either through observations or conditional effects

Formally:

– X is an **immediate cause** of Y if $X \neq Y$, and either X occurs in body of effect $C \to E$ and $Y \in E$, or X appears in $W(Y = y)$
– X is **causally relevant** to Y if $X = Y$, or X is an imm. cause of Y, or X is causally relevant to Z and Z is imm. cause of Y
– X is **evidentially relevant** to Y if X is observable and Y is causally relevant to X
– X is **relevant** to Y if X is causally or evidentially relevant to Y, or X is relevant to Z which is relevant to Y

Width of X, $w(X)$, is # of state variables relevant to X that are not determined.

http://code.google.com/p/cp2fsc-and-replanner/   bonet@ldc.usb.ve   hector.geffner@upf.edu