Adam GAŁUSZKA
Politechnika Śląska

PLAN RÓWNOLEGŁY ODPORNY JAKO ROZWIĄZANIE ZADANIA PROGRAMOWANIA LINIOWEGO

Streszczenie. Klasyczne planowanie w Sztucznej Inteligencji jest wnioskowaniem mającym na celu znalezienie sekwencji akcji pozwalającej na osiągnięcie sytuacji docelowej wychodząc z danej sytuacji początkowej. W pracy przedstawiono problem planowania równoległego odpornego, tj. planowania w sztucznej inteligencji przy założeniach: niekompletności informacji o stanie początkowym problemu, braku dostępności sensorów umożliwiających redukcję tej niekompletności oraz możliwości wykonywania wielu akcji równocześnie. Poszukiwanie rozwiązania w takim przypadku jest trudne obliczeniowo. Celem zwiększenia efektywności obliczeniowej zaproponowano transformację do Zadania Programowania Liniowego, którą zilustrowano przykładem.

PARALLEL CONFORMANT PLANNING AS A LINEAR PROGRAMMING PROBLEM

Summary. Planning in Artificial Intelligence is a problem of finding a sequence of actions that transforms a given initial state of the problem to a desired goal situation. Conformant planning is a problem of searching for a non-conditional plan for a problem with an uncertain initial state. A parallel plan is a plan in which some actions can be executed in parallel, usually leading to a decrease of the plan execution time. In this work we consider a problem of finding a parallel conformant plan which is computationally difficult. To avoid this difficulty, a transformation of the problem to Linear Programming Problem, illustrated by an example, is proposed.

1. Introduction

Planning as Artificial Intelligence problem is formulated as a searching process leading to a sequence of agent's actions (called a *plan*) that transforms an initial agent environment (called *initial state of a planning problem*) to a desired goal situation (e.g. Russell and Norwig 2009, Skrzypczyk 2010, Palacios and Haffner 2009, Rosa et al. 2011, Galuszka and Swierniak 2010).

The problem becomes more complicated, if information about the modeled world is not sufficient to determine all facts necessary to describe an initial state of the world. Then, we say that the initial state of the problem is uncertain but can be represented

by a set of possible initial states. A plan for solving such problem may take the form of actions that are executed conditionally, based on new information emerging during the search for the plan. This approach is called *conditional planning* (Russell and Norwig 2009, Weld et al. 1998).

In some cases, information from sensors may be unavailable e.g. sensors are damaged or broken down, receiving sensory information is too expensive or dangerous. Then, it is reasonable to search for a plan that is a solution to the planning problem independently of possible initial states. This approach is called *conformant planning* (also Russell and Norwig 2009, Weld et al. 1998). Both conditional and conformant planning are more difficult to solve than a classical planning (Bonet 2010).

The cases, in which more than one action can be applied in one planning step, i.e. some actions can be performed simultaneously, constitute a large class of important planning problems. Such problem formulation allows to model multi-agent and multi-robot environments and is called *parallel planning* (e.g. Ghooshchi et al. 2015). Combining *conformant* and *parallel planning* leads to a problem, in which many agents interact in an uncertain environment with no possibility of performing sensing actions. Finding a solution to a parallel conformant planning problem is more difficult than for previous problems. To avoid this difficulty, in the paper we propose a heuristics for transformation of the problem to a Linear Programming Problem, illustrated by an example.

Contribution. In the paper a polynomial transformation of planning problem to LP problem, proposed in (Bylander 1997) and developed in (Galuszka 2011), is extended to handle parallel conformant planning. The problem of transforming conformant planning problems with single action in each planning step to LP has been studied earlier (Galuszka et al. 2015).

In chapter 2 planning and conformant planning problems are formulated, in chapter 3 the transformation of conformant planning to LP as well as an example are presented.

2. Preliminaries

Following Bylander (1994) it is assumed that classical planning problem is denoted by Π (also called STRIPS planning) and consists of four sets $\Pi=\{C, O, I, G\}$, where:

- C is a finite set of *conditions*,
- O is a finite set of actions, where each action $o \in O$ takes the form $c^+, c^- \rightarrow c_+, c_-$, where:
  - $c^+ \subseteq C$ are called *positive preconditions*,
  - $c^- \subseteq C$ are called *negative preconditions*,
  - $c_+ \subseteq C$ are called *positive postconditions*,
  - $c_- \subseteq C$ are called *negative postconditions*,
- $I \subseteq C$ is an *initial state*,

- $G = \{G_+, G_-\}$ is a *goal situation*, where $G_+ \subseteq C$ are positive conditions (i.e. are true) and $G_- \subseteq C$ are negative conditions (i.e. are false).

In order to include information that some conditions are unknown (assume k conditions can be true or false) in the description of the current problem state, one can introduce so called *k-states* proposed by Baral et al. (2000). In simple terms, the *k-state* is a pair (s, Σ), where s is the current problem state, and Σ is a set that consists of all possible initial states I. For unknown initial state the set Σ consists of all states s, for which:

- condition $c \in C$ is true in the initial state (i.e. $c \in I$),
- condition $c \in C$ is false (i.e. $\neg c \in I$),
- if it is unknown whether condition $c \in C$ is true or false in the initial state, then the set Σ includes both states, for which this condition is true and false, respectively.

The initial state I can be potentially any state from states included in set Σ. It follows that planning problem with incomplete information about initial state takes the form:

$$\Pi = (C, O, \Sigma, G). \quad (1)$$

The result of applying action to the current state is presented below, is based on (Baral 2000) and is adopted to STRIPS problem.

For action o, *k-state* is described by a set $\{Result(S,<o>), Result(\Sigma,<o>)\}$, where $Result(S,<o>)$ is the same like incase with complete information, e.g.:

$$Result(S, \{\}) = S,$$
$$Result(S, \{o\}) = (S \cup c_+) \setminus c_- \text{ if } c^+ \subseteq S \wedge c^- \cap S = \varnothing; S \text{ in opposite case},$$
$$Result(S, <o_1, o_2, ..., o_n>) = Result(Result(S, <o_1>), <o_2, ..., o_n>),$$
$$Result(\Sigma, <o>) = \{ Result(S', <o>) \mid S' \in \Sigma \}.$$

The plan $\Delta_C = <o_1, o_2, ..., o_n>$ solves conformant planning problem, if $Result(\Sigma, \Delta_C) = G$. Since all actions in Δ_C are ordered, Δ_C is called a *total order conformant plan*.

The *partial-order conformant plan* is denoted as $\Delta_{POC} = \{\Delta_{SetC}, \pi\}$, where $\Delta_{SetC} = \{o_1, o_2, ..., o_n\}$ is the set of actions, and π is the non-returnable partial order defined on Δ_{SetC} (compare Backstrom 1998). So, a *partial-order conformant plan* is a compact representation of a set of possible *total* ordered plans.

The *parallel partial-order conformant plan* is denoted as $\Delta_{PPOC} = \{\Delta_{SetC}, \pi, \#\}$, where $\{\Delta_{SetC}, \pi\}$ is Δ_{POC}, while # is a symmetrical relation, defined on the set Δ_{SetC}. $\# \subseteq (\pi \cup \pi^{-1})$ is called a non-concurrency relation and it indicates which actions cannot be applied in parallel.

Example. To illustrate a parallel conformant planning problem consider the following simple *bomb in the toilet* problem (Son Tran Cao et al. 2005) with one action containing conditional effects:

Dunk(P): preconditions: package(P), bomb(B)
effects: if in(P, B) then defused(B), (2)

meaning that if there is a bomb B and there is a package P then action *dunk* causes that bomb B is defused if it was in package P. So if there is no bomb in package, the action *dunk* has no effects.

Action model in formula (2) is different than classical STRIPS action. It is caused by a fact that actions effects are formulated conditionally with general schema: actions causes *set1_of_conditions* if *set2_of_conditions*. Please note that an action defined in such way has no preconditions but action effects are formulated conditionally. It implies that preconditions are indirectly defined by effects, so action *Dunk* is equivalent to two classical STRIPS actions:

$Dunk_1$(P): pre: package(P), bomb(B), in(P, B); eff: defused(B)
$Dunk_2$(P): pre: package(P), bomb(B), not(in(P, B)); eff: no effects

Now, let us consider the following problem Π_{BT} with two possible initial states (bomb is in package 1 or in package 2):

$$\Pi_{BT} = (C_{BT}, O_{BT}, \Sigma_{BT}, G_{BT}), \quad (3)$$

where:
C_{BT} = {package(P1),package(P2),bomb(B),in(P1,B),in(P2,B), defused(B)},
O_{BT} = { Dunk },
Σ_{BT} = {{ package(P1),package(P2),bomb(B),in(P1,B)},
{(package(P1),package(P2),bomb(B),in(P2,B)}},
G_{BT} = { defused(B) }.

The conformant plan that solves the Π_{BT} problem is:

$$\Delta_{CBT} = < \text{Dunk(P1), Dunk(P2)} > \text{ or } \Delta_{CBT} = < \text{Dunk(P2), Dunk(P1)} >. \quad (4)$$

The partial-order conformant plan that solves the Π_{BT} problem is:

$$\Delta_{POCBT} = \{ \text{Dunk(P2), Dunk(P1)} \}. \quad (5)$$

If actions in Δ_{POCBT} can be performed in parallel ($\#_{BT} = \emptyset$), then $\Delta_{POCBT} = \Delta_{PPOCBT}$ and the problem is solved in one step. *End of example*.

3. Translation to Linear Programming problem

Following (Bylander 1997), the transformation from planning to Linear Programming is based on mapping of conditions and operators in each plan step to

variables. Truth values of conditions are mapped to ”0” and “1” for the planning without incompleteness, and to any values between “0” and “1” for planning with incomplete information. The objective function reaches the maximum, if the goal situation is true in the last step of planning.

If l is the number of planning steps and w is the number of possible initial world states then variables of problem (3) for conditions are:

$$c_1(i) = \text{package}(P1,i),\ c_2(i) = \text{package}(P2,i),\ c_3(i) = \text{bomb}(B,i),$$
$$c_4^w(i) = \text{in}(P1,B,i)^w,\ c_5^w(i) = \text{in}(P2,B,i)^w,\ c_6^w(i) = \text{defused}(B,i)^w,$$
$$i = 0, 1, \ldots l, \qquad w = 1, 2, \tag{6}$$

for actions:

$$o_1^w(i) = \text{Dunk}_1(P1,i)^w,\ o_2^w(i) = \text{Dunk}_2(P1,i)^w,\ o_3^w(i) = \text{Dunk}_1(P2,i)^w,$$
$$o_4^w(i) = \text{Dunk}_2(P2,i)^w,\ i = 0, 1, \ldots l\text{-}1,\ w = 1, 2. \tag{7}$$

The initial state is a disjunction of two possibilities. It is modelled by a set of equality constraints:

$$\text{package}(P1,0) = 1,\ \text{package}(P2,0) = 1,\ \text{bomb}(B,0) = 1,$$
$$\text{in}(P1,B,0)^1 = 1,\ \text{in}(P2,B,0)^1 = 0,\ \text{defused}(B,0)^1 = 0,$$
$$\text{in}(P1,B,0)^2 = 0,\ \text{in}(P2,B,0)^2 = 1,\ \text{defused}(B,0)^2 = 0, \tag{8}$$

Goal state G_{BT} is reached if condition *(defused B)* is *true* in last planning step in each world, so the objective function of LP is:

$$\text{Max} \leftarrow f = \text{defused}(B,l)^1 + \text{defused}(B,l)^2. \tag{9}$$

The planning problem is solved, if the optimal value of f is equal to 2, meaning that the condition in both worlds is true. It leads to following formulation of optimization problem: *Find minimal number of planning steps l, such that f* = 2.

The set of constraints is given by:

- actions can be applied if preconditions are true (these are inequality constraints), so basing on formula (3) we have:

$$\text{package}(P1,i) \geq \text{Dunk}_1(P1,i)^w + \text{Dunk}_2(P1,i)^w,$$
$$\text{package}(P2,i) \geq \text{Dunk}_1(P2,i)^w + \text{Dunk}_2(P2,i)^w,$$
$$r * \text{bomb}(B,i) \geq \text{Dunk}_1(P1,i)^w + \text{Dunk}_2(P1,i)^w + \text{Dunk}_1(P2,i)^w + \text{Dunk}_2(P2,i)^w,$$
$$\text{in}(P1,B,i)^w \geq \text{Dunk}_1(P1,i)^w,$$
$$\text{in}(P2,B,i)^w \geq \text{Dunk}_1(P2,i)^w,$$
$$(1 - \text{in}(P1,B,i)^w) \geq \text{Dunk}_2(P1,i)^w,$$
$$(1 - \textit{in(P2,B,i)}^w) \geq \textit{Dunk}_2\textit{(P2,i)}^w,\ i = 0, 1, 2, \ldots l\text{-}1, w=1,2\ , \tag{10}$$

where r is a natural number indicating how many actions can be performed in parallel (in our example $r = 2$),

- changes of variables for conditions due to action application (these are equality constraints), so basing on formula (3) we have:

$$\begin{aligned} &\text{defused(B, i + 1)}^{\text{w}} = \text{defused(B,i)}^{\text{w}} + \text{Dunk}_1\text{(P1,i)}^{\text{w}} + \text{Dunk}_1\text{(P2,i)}^{\text{w}} \\ &i = 0, 1, 2, \ldots l\text{-}1, w = 1,2. \end{aligned} \tag{11}$$

The equality constraint (11) should be studied more carefully. If actions $Dunk_1(P1,i)^w$ and $Dunk_1(P2,i)^w$ are applied in parallel in the same planning step, then the value of condition $defused(B, i + 1)^w$ becomes infeasible. In this case, one should introduce an additional balancing variable for each condition in each planning step to avoid infeasibility:

$$\begin{aligned} &\text{defused(B, i + 1)}^{\text{w}} + \text{defused(B, i + 1)}_{\text{b}}{}^{\text{w}} = \\ &\quad = \text{defused(B,i)}^{\text{w}} + \text{Dunk}_1\text{(P1,i)}^{\text{w}} + \text{Dunk}_1\text{(P2,i)}^{\text{w}}, \\ &\text{i} = 0, 1, 2, \ldots \text{l-1}, \text{w} = 1, 2. \end{aligned} \tag{12}$$

Basing on formulas (6) to (12) it is easy derive general formulas for any problem (1). Finally, the LP problem takes the following form:

$$\begin{aligned} \max_x \leftarrow f^T x \qquad & Ax \le b \\ & A_{eq}x = b_{eq} \\ & 0 \le x \le 1 \end{aligned} \tag{13}$$

Table 1 presents the optimal solution of the problem (13) as well as two additional test problems with possible initial states given by set of equalities (14) and (15):

$$\begin{aligned} &\text{in(P1,B,0)}^{1} = 1, \text{in(P2,B,0)}^{1} = 0, \text{defused(B,0)}^{1} = 0, \\ &\text{in(P1,B,0)}^{2} = 1, \text{in(P2,B,0)}^{2} = 1, \text{defused(B,0)}^{2} = 0; \end{aligned} \tag{14}$$

$$\begin{aligned} &\text{in(P1,B,0)}^{1} = 1, \text{in(P2,B,0)}^{1} = 0, \text{defused(B,0)}^{1} = 0, \\ &\text{in(P1,B,0)}^{2} = 0, \text{in(P2,B,0)}^{2} = 0, \text{defused(B,0)}^{2} = 0. \end{aligned} \tag{15}$$

In the first one (14) there is a bomb in first package but it is uncertain whether it is in the second package, in second one (15) there is no bomb in second package but it is uncertain whether it is in first package.

It should be noted that values of variables for actions are binary integer, so the solution presented in table 1 can be directly interpreted as a plan:

$\Delta_{\text{PPOCBT}} = \{ Dunk(P2), Dunk(P1)\}$.

In the opposite case, one should apply additional heuristics or methods that lead to a binary integer solution (see e.g. Galuszka 2011).

Table 1

Optimal solution *xopt* for problems (14,15,16)

	LP variable	problem (13)	problem (14)	problem (15)
	package(P1,0)	1	1	1
	package(p2,0)	1	1	1
	bomb(B,0)	1	1	1
world 1	in(P1,B,0)	1	1	1
	in(P2,B,0)	0	0	0
	defused(B,0)	0	0	0
world 2	in(P1,B,0)	0	1	0
	in(P2,B,0)	1	1	0
	defused(B,0)	0	0	0
world 1	Dunk1(P1,0)	1	1	1
	Dunk2(P1,0)	0	0	0
	Dunk1(P2,0)	0	0	0
	Dunk2(P2,0)	0	0	0
world 2	Dunk1(P1,0)	0	1	0
	Dunk2(P1,0)	0	0	0
	Dunk1(P2,0)	1	0	0
	Dunk2(P2,0)	0	0	0
world 1	defused(B,1)	1	1	1
world 2	defused(B,1)	1	1	0
	objective *f*	2	2	1

4. Conclusion and future work

Important planning problems are those where more than one agent interacts with the problem environment simultaneously. They arise in multi-agent and multi-robot environments. It leads to changes in constraints (10). Additionally, it is assumed here that maximal number of actions applied to current problem state is m. It can occur when m agents act on the same problem state or one agent is able to perform m actions at a time. It should be noted that for some problem states it may be impossible to perform m actions at a time, so to the set of inequalities (10) now (i.e. for $|O| = n$) should be extended by additional formula: $o_1 + o_2 + ... + o_n \leq m$.

It should be noted that in real-life problems application of an action to a problem state does not always lead to expected effects. It is particularly important in cases where action outcomes are uncertain, as well, and when a condition that is determined can become undetermined. It leads to changes in constraints (12) and will be considered in future works.

Acknowledgment. *This work has been supported by Institute of Automatic Control BK grant 02/010/Bk_17/0060 in the year 2018.*

REFERENCES

1. Baral Ch., Kreinovich V., Trejo R.: Computational complexity of planning and approximate planning in the presence of incompleteness. Artificial Intelligence, 122, 2000, p. 241-267.
2. Bonet B.: Conformant plans and beyond: Principles and complexity. Artificial Intelligence 174, 2010, p. 245-269.
3. Bylander, T.: The Computational Complexity of Propositional STRIPS Planning. Artificial Intelligence, 69, 1994, p. 165-204.
4. Bylander T.: A Linear Programming Heuristic for Optimal Planning. In AAAI97/IAAI-97 Proceedings, 1997, p. 694-699.
5. Galuszka A., Swierniak A.: Planning in Multi-agent Environment Using Strips Representation and Non-cooperative Equilibrium Strategy. Journal of Intelligent and Robotic Systems, Vol. 58, Issue 3, 2010, p. 239-251.
6. Gałuszka A.: On transformation of STRIPS planning to linear programming. Archives of Control Sciences, Volume 21(LVII), No. 3, 2011, p. 227-251.
7. Gałuszka A., Ilewicz W., Olczyk A.: On Translation of Conformant Action Planning to Linear Programming, 20th Int. Conf. on Methods and Models in Automation and Robotics (MMAR), ISBN: 978-1-4799-8701-6, 2015, p. 353-357.
8. Ghooshchi N.G., Namazi M., Newton M.A., Sattar A.: Transition Constraints for Parallel Planning. Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015, p. 3268-3274.
9. Oglietti M.: Understanding planning with incomplete information and sensing. Artificial Intelligence 164, 2005, p. 171-208.
10. Palacios H., Geffner H.: Compiling Uncertainty Away in Conformant Planning Problems with Bounded Width. Journal of Artificial Intelligence Research 35, 2009, p. 623-675.
11. Rintanen J.: Constructing Conditional Plans by a Theorem-Prover. Journal of Artificial Intelligence Research, 10, 1999, p. 323-352.
12. Rosa T., Jimenez S., Fuentetaja R., Barrajo D.: Scaling up Heuristic Planning with Relational Decision Trees. Journal of Artificial Intelligence Research, 40, 2011, p. 767-813.
13. Russell S.J., Norvig P.: Artificial Intelligence, A Modern Approach. 3rd Edition. Prentice Hall Series in Artificial Intelligence, Berkeley, 2009.
14. Skrzypczyk K.: Time optimal tracking a moving target by a mobile vehicle – game theoretical approach, Przegląd Elektrotechniczny (Electrical Review), R. 86 NR 3/2010, p. 211-215
15. Son T.C., Phan H.T., Gelfond M., Morales A.R.: Conformant planning for domains with constraints: a new approach. In Proceedings of the 20th national conference on Artificial intelligence – Volume 3 (AAAI'05), Anthony Cohn (Ed.), Vol. 3. AAAI Press, 2005, 1211-1216.
16. Weld D.S., Anderson C.R., Smith D.E.: Extending Graphplan to Handle Uncertainty & Sensing Actions. Proc. 15th National Conf. on AI, 1998, p. 897-904.