# CASL Single Lecture Notes

SLN - AIB - 01

2006-12-05

Cognitive Autonomous Systems Laboratory
Department of Computer and Information Science
Linköping university, Sweden

Erik Sandewall

# Methods for Classical Planning

Subject Area: Artificial Intelligence Basic

Date of lecture: 2006-10-23

"Single Lecture Notes" are notes corresponding to one lecture.

Related information can be obtained via the following WWW pages:

| | |
|---|---|
| Linköping university: | http://www.liu.se/ |
| This course: | http://www.ida.liu.se/~TDDA23/ |
| CASL Group: | http://www.ida.liu.se/ext/casl/ |
| The author: | http://www.ida.liu.se/~erisa/ |

# 1 The classical planning problem

Classical planning is concerned with creating a plan in terms of actions that are defined by their preconditions and postconditions in a 'world' that is modelled by a set of distinct properties. The following are the basic definitions for the state of the 'world' at specific points in time, and for actions that can be performed in this 'world'. By 'world' in this context we always mean the context in which the plan is to be executed.

The state of the world is represented as a mapping from propositions $u_i$ to truth-values. The general notation is as in the following example of a state:

$$\{u : \mathrm{T}, v : \mathrm{F}, w : \mathrm{T}\}$$

which assigns the value T to the proposition $u$, F to the proposition $v$, and so on. This notation is easily generalized to the case where the elements of the state can have more than two values. (In this case they are called state variables and not propositions).

However, for the particular case of truth-valued state variables we also use the following notation

$$\{u, \neg v, w\}$$

where one just mentions the proposition itself, and precedes it with a - or $\neg$ sign if it is assigned the value F.

An expression consisting of either only a proposition symbol, or a negation sign followed by a proposition symbol, is called a *literal.* If $p$ is a literal that happens to have the form $\neg v$, then $\neg p$ is taken to mean the literal $v$, that is, negation of negation goes back to the un-negated symbol. The expression $\neg\neg v$ is not a literal.

When one decides on the formal representation for a particular application, one decides on what propositions (or state variables) are going to be used, and what is their intended meaning. A *partial state* is a state assigning values to some of the propositions, but it does not have to assign values to all of them. A *complete state* assigns values to all of them. Both complete states and the empty state (not assigning a value to any proposition) are included among partial states. If we just write *state* we mean complete state.

If $P$ is a partial state where the proposition $u$ is defined, then we write $P(u)$ for the value that $P$ assigns to $u$.

Notice that if $P$ and $Q$ are partial states and $P$ is a subset of $Q$ (written $P \subseteq Q$ as usual) then this means that all the value assignments that are made by $P$ are also made by $Q$, but possibly $Q$ makes some additional assignments as well.

An *action* is represented as a pair $\langle P, E\rangle$ of partial states over the same propositions. In other words, $P$ and $E$ must be sets that mention the same propositions, but normally $P(u)$ is different from $E(u)$ for at least some of the propositions $u$. For example,

$$\langle\{u, v\}, \{u, \neg v\}\rangle$$

represents an action that takes a state where $u$ and $v$ are true, and changes it into a state where $u$ is still true, $v$ has become false, and all other propositions are also unchanged. This captures the intuition of having worlds where things only change if there is an action that explicitly changes them.

Therefore, if $A = \langle P, E \rangle$ is an action and $S$ is a state such that $P \subseteq S$, then we write $A(S)$ for the new state that is obtained as $(S - P) \cup E$, where - means set subtraction.

If $\langle A, B, C, D \rangle$ is a sequence of actions, then the effect of applying those actions on a state $S$ is written $\langle A, B, C, D \rangle[S]$ and is defined as

$$D(C(B(A(S))))$$

and similarly of course for sequences of actions with other length.

A *classical planning problem* is defined as follows. Given a set of actions $A_i = \langle P_i, E_i \rangle$, a complete state $S_0$ called the *initial state*, and a partial state $G$ that is called the *goal state*, find a sequence of actions $A1, A2, ...An$ such that

$$G \subseteq \langle A1, A2, ...An \rangle[S_0]$$

Another way of saying this is as follows: one shall find a sequence of actions $A1, A2, ...An$ with $Ai = \langle Pi, Ei \rangle$ for which there is a sequence of states $S_1, S_2, ...S_n$ and

$$Pi \subseteq S_{i-1}$$

$$S_i = (S_{i-1} - Pi) \cup Ei$$

$$G \subseteq S_n$$

Notice in particular that $P1 \subseteq S_0$ which is the initial state.

Classical planning as now defined can be done by search forward or backward, using the general methods for tree search or graph search that have been shown earlier in the course. However, that view does not make use of the fact that a state is defined as a set of value assignments. Specific methods for planning, or more precisely for *action planning* have been developed that use that structure. Several methods are covered in the textbook; here we only show a few of them.

## 2 Partial order planning

The idea in partial order planning (POP) is to make a search where one starts with an *initial partial plan* and searches for a *satisfactory partial plan*. The intermediate steps in the search represent partial plans that become successively more detailed.

As a first step, one takes the set of actions and adds two more "artificial" actions, called $Start$ and $Finish$, which are defined as follows:

$$Start = \langle \emptyset, S_0 \rangle$$

$$Finish = \langle G, \emptyset \rangle$$

where $\emptyset$ is the empty set, and $S_0$ and $G$ were defined above.

After this, a partial plan in general is defined as a tuple of four elements:

- A set $N$ of actions,

- A set of ordering constraints on members of $N$,
- A set of causal links which are written $A \xrightarrow{u} B$,
- A set of the open preconditions of the actions in $N$

that satisfies the following restrictions:

- The fourth element shall be a set of literals that appear in some precondition of some action in $N$ for which there is no causal link. (Its construction and use is described below).
- The ordering constraints must specify $Start \prec A$ for all members $A$ of $N$ besides $Start$, and $A \prec Finish$ for all members $A$ of $N$ besides $Finish$.

The set of ordering constraints is written as $\prec$. The intended meaning of $A \prec B$ is that $A$ shall occur before $B$.

In a causal link, $A$ and $B$ must be actions and $u$ shall be a literal as defined above.

A causal link $A \xrightarrow{u} B$ can be included in the third of those elements iff all of the following conditions hold:

- $A$ and $B$ are members of $N$
- $u$ is an effect of $A$, that is, if $A = \langle P, E\rangle$ then $u$ is a member of $E$
- $u$ is a precondition of $B$, that is, if $B = \langle P', E'\rangle$ then $u$ is a member of $P'$.

Notice that it is not *necessary* to have such a causal link just because these conditions hold. Causal links are introduced as a part of the plan construction process.

The causal link is pronounced "$A$ achieves $u$ for $B$". The intuition is that $A$ arranges one of the requirements for doing $B$, and therefore one is interested in not having any other action appear in-between in such a way that it 'spoils' this condition for $B$.

A *conflict* in causal links is defined as follows. The causal link $A \xrightarrow{u} B$ conflicts with an action $C$ iff all of the following conditions hold:

- $A, B$, and $C$ are members of $N$
- $C$ has the effect $\neg u$
- neither $C \prec A$ nor $B \prec C$ is in the ordering

Intuitively, such a conflict means that there is a possibility, if the actions in the $N$ component of the partial plan are performed in some order that is permitted by its $\prec$ constraints, that the action $C$ will spoil the precondition $u$ that $A$ has achieved for $B$.

A conflict as just defined can be *resolved* by adding either $C \prec A$ or $B \prec C$ to the ordering constraints of the partial plan.

A partial plan is said to be *consistent* iff both the following hold:

- No cycles in its ordering constraints, $\prec$
- No conflicts in its causal links

A *solution* for the POP planning problem is defined as a consistent partial plan where the set of open preconditions (the fourth element in the plan) is the empty set.

This definition means that all the preconditions in all actions in the plan are guaranteed by the postconditions of some earlier action, and that no other action can come in-between and spoil that precondition. Those cases where a precondition is obtained directly from the starting state is accounted for by the definition of the action called *Start*. The requirement that there are no cycles in $\prec$ means that it is possible to order the actions in $N$ in a sequence where an action $A$ appears before an action $B$ in those cases where the ordering constraints specifies that $A \prec B$.

## An Example

Given a classical planning problem:

$$S_0 = \{u, v, w, x\}$$

$$G = \{\neg u, \neg w\}$$

$$A = \langle\{u, v\}, \{\neg u, v\}\rangle$$

$$B = \langle\{w, x\}, \{\neg w, \neg x\}\rangle$$

$$C = \langle\{u, \neg x\}, \{u, x\}\rangle$$

Reformulation as a POP problem: the set of actions is

$$Start = \langle\emptyset, \{u, v, w, x\}\rangle$$

$$Finish = \langle\{\neg u, \neg w\}, \emptyset\rangle$$

$$A = \langle\{u, v\}, \{\neg u, v\}\rangle$$

$$B = \langle\{w, x\}, \{\neg w, \neg x\}\rangle$$

$$C = \langle\{u, \neg x\}, \{u, x\}\rangle$$

Solution: A partial plan where the set $N$ of actions is

$$\{Start, A, B, Finish\},$$

the ordering constraints are

$$Start \prec A \prec Finish$$

$$Start \prec B \prec Finish$$

and the causal links are

$$Start \xrightarrow{u} A$$

$$Start \xrightarrow{v} A$$

$$Start \xrightarrow{x} B$$

$$Start \xrightarrow{w} B$$

$$A \stackrel{\neg u}{\to} Finish$$

$$B \stackrel{\neg w}{\to} Finish$$

It is easily verified that there are no conflicts. The final state after executing the plan, regardless of the order of executing $A$ and $B$, is

$$\{\neg u, v, \neg w, \neg x\}$$

Notice however that there may be additional solutions, and in particular the solution to the next example is also a solution (although a longer one) to the present example.

## Modified example

Like before except that now

$$G = \{\neg u, \neg w, x\}$$

Outline of solution: A partial plan where the set $N$ of actions is

$$\{A, B, C\},$$

the ordering constraints are

$$Start \prec B \prec C \prec A \prec Finish$$

and the final state is

$$\{\neg u, v, \neg w, x\}$$

Exercise: construct the causal links and verify the solution.

## Linearization of a partial plan

A partial order only imposes some restrictions on the order of the actions in its action set $N$, but for some of the actions the order is maybe not determined. A *linearization* of a partial-order plan is a sequence of actions obtained by taking the actions in the $N$ component of the PP and adding members (constraints) to its $\prec$ ordering so that the ordering of the actions becomes complete.

It is easy to verify the following:

- A linearization of a consistent plan is consistent
- If a classical planning problem is given, this problem is converted to a partial-order planning in the way described above, and a solution to the partial-order planning problem is found, then every linearization of that solution is a solution to the original, classical planning problem.

## The algorithm of Partial-Order Planning

Perform a tree-search or graph-search process where the nodes in the graph are partial-order plans, and where the graph is constructed as follows:

1. Start with a single node consisting of the initial PP (partial-order plan) which is constructed in the way specified above.
2. In each cycle until the process has terminated,
   - Pick one of the nodes.
   - In that node, pick an open precondition $u$ in the fourth element of the PP. Let $B$ be the action (or an action) where $u$ is a precondition.
   - Pick an action $A$ that has $u$ as an effect. This may be an action that is already in $N$, or a new action in the set of available actions.
   - If $A$ is not yet in $N$ then add $A$ to $N$ and $\mathit{Start} \prec A$ to the ordering constraints.
   - Always add $A \prec B$ to the ordering constraints.
   - Add $A \xrightarrow{u} B$ to the causal links.
   - If necessary, add additional ordering constraints to $\prec$ so that the resulting PP is consistent, i.e. there are no conflicts.
   - Revise the fourth element of the PP.
   - If a consistent PP can be constructed in this way and it is a new one (not already a member of the search graph) then add it as a new node to the search graph.
3. While performing step (2) check for the following:
   - If the new node is a solution to the partial-order planning problem then it is a solution: terminate with success.
   - If no new node can be added then the search has failed: terminate with failure.

Note: the above explanation has bypassed the problem of what happens if the same action occurs several times in the plan. To be precise, in that case it is necessary to label those occurrences so that they appear as different elements in the plan. For example, if action $B$ occurs twice, one uses two actions called $B1$ and $B2$ although they have the same definitions in terms of preconditions and effects, that is, the pair $\langle P, E \rangle$. This is necessary because each instance of the action shall play its own role in the ordering restriction $\prec$ and in the causal links.

# 3 Planning graphs

The following notes follow the lecture and they are therefore only a summary of the method of planning graphs. A fuller account is given in the textbook, providing additional properties of the planning graphs and the Graphplan method.

## Construction of planning graphs

Basic idea: do a kind of weakened "forward planning" where all actions are performed in parallel, only ruling out some obviously impossible cases. This provides an upper bound on the space of possible solutions. After that, make additional search within that space.

This is done by calculating sets of literals on successive *levels*. Level 0 consists of the initial state, $S_0$. Level 1 is formed as follows: start with the literals on level 0. For all actions, if all the precondition literals of the action are present on level 0, then add the effects of that action on level 1. Continue in the same way for successive levels.

This means of course that each level can contain both a literal and its negation, and in the extreme case a level can contain all possible literals with their negations. However, depending on the structure of the actions, it does not have to go that far.

This process clearly converges after a while, so after some level all the levels have the same contents.

This structure does provide an upper bound on reachable states, but it is grossly optimistic because preconditions have not been taken sufficiently into account. Anyway, if the goal state should not be a subset of the higher levels after convergence, then the planning problem is unsolvable.

Now insert another kind of information between successive levels: between levels 0 and 1, between levels 1 and 2, and so on. The original levels are called *state levels*; the new ones are called *action levels*. Each action level contains a set of actions: the action level between state level $k$ and state level $k+1$ shall contain all the actions that can be performed from state level $k$, contributing to state level $k+1$ by the construction defined above.

Now add *mutex links* to both state levels and action levels. Mutex links can be between two literals, or between two actions, but not between an action and a literal. The rules are as follows:

- If the literals $u$ and $\neg u$ both occur on a level then they have a mutex link in that level.
- If the actions $A$ and $B$ occur on the same level, they have a mutex link iff one of the following applies:
  - Inconsistent effects: one effect literal in $A$ is inconsistent with an effect literal in $B$
  - Interference: one effect literal in $A$ is inconsistent with a precondition literal in $B$
  - Competing needs: one precondition literal in $A$ is inconsistent with a precondition literal in $B$
- If the literals $u$ and $v$ occur on level $k$, and every possible pair of actions on the preceding action level that produces them have a mutex link, then $u$ and $v$ have a mutex link.

The purpose of the mutex links is to provide more information, or in other words to strengthen the restriction, about what literals can be achieved

The planning graph produces information that may be useful in various ways, for example as an estimate of the 'cost' of achieving a goal. A literal that does not occur on the final level can not be achieved. The achievement cost can be estimated by the level where the literal first appears.

## The Graphplan method

1. Initialize the planning graph.

2. Do the following repeatedly:
   (a) Add one more level to the planning graph.
   (b) Check if all goal literals are present there. If so, try to find the final solution by searching for solutions that are no longer than the present level. If not, or if the search for such a solution fails, then iterate.