Dipartimento di elettronica e informazione
Politecnico di Milano

Artificial Intelligence 2010-11

9. Planning in the Situation Calculus

9.1 What is planning

Planning, or *plan formation*, is the rational activity of building a *plan* (i.e., a sequence of actions) that allows an agent to reach a goal from an initial state or situation. So defined, a planning task looks indistinguishable from a state space problem. Indeed, the difference is no so much in what one wants to achieve (which is very similar in the two cases), but in the methods that are used to do so. In particular, planning methods are based on *symbolic representations of states* and *declarative representations of actions*; on the contrary, as we have already observed (Section 5.5), state space methods are typically based on *iconic representations of states* and *procedural representations of actions* (for the difference between iconic and symbolic representations see Section 3.1).

Two issues deserve some attention:

(i) Why do *symbolic* representations of states go along with *declarative* representations of actions, and *iconic* representations of states with *procedural* representations of actions? Is there anything wrong with, say, iconic representations of states with declarative representations of actions, or symbolic representations of states with procedural representations of actions?

(ii) What are the advantages, if any, of using symbolic representations of states and declarative representations of actions? In other words, why should the planning approach be better that the state space approach?

Issue (i)

By a declarative representation we mean a representation based on a logical language, like First Order Logic (FOL). It is easy to understand that if you choose to use logic to represent actions, you are bound to adopt a symbolic representation of states. Indeed, a logical representation of actions will consist of a set of logical axioms, representing the conditions and effects of action execution. Significant features of states will be described in such axioms; but to do so, it is necessary to use logic also to represent states: therefore, a declarative representation of actions presupposes a symbolic representation of states.

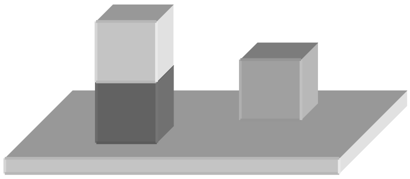
But now, why cannot we use a symbolic representation of states also when actions are represented procedurally? Well, in principle we can. However, there is nothing to gain, here, in avoiding iconic representations of states. In fact, when actions are represented procedurally, any efficient representation of states will do; and iconic representations (better: *mixed* representations, including both iconic and symbolic elements) are typically simpler, more compact, and easier to process than symbolic representations.

Issue (ii)

This is a more difficult issue: indeed, Chapter 10 is devoted to clarify some of the advantages of the planning approach over the state space approach.

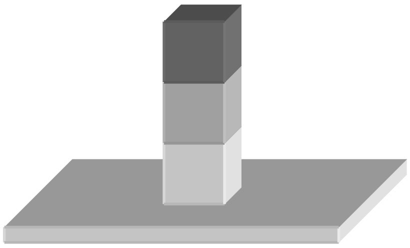
9.2 The Blocks World

The Blocks World is a toy application often used in AI to exemplify different approaches to planning. In a given situation, a number of blocks are placed on a floor like in the following picture:



Situation 0

The problem is to find a plan that allows an agent to achieve a given goal situation, like for example:

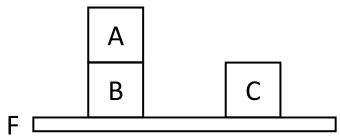


We can represent out Blocks World in various ways. As a first attempt we can:

- name all objects in the world, by introducing suitable constants A, B, C, and F (for "floor")
- classify such objects as blocks or the floor, using predicates Block(x) and Floor(x)
- introduce a predicate, On(x,y), with the intuitive meaning "x is on y"
- introduce another predicate, Clear(x), with the intuitive meaning of "x can take an object on it" (we assume that every block can take exactly one block on it, and that the floor can take any number of blocks)

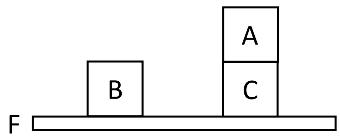
Having done this, we could represent Situation 0 above as:

Block(A)
Block(B)
Block(C)
Floor(F)
On(A,B)
On(B,F)
On(C,F)
Clear(A)
Clear(C)
Clear(F)



Situation 0

But now, how can we represent the fact in a particular Blocks World the situation may change, for example from Situation 0 to Situation 1 below?



Situation 1

If we want to use FOL, we cannot represent change just by removing and adding axioms in a logical theory: logical theories are not like databases, where you can add and delete data as you please. Why? Well, if you remove an axiom from a theory, then you lose all the theorems that you proved before using that axiom: you do not have *the same theory* any longer. Therefore, we must formulate *a single theory* that can simultaneously specify different situations. To solve this problem, we can use Situation Calculus, first proposed by McCarthy and Hayes in 1969.

9.3 Situation Calculus

Situations and fluents

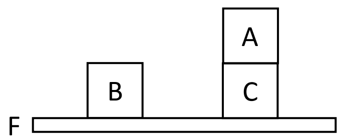
To deal with change we need to represent different situations in the same theory. First we have to distinguish between those facts that *cannot change* in time (like being a block or being a floor) and those that *can change* (like a block being on some object, or a block being clear). For every type of fact that can change, we must specify that the fact does not hold in general, but only in a specific *situation*. A way of doing this is adding a new argument to those predicates (called *fluents*) that represent facts that are subject to change. Below we use constant S_0 to denote Situation 0:

A1. Block(A)
A2. Block(B)
A3. Block(C)
A4. Floor(F)
A5. On(A,B,S_0)
A6. On(B,F,S_0)
A7. On(C,F,S_0)
A8. Clear(A,S_0)
A9. Clear(C,S_0)
A10. $\forall$s Clear(F,s)

Note that Axiom A10 states that the floor is clear in every possible situation.

Actions

Suppose we want to describe the situation obtained by moving block A from block B to block C, starting from S_0:



How are we going to call this situation? Of course we could give it a name, like S_{23}, but this approach would oblige us to predefine a name for every possible situation. There is another option, however: instead of giving this situation a name, we can use a *definite description*; that is, we can identify the new situation as “the result of moving A from B to C, starting from S_0.” We can represent this in FOL with the functional term

$do(move(A,B,C),S_0)$

where:

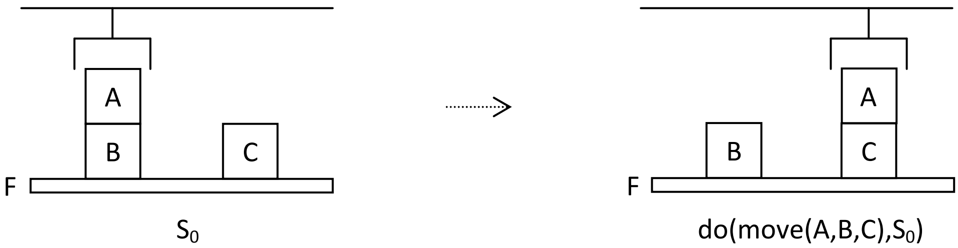
- result and move are function symbols
- a funtional term of the form do(a,s) denotes the situation resulting from doing action a in situation s
- a funtional term of the form move(x,y,z) denotes the action of moving object x from y to z.

Now, if we want to say that in this situation block A is on block C, we can use the statement:

$On(A,C,do(move(A,B,C),S_0))$

which we can read as: A is on C in the situation obtained by moving A from B to C, starting from S_0. Note that the third argument of predicate On is a situation, denoted by the functional term $do(move(A,B,C),S_0)$.

The function symbol move can be regarded as representing a type of action, which could be performed by a mechanical arm:



We now want to describe all possible actions. More precisely, for each possible action we want to specify the *preconditions* (i.e., the conditions under which the action can be performed) and the *effects*. The list of actions of our Blocks World includes 27 different logically possible actions (some of these actions, like move(A,B,B), are useless, but still logically possible):

move(A,B,B)	move(B,A,A)	move(C,A,A)
move(A,B,C)	move(B,A,C)	move(C,A,B)
move(A,B,F)	move(B,A,F)	move(C,A,F)
move(A,C,B)	move(B,C,A)	move(C,B,A)
move(A,C,C)	move(B,C,C)	move(C,B,B)
move(A,C,F)	move(B,C,F)	move(C,B,F)
move(A,F,B)	move(B,F,A)	move(C,F,A)
move(A,F,C)	move(B,F,C)	move(C,F,B)
move(A,F,F)	move(B,F,F)	move(C,F,F)

We shall now try to specify the preconditions and effects of these actions with as few axioms as possible, exploiting the fact that such axioms can be *parametric*.

Axiom A11. Moving x from y to z in situation s: is possible if, in s, x is on y, x is clear, and z is clear (preconditions); and results into x being on z (effect):

A11. $\forall x\, \forall y\, \forall z\, \forall s\, (On(x,y,s) \land Clear(x,s) \land Clear(z,s) \rightarrow On(x,z,do(move(x,y,z),s)))$

This axiom has the form of a conditional, whose antecedent specifies the preconditions for the execution of every action of the form move(x,y,z), and whose consequent specifies an effect of executing such an action.

For example, suppose that in S_0 we try to perform action

$\text{move}(A,B,C)$

With suitable substitutions, from Axiom A11 we derive:

$$On(A,B,S_0) \wedge Clear(A,S_0) \wedge Clear(C,S_0) \rightarrow On(A,C,do(move(A,B,C),S_0))$$

Now we can repeatedly apply the rule of Modus Ponens, exploiting Axioms A5, A8, and A9. Finally we derive:

$$On(A,C,do(move(A,B,C),S_0))$$

Axiom A11, however, does not specify all the effects of actions of the form move(x,y,z). Another effect is that after executing the action, y will be clear (this is trivial if y is the floor, but is also true if y is a block).

Axiom A12. Moving x from y to z in situation s: is possible if, in s, x is on y, x is clear, and z is clear (preconditions); and results into y being clear (effect):

$$\text{A12.} \quad \forall x\ \forall y\ \forall z\ \forall s\ (On(x,y,s) \wedge Clear(x,s) \wedge Clear(z,s) \rightarrow Clear(y,do(move(x,y,z),s)))$$

Now we can derive

$$Clear(B,do(move(A,B,C),S_0))$$

Axioms A11 and A12 describe two effects that we can call “positive,” in the sense that they bring in positive facts. We may also want to consider A11 as specifying the “main effect” of actions of the form move(x,y,z,s), and A12 as specifying a “side effect.” However, we must be cautious: an effect is a main or a side effect only *relative to a goal*. So, if our goal is to have x on z, than A11 describes the main effect, and A12 a side effect; if on the contrary our goal is to clear y, than A12 describes the main effect, and A12 describes a side effect.

An action of the form move(x,y,z) may also bring in a “negative effect”: if z is a block, then it will be no longer clear after executing the action.

Axiom A13. Moving x from y to z in situation s: is possible if, in s, x is on y, x is clear, and z is clear (preconditions); and results into z being no longer clear, if z is a block (effect):

$$\text{A13.} \quad \forall x\ \forall y\ \forall z\ \forall s\ (On(x,y,s) \wedge Clear(x,s) \wedge Clear(z,s) \wedge Block(z) \\ \rightarrow \neg Clear(z,do(move(x,y,z),s)))$$

Now we can derive

$$\neg Clear(C,do(move(A,B,C),S_0))$$

It may seem that, with Axioms A11–A13, we have described all effects of actions of the form move(x,y,z), which, by the way, are all the actions that we can perform in the Blocks World. In a sense, this is true; but our theory is still incomplete, and we shall now see why.

The frame problem

Is A clear in situation $do(move(A,B,C),S_0)$? A quick glance at the graphical representation above suggests that this is the case. However, we are unable to prove $Clear(A,do(move(A,B,C),S_0))$ from Axioms A11–A13. Why? Well, we specified what facts are the effects of move(x,y,z), but we did not specify the facts that are left unchanged by performing an action! This may seem superfluous, but it isn’t. It seems superfluous because we, as human beings, automatically assume that every fact which is not a (positive or negative) effect of an action is left unchanged when we perform the action. This way of thinking is so natural for us, that it is difficult to understand why a logical system may be unable to derive the same consequences.

First, let us remember that the situation we have in mind, depicted above, is just one of the possible models of theory A1–A13. The formulas that are true in such a model are not necessarily true in all models of the theory, and therefore are not necessarily derivable as theorems. In particular, that A is clear in $do(move(A,B,C),S_0)$ is not a theorem, even if it is true in the model that we have in mind. It is plausible to assume that when humans reason about the Blocks World, they do not perform logical inferences, but manipulate some form of “mental model,” from which they can “read” what is true and what is false. But logical systems do not work this way: they work by deriving theorems, and in doing this they keep into account not a single, intended model, but all possible models of the theory. If we want to use logic, then, we have to make our theory stronger, thus reducing the set of models. To be able to derive $\mathrm{Clear}(A,do(move(A,B,C),S_0))$, we have to introduce a new axiom.

Axiom A14. Moving x from y to z in situation s leaves x clear:

$$\text{A14.} \quad \forall x\ \forall y\ \forall z\ \forall s\ (\mathrm{On}(x,y,s) \land \mathrm{Clear}(x,s) \land \mathrm{Clear}(z,s) \rightarrow \mathrm{Clear}(x,do(move(x,y,z),s)))$$

In a sense, this axiom does not describe an "effect" (either positive or negative) of an action of the form move(x,y,z); rather, it says that certain facts are left unchanged by such actions. Axioms of this type are called *frame axioms* (the term, here, comes from the "frame" of cartoons, which is left unchanged while the characters move in the scene). Another frame axiom is:

Axiom A15. Moving x from y to z in situation s leaves y where it is in s:

$$\text{A15.} \quad \forall x\ \forall y\ \forall z\ \forall u\ \forall s\ (\mathrm{On}(x,y,s) \land \mathrm{Clear}(x,s) \land \mathrm{Clear}(z,s) \land \mathrm{On}(y,u,s) \\ \rightarrow \mathrm{On}(y,u,do(move(x,y,z),s)))$$

But now we have a difficulty: in any planning problem, we shall be obliged to specify not only the effects of actions, but also all facts that are left unchanged. Clearly, this will make our theories extremely heavy, in particular when the world is more complex that the Blocks World. Such a difficulty is known as the *frame problem*.

Indeed, the frame problem can be partially solved by adopting a more complex representation of actions, that allows one to specify what is left unchanged using a very limited number of axioms. In any case, it will still be the case that what is left unchanged by an action is not implicit in the representation, but has to be explicitly derived by suitable inferences. This problem has led AI specialist to abandon the use of pure FOL in planning, and to adopt different approaches, one of which will be described in the Chapter 10.

Other problems of the logical approach to planning

The frame problem is not the only difficulty of a logical approach to planning. Two other difficulties are knows as the *qualification problem* and the *ramification problem*.

The qualification problem is the problem of *establishing sufficient conditions* for the effects of an action to take place. For example, Axiom A11 says that x will be on z if we move x from y to z in situation in which: x is on y, x is clear, and z is clear. This means that we consider the conjunction of x being on y, x being clear, and z being clear as sufficient conditions for a successful execution of the action. But is this the case? What happens, for example, if the mechanical arm drops block x while trying to move it from y to z? In other words, our axioms give sufficient conditions for the success of an action only under the assumption that no unforeseen accident occurs: but this is not always the case in the real world.

How can we solve the qualification problem? It is clear that we cannot hope to specify in advance all unforeseen events: this is logically impossible, if we take the term "unforeseen event" seriously. What we can do is to design a system that can carry out *defeasible reasoning*: by this we mean that even if a formula is derived as a theorem, it can be denied later on without introducing a logical contradiction. This type of solution is studied by the branch of logic called *nonmonotonic logic*.

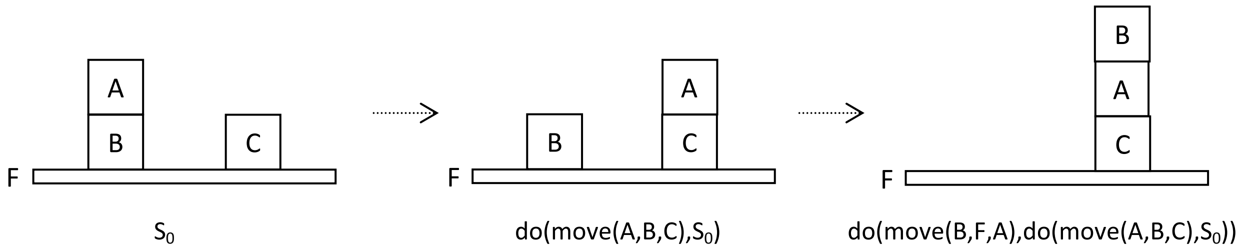
The ramification problem is the problem of keeping track of all side effects of an action. Suppose for example that you drive your car from Milan to Berlin. At the end of the trip, both your car and you will be in Berlin. But so will also be every single object contained in the car. And what about the spare tyre? Well, it should also be in Berlin—but what if you forgot it in your garage? In principle, the problem can be solved by adding suitable axioms; in practice, the resulting theory is going to be exceedingly complex.

9.4 Plan formation in the Situation Calculus

So far we have not discussed how one can build a plan in Situation Calculus. There are three issues here: (i), how a plan can be represented; (ii), how a goal can be stated; and (iii), how a plan that achieves a goal can be automatically built.

Representing plans

A plan is a sequence of actions. Let us consider a simple sequence of two actions:



We start from S_0, and the first action is move(A,B,C), executed in situation S_0; therefore, the second situation is therefore represented by the term do(move(A,B,C),S_0). The second action is move(B,F,A), executed in situation do(move(A,B,C),S_0); therefore the third situation is represented by do(move(B,F,A),do(move(A,B,C),S_0)).

It is easy to see that the functional term that represents the third situation also represents a plan for reaching such this situation from S_0. It is sufficient to process the term *from inside out*:

do(move(B,F,A),do(move(A,B,C),**S_0**))	⟶	start from S_0
do(move(B,F,A),do(**move(A,B,C)**,S_0))	⟶	then move A from B to C
do(**move(B,F,A)**,do(move(A,B,C),S_0))	⟶	then move B from F to A

The same method can be applied to represent plans of any finite length.

Representing goals

The goal to reach in the previous example is to have B on A and A on C. This can be represented as the goal of proving that there is a situation, s, such that: in s, B is on A, and A is on C:

G. $\exists$s (On(B,A,s) $\wedge$ On(A,C,s))

Note that G is not a new axiom, but a sentence that we want to prove as a theorem from A1–A15.

Building plans

To show how a plan can be built automatically, let us first transform our axioms in clausal form, so that resolution theorem proving can be applied.

A1.	Block(A)	A5.	On(A,B,S_0)	A8.	Clear(A,S_0)
A2.	Block(B)	A6.	On(B,F,S_0)	A9.	Clear(C,S_0)
A3.	Block(C)	A7.	On(C,F,S_0)	A10.	Clear(F,s)
A4.	Floor(F)				

A11. $\neg$On(x,y,s) $\vee$ $\neg$Clear(x,s) $\vee$ $\neg$Clear(z,s) $\vee$ On(x,z,do(move(x,y,z),s))
A12. $\neg$On(x,y,s) $\vee$ $\neg$Clear(x,s) $\vee$ $\neg$Clear(z,s) $\vee$ Clear(y,do(move(x,y,z),s))
A13. $\neg$On(x,y,s) $\vee$ $\neg$Clear(x,s) $\vee$ $\neg$Clear(z,s) $\vee$ $\neg$Block(z) $\vee$ $\neg$Clear(z,do(move(x,y,z),s))
A14. $\neg$On(x,y,s) $\vee$ $\neg$Clear(x,s) $\vee$ $\neg$Clear(z,s) $\vee$ Clear(x,do(move(x,y,z),s))
A15. $\neg$On(x,y,s) $\vee$ $\neg$Clear(x,s) $\vee$ $\neg$Clear(z,s) $\vee$ $\neg$On(y,u,s) $\vee$ On(y,u,do(move(x,y,z),s))

Now we formulate the goal by negating the formula to be proved, that is, G:

$\neg$G. $\neg$On(B,A,s) $\vee$ $\neg$On(A,C,s))

Here is a sketch of a proof. Step *k* of the resolution process is identified as R*k*; the next literal to be resolved is underlined; the new variables introduced at step *k* are indexed with *k* to guarantee uniqueness of variable names; the new literals introduced by the current resolution step are in bold.

R0. $\underline{\neg On(B,A,s_0)} \vee \neg On(A,C,s_0))$

R1. resolve R0 with A11: $x_1 = B$, $z_1 = A$, $s_0 = do(move(x_1,y_1,z_1),s_1)$

$\mathbf{\neg On(B,y_1,s_1))} \vee \mathbf{\neg Clear(B,s_1)} \vee \mathbf{\neg Clear(A,s_1)} \vee \underline{\neg On(A,C,do(move(x_1,y_1,z_1),s_1))}$

R2. resolve R1 with A11: $x_2 = A$, $z_2 = C$, $s_1 = do(move(x_2,y_2,z_2),s_2)$

$\neg On(B,y_1,do(move(A,y_2,C),s_2)) \vee \neg Clear(B,do(move(A,y_2,C),s_2)) \vee \neg Clear(A,do(move(A,y_2,C),s_2))$
$\vee \underline{\mathbf{\neg On(A,y_2,s_2)}} \vee \mathbf{\neg Clear(A,s_2)} \vee \mathbf{\neg Clear(C,s_2)}$

R3. resolve R2 with A5: $y_2 = B$, $s_2 = S_0$

$\neg On(B,y_1,do(move(A,B,C),S_0)) \vee \neg Clear(B,do(move(A,B,C),S_0)) \vee \neg Clear(A,do(move(A,B,C),S_0))$
$\vee \underline{\neg Clear(A,S_0)} \vee \neg Clear(C,S_0)$

R4. resolve R3 with A8:

$\neg On(B,y_1,do(move(A,B,C),S_0)) \vee \neg Clear(B,do(move(A,B,C),S_0)) \vee \neg Clear(A,do(move(A,B,C),S_0))$
$\vee \underline{\neg Clear(C,S_0)}$

R5. resolve R4 with A9:

$\underline{\neg On(B,y_1,do(move(A,B,C),S_0))} \vee \neg Clear(B,do(move(A,B,C),S_0)) \vee \neg Clear(A,do(move(A,B,C),S_0))$

The rest of the proof is left to the reader. Note that literal $\neg On(B,y_1,do(move(A,B,C),S_0))$ can be eliminated exploiting the frame axiom A15, which eventually will give the binding $y_1 = F$. The other two literals are then eliminated exploiting the frame axiom A14. Finally we reach the empty clause. Now, we get the plan by spelling out the binding of our initial variable, s_0, which we used to denote the goal situation. Tracing the relevant variable bindings in the proof we have:

$$\begin{aligned} s_0 &= do(move(x_1,y_1,z_1),s_1) \\ &= do(move(B,F,A),do(move(A,B,C),S_0)) \end{aligned}$$

which gives us the plan:

move(A,B,C)
move(B,F,A)

References

McCarthy, J., and P. Hayes (1969). Some philosophical problems from the standpoint of Artificial Intelligence. *Machine Intelligence*, 4, 463–502.