Situation Calculus

CS 475

April 25, 2003

1 Introduction

Situation calculus is an approach to reasoning about action and change (RAC). In this approach, a dynamic domain (a dynamic world, or a world, or a domain) is represented by a situation calculus theory, which is essentially a first-order theory. Unlike first-order theory where variables are of a single type, variables in situation calculus are typed. Two specific types are situation and fluent. A *fluent* is a property of the world that changes over time. The world (at any particular time), called a *situations*, can be described by a set of fluents. To say that fluent F holds in the situation S, we write

$$holds(F, S) \tag{1}$$

We will use a special constant s_0 to denote the initial situation. (In the book, it is called $init$).

For example, in the robot delivery world represented by the following figure,

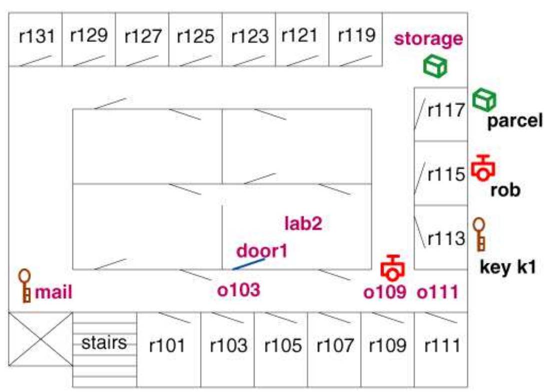

Figure 1: The delivery robot world with a key and a parcel

we can represent the fact that the robot (we will call him *robot*) is at the location $o109$ by the atom

$$holds(at(robot, o103), s_0) \tag{2}$$

Fluents only represent the properties of situations. *Actions* make the world changes from one situation to another situation, i.e., if the agent (or robot) executes an action, it will change the world state. Actions have their *effects* such as the action of *move(robot, o109, o111)* that states that the robot moves from $o109$ to $o111$ will make the robot being at $o111$ after the action is completed. We write $do(a, s)$ to denote the situation resulting from the execution of action a in situation s. With this notation, we can represent the fact that moving from $o109$ to $o111$ ($a = move(robot, o109, o111)$) in situation s, in which the robot is at $o109$, will result in the robot being at $o111$ as follows:

$$\forall S[holds(at(robot, o109), S) \rightarrow holds(at(robot, o111), do(move(robot, o109, o111), S))] \tag{3}$$

Similarly, we can represent the effect of the action *pickup(X, Y)* that says that if X picks up Y, it will carry the object Y as follows:

$$\forall S\ [holds(at(robot, L), S) \wedge holds(at(O, L), S) \rightarrow holds(carrying(robot, O), do(pickup(robot, Y), S))] \tag{4}$$

2 Representing the Robot Delivery World

In the robot delivery world, each situation can be characterized by the map and

- the position of the robot,
- the status of the door, and
- the location of the key and the parcel.

Since the map is rather static (unless we decide to remodel the building:-), it is more convenient for us to consider the map as background knowledge theory that does not change over time. For example, we can represent it using the predicates (relations) $between(D, X, Y)$ and $adjacent(X, Y)$ that say that "door D is between location X and location Y" and "location X is adjacent to location Y, respectively. The map in Figure 1 can then be represented as a set of atoms of the form

$$\begin{array}{ll} adjacent(o109, o103) & adjacent(o103, o109) \\ adjacent(o109, storage) & adjacent(storage, o109) \\ adjacent(o109, o111) & adjacent(o111, o109) \\ adjacent(o111, o103) & adjacent(o103, o111) \\ adjacent(o103, mail) & adjacent(mail, o103) \\ adjacent(r101, r103) & adjacent(r103, r102) \\ \ldots & \end{array} \tag{5}$$

Note: I simplify the representation of the map by assuming the location adjacency is static. (This is different from the book).

In all, we have the following constants

- *robot* – the robot
- *key* – the key needed for open door
- *pacel* – a package at the mail station
- $door1$ – the door 1
- the rooms $r101, r103, \ldots$
- the locations $o109, o111, \ldots$ including the laboratory $lab2$, the mail station $mail$

we can use the following fluents (predicates) to represent a situation (again, I make some simplification to what is presented in the book):

- $at(X, Y)$ – "object X is at the location Y", where X is an object (*robot*, *key*, *parcel*), and Y is a location or a room;
- $carrying(X, Y)$ – "the robot X carries object Y";
- $unlocked(X)$ – "door X is unlocked".

Some formulas about the properties of the world

1. We can represent the knowledge *for every object O, if the robot carries O then O is at the same location as the robot* by the following formula:

$$\forall S\ \forall O\ \forall L[(holds(carrying(robot, O), S) \wedge holds(at(robot, L), S)) \rightarrow holds(at(O, L), S)] \tag{6}$$

2. On the other hands, if the robot and the object O is at different location, then we can be sure that the robot does no carry O. This is described by the formula:

$$\forall S\ \forall O\ [\exists X\ \exists Y((holds(at(robot, X), S) \wedge holds(at(O, Y), S) \wedge X \neq Y) \rightarrow \neg holds(carrying(robot, O), S))] \tag{7}$$

So far, we have only represented the properties of the environment (the relationship between fluents). What is missing here is the representation of actions and their effects. Now, let see which actions are necessary for us. From the description of the environment, we see the following:

1. $move(A, X, Y)$ – agent A moves from location X to location Y
2. $pickup(A, O)$ – agent A picks up object O
3. $putdown(A, O)$ – agent A puts down object O
4. $unlock(A, D)$ – agent A unlocks the door D

We can write down the effect of the action of moving from one location to an adjacent location as follows:

$$\begin{aligned} &\forall L_1 \forall L_2[adjacent(L_1, L_2) \rightarrow \\ &\qquad \forall S(holds(at(robot, L_1), S) \rightarrow holds(at(robot, L_2), do(move(robot, L_1, L_2), S)))] \end{aligned} \tag{8}$$

Question: Formula (8) requires that the two locations L_1 and L_2 are adjacent. Is it really necessary?

How can we represent the effect of the action *pickup*. Since picking up an object means that the agent will carry the object. We can write

$$\forall S\ \forall O[holds(carrying(robot, O), do(pickup(robot, O), S))] \tag{9}$$

But this could be **incorrect** since the robot can only pickup the object if they are at the same location, (9) is true only if the robot and object O are at the same location in the situation S. This means also that $do(pickup(robot, O), S)$ is *not a possible situation* whenever the robot cannot pickup O. We can of course modifying (9) to

$$\begin{aligned} &\forall S\ \forall O[\exists L(holds(at(robot, L), S) \wedge holds(at(O, L), S)) \rightarrow \\ &\qquad holds(carrying(robot, O), do(pickup(robot, O), S))] \end{aligned} \tag{10}$$

The condition $\exists L(holds(at(robot, L), S) \wedge holds(at(O, L), S))$ is called the *precondition* of the action $pickup(robot, O)$. For an action a, $poss(a, S)$, denoted the conditions under which the action a is possible. To simplify the notation, in what follows, we will ignore the universal quantification symbols in the formulae. For $pickup(robot, O)$, we have:

$$poss(pickup(robot, O), S) \leftrightarrow \exists L(holds(at(robot, L), S) \wedge holds(at(O, L), S)) \tag{11}$$

With this, we can rewrite (10) as follows:

$$poss(pickup(robot, O), S) \rightarrow holds(carrying(robot, O), do(pickup(robot, O), S))$$

Similarly, we can have the following axioms for the action $putdown(robot, O)$:

$$\begin{array}{l} poss(putdown(robot, O), S) \leftrightarrow holds(carrying(robot, O), S) \\ poss(putdown(robot, O), S) \rightarrow \neg holds(carrying(robot, O), do(putdown(robot, O), S)) \end{array} \tag{12}$$

Question: Write axioms for the precondition of the action $move(robot, L_1, L_2)$ assuming that the two locations are adjacent.

To specify what is true in the initial situation, we write axioms specifying which fluents are true and which fluents are false. For the robot delivery domain, we have:

$$\begin{array}{c} holds(at(robot, o109), s_0) \\ holds(at(key, mail), s_0) \\ holds(at(parcel, storage), s_0) \\ holds(unlocked(door1), s_0) \end{array} \tag{13}$$

Question: Let T be the theory consisting of the axioms (5), (6), (7), (8), (10), (11), (12), (13). Does the following hold:

$$T \models holds(at(robot, o111), do(move(robot, o109, o111), s_0)).$$

How about the following:

$$T \models holds(carrying(robot, parcel), do(pickup(robot, parcel), do(move(robot, o109, storage), s_0))).$$

3 Situation Calculus

The theory in the previous section is a situation calculus theory. In short, a situation calculus theory consists of

- A set of fluents
- A set of actions
- Situations, with s_0 is a constant denoting the initial situation
- Function *do* that maps a pair of an action and a situation into a situation; $do(a, s)$ denotes the successor situation to s resulting from executing action a
- Predicate *holds* whose first parameter is a fluent and second parameter is a situation
- A set of axioms about the initial situation (what is true/false in the initial situation? e.g. $holds(at(robot, o109), s_0)$ etc.)
- A set of axioms that describes the effects of actions

- A set of axioms that describes the precondition of actions; for each action a, the theory consists of one formula of the form $poss(a, s) \leftrightarrow holds(\varphi, s)$ where φ is a fluent formula.

Given a situation calculus theory (which is essential a set of first order axioms) – under certain conditions we can prove (or predict) what will be true/false after executing a sequence of actions in a situation. This is called the *projection problem*. We next discuss the assumptions needed in solving the projection problem.

We now rewrite the situation calculus theory for the delivery robot world. For convenience, we repeat many of the axioms.

1. Fluents:
   - $at(X, Y)$ – "object X is at the location Y", where X is an object ($robot$, key, $parcel$), and Y is a location or a room;
   - $carrying(X, Y)$ – "the robot X carries object Y";
   - $unlocked(X)$ – "door X is unlocked".
2. Actions:
   (a) $move(A, X, Y)$ – agent A moves from location X to location Y
   (b) $pickup(A, O)$ – agent A picks up object O
   (c) $putdown(A, O)$ – agent A puts down object O
   (d) $unlock(A, D)$ – agent A unlocks the door D
3. Axioms about the initial situation:

$$\begin{array}{c} holds(at(robot, o109), s_0) \\ holds(at(key, mail), s_0) \\ holds(at(parcel, storage), s_0) \\ holds(unlocked(door1), s_0) \end{array} \tag{14}$$

4. Axioms about precondition of actions

$$\begin{array}{lcl} poss(move(A, X, Y), S) & \leftrightarrow & X \neq Y \wedge holds(at(A, X), S) \\ poss(pickup(A, O), S) & \leftrightarrow & \exists L\, [holds(at(A, L), S) \wedge holds(at(O, L), S)] \\ & & \wedge \neg \exists X\, holds(carrying(A, X), S) \\ poss(putdown(A, O), S) & \leftrightarrow & holds(carrying(A, O), S) \\ poss(unlock(A, D), S) & \leftrightarrow & holds(at(A, D), S) \wedge holds(carrying(A, key), S) \end{array} \tag{15}$$

5. Axioms about effects of actions:

$$\begin{array}{lcl} poss(move(A, X, Y), S) & \rightarrow & holds(at(A, Y), do(move(A, X, Y), S)) \wedge \\ & & \forall O[holds(carrying(A, O), S) \rightarrow \\ & & \qquad holds(at(O, Y), do(move(A, X, Y), S)] \\ poss(pickup(A, O), S) & \rightarrow & holds(carrying(A, O), do(pickup(A, O), S)) \\ poss(putdown(A, O), S) & \rightarrow & \neg holds(carrying(A, O), do(putdown(A, O), S)) \\ poss(unlock(A, D), S) & \rightarrow & holds(unlocked(D), do(unlock(A, D), S)) \end{array} \tag{16}$$

With the axioms in (14)-(16) we can write the *successor state axioms* for every fluent:

$$\begin{array}{rcl} holds(at(robot, L), do(Act, S)) & \leftrightarrow & ((Act = move(robot, B, L)) \wedge holds(at(robot, B), S)) \\ & & \vee\ (holds(at(robot, L), S) \wedge (Act \neq move(robot, L, C))) \end{array} \tag{17}$$

$$holds(at(key, L), do(Act, S)) \leftrightarrow ((Act = move(robot, B, L)) \wedge \tag{18}$$

$$
\begin{aligned}
& holds(carrying(robot, key), S)) \\
& \lor (holds(at(key, L), S) \land \\
& \neg(holds(carrying(robot, key), S) \land Act = move(robot, L, C))) \\
holds(at(parcel, L), do(Act, S)) \leftrightarrow \; & ((Act = move(robot, B, L)) \qquad (19) \\
& \quad \land holds(carrying(robot, parcel), S)) \\
& \lor (holds(at(parcel, L), S) \land \\
& \neg(holds(carrying(robot, parcel), S) \land Act = move(robot, L, C))) \\
holds(carrying(A, O), do(Act, S)) \leftrightarrow \; & (Act = pickup(A, O) \land \qquad (20) \\
& \quad \exists L(holds(at(A, L), S) \land holds(at(O, L), S))) \\
& \lor (holds(carrying(A, O), S) \land Act \neq putdown(A, O)) \\
holds(unlocked(D), do(Act, S)) \leftrightarrow \; & (Act = unlock(D) \land \qquad (21) \\
& \quad holds(at(A, D), S) \land holds(carrying(A, key), S)) \\
& \lor (holds(unlocked(D), S))
\end{aligned}
$$

These axioms represent our knowledge about the dynamics of things: "*things tend to stay the same unless it has been changed by an action.*" For example, axiom (17) says that the robot will be at the location L if it moves to L or it is already at L previously and does not move to somewhere else.

The axioms (14)-(16) allow us to prove several things. For example, we can check if the robot is carrying the key after it moves from $o109$ to the mail station and pickup the key, i.e., we can check whether

$$D \models holds(carrying(robot, key), do(pickup(robot, key), do(move(robot, o109, mail), s_0)))$$

where D is the set of axioms given (14)-(16). The steps of the proof is given below:

$$
\begin{aligned}
& holds(carrying(robot, key), do(pickup(robot, key), do(move(robot, o109, mail), s_0))) \leftrightarrow \qquad \text{(because of (20))} \\
& \quad \exists L \; [holds(at(robot, L), do(move(robot, o109, mail), s_0)) \land \\
& \qquad holds(at(key, L), do(move(robot, o109, mail), s_0))] \leftrightarrow \qquad (14) \\
& \quad holds(at(robot, mail), do(move(robot, o109, mail), s_0)) \leftrightarrow \qquad (17) \\
& \quad holds(at(robot, o109), s_0) \leftrightarrow \qquad (14) \\
& \quad \text{true}
\end{aligned}
$$

This also allows us to do planning, i.e., to find a sequence of actions that will change the world to a goal situation from the initial situation. If the goal needs to satisfy a fluent, say f, then the answer of the query `?holds(f,S)` would be a situation of the form $do(a_n, do(a_{n-1}, \ldots, do(a_1, s_0)))$ such that

- $poss(a_i, do(a_{i-1}, \ldots, do(a_1, s_0)))$ is true for $i = 1, \ldots, n$, and
- $holds(f, do(a_n, do(a_{n-1}, \ldots, do(a_1, s_0))))$ is true.

In other words, the sequence of actions $a_1, a_2, \ldots, a_n$ is a plan achieving the goal f. For instance, if our goal is to have the robot carrying the key, then it follows from the above computation that $move(robot, o109, mail), pickup(robot, key)$ is a plan achieving the goal.

Notice that the problem of finding a plan could be viewed as a search problem as well. The search problem is defined as follows:

- A state (of the search problem) is a situation (of the situation calculus theory);
- The set of neighbors of a state S is the set of situations that are reachable from it, i.e., it contains of situation of the form $do(A, S)$ where $poss(A, S)$ is true.

For the delivery robot world, we have the following:

- the initial situation s_0 is one state of the search problem;
- the set of neighbors of s_0 consists of situations of the form $do(A, s_0)$ where A is an action of the form $move(robot, o109, L)$, L is a location (if we allow the robot to move only to adjacency neighbors, we need to add that $adjacent(o109, L)$ is true);
- the set of neighbors of $do(move(robot, o109, o111), s_0)$ is again the set of situations of the form $do(A, do(move(robot, o109, 0111), s_0))$ where A is a move action;
- the set of neighbors of $do(move(robot, o109, mail), s_0)$ contains situations resulting from the action $move$ of the robot but also contains the situation $do(pickup(robot, key), do(move(robot, o109, mail), s_0))$ since $poss(pickup(robot, key), do(move(robot, o109, mail), s_0))$ is true;
- etc...

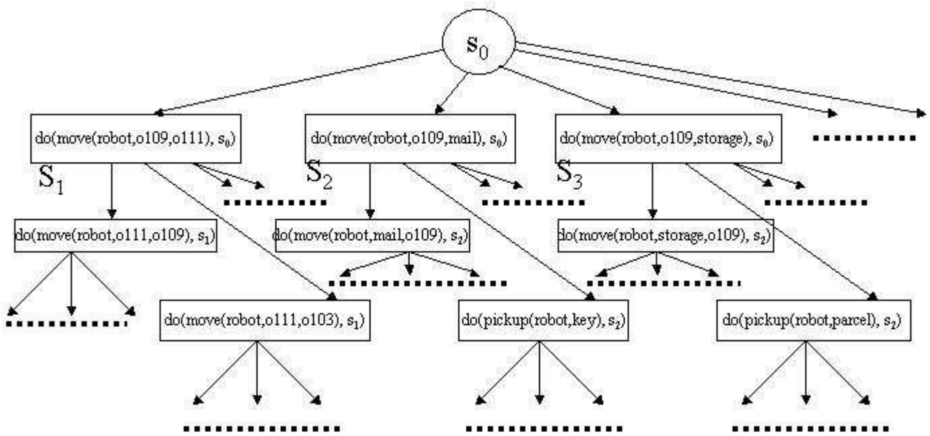

Figure 2: A situation tree for the delivery robot world

Viewing the planning problem as a search problem will allow us to apply the search techniques that we have learned in solving the planning problem. Try to formulate and solve the following problem: what does the robot need to do in order to have the parcel and the key in the $lab2$? Have a look at the planners that the book provides.

4 Another Example: The Block World

The block world is one of the classic examples in AI planning literature. It is quite simple. It consists of a table and cubic blocks of the same size. Let us consider the block world in the following picture.

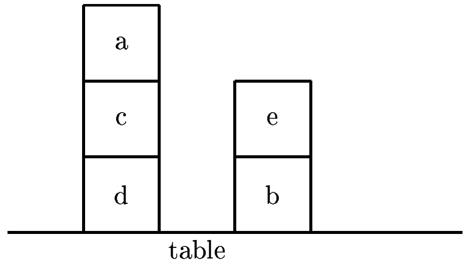

Figure 3: A Block World

We assume that the world consists of the above blocks and an agent who can pick up block and put it down on the table or on top of other blocks which are clear.

Objects of the block world are the blocks, the table, and the agent. The blocks are described by their names, *a,b,c,...*. The table will be denoted by *t*. We simplify the domain by not having a constant to denote the agent/robot.

We will use the predicate $block(X)$ to denote that object x is a block.

The actions of the agents are:

1. $pickup(X)$ - pickup block X,
2. $put(X, Y)$ - put block X on top of block Y, and
3. $put(X, table)$ - put block X on the table.

The fluents of the block world are:

1. $on(X, Y)$ - block X is on the block X,
2. $on(X, table)$ - block X is on the table,
3. $holding(X)$ - (the robot) is holding block X, and
4. $clear(X)$ - block X is clear.

We note that

$$clear(X) \leftrightarrow \neg \forall Y (block(Y) \wedge \neg on(Y, X))$$

which means that some properties of the world can be automatically deduced when we know about other properties.

The initial situation can be described by the following axioms

$$holds(on(a, c), s_0) \tag{22}$$
$$holds(on(c, d), s_0) \tag{23}$$
$$holds(on(d, table), s_0) \tag{24}$$
$$holds(on(e, b), s_0) \tag{25}$$
$$holds(on(b, table), s_0) \tag{26}$$

Now, let us describe the set of precondition axioms and the set of effect axioms.

Precondition axioms: Before an action is executed, it requires that some conditions are true. This condition is called the action's *precondition*. For example, to put down a block X on top of block Y, the robot must hold X and Y must be clear. Another example is that to pick up a block X, the robot must not hold another block and X must be clear etc... We represent these by

$$poss(pickup(X), S) \rightarrow \forall Y(holds(\neg holding(Y), S)) \wedge holds(clear(X), S) \tag{27}$$
$$poss(put(X, Y), S) \rightarrow holds(holding(X), S) \wedge holds(clear(Y), S) \tag{28}$$
$$poss(put(X, table), S) \rightarrow holds(holding(X), S) \tag{29}$$

Problem The above formulas do not allow us to prove when an action, say *pickup*(X), is possible. I.e., when $poss(pickup(X), S)$ is true. Remember how to prove whether a literal is true?

Can we reverse the axioms (change $\rightarrow$ with $\leftarrow$) and then use them to prove when $poss(pickup(X), S)$ is true? No, this will not work since it is not correct! There might be conditions which we have not considered? For example, the block can be too heavy, the agent might be tired etc... So, we need to specify all of these conditions, called *qualifications*. But there are infinitely many conditions like this?

The above is called the *qualification problem*. It calls for the need for *nonmonotonic reasoning* in AI.

To overcome the qualification problem, we make the assumption that the antecedent of the above formula represents the sufficient and necessary conditions for the action to be executed. The *final* action precondition axioms for the above three actions are:

$$poss(pickup(X), S) \leftrightarrow \forall Y(holds(\neg holding(Y), S)) \wedge holds(clear(X), S) \tag{30}$$
$$poss(put(X, Y), S) \leftrightarrow holds(holding(X), S) \wedge holds(clear(Y), S) \tag{31}$$
$$poss(put(X, table), S) \leftrightarrow holds(holding(X), S) \tag{32}$$

We will use the above axioms instead of the axioms (27)-(29).

Given the above axioms and some axioms about the , we can now deduce whether the action *pickup*(A) is possible in the initial situation, i.e., $poss(pickup(A), S_0)$ is true. How?

Effect axioms: Actions have effects. For example, if the agent pickups a block he will hold it in the successor situation.

$$poss(pickup(X), S) \rightarrow holds(holding(X), do(pickup(X), S)) \tag{33}$$
$$poss(put(X, Y), S) \rightarrow holds(on(X, Y), do(put(X, Y), S)) \tag{34}$$
$$poss(put(X, table), S) \rightarrow holds(on(X, table), do(put(X, table), S)) \tag{35}$$
$$poss(pickup(X), S) \rightarrow [holds(on(X, Y), S) \rightarrow holds(\neg on(X, Y), do(pickup(X), S))] \tag{36}$$

But, the above equations only represent the 'positive effects' of actions (what becomes true) and 'negative effects' of actions (what becomes false) as well. In (36), for example, if the agent pickups block x which is on a block Y in the situation S then $on(X, Y)$ will not hold in $do(pickup(X), S)$; or if the agent puts block X on a block Y in the situation S then $clear(Y)$ will not hold in $do(put(X, Y), S)$, etc. **Try to complete the set of effect axioms.**

Furthermore, there are fluents whose values do not change after the execution of an action. For example, if X is on Y in situation S ($holds(on(X,Y),S)$ is true) and the robot puts block Z on top of X, then X is still on Y in situation $do(put(Z,X),S)$ ($holds(on(X,Y),do(put(Z,X),S))$ is true). Representing the unchanged effects of actions require axioms of the form

$$holds(on(X,Y),S) \rightarrow holds(on(X,Y),do(pickup(Z),S))$$

These axioms are called *frame axioms.*

If we have n actions, m fluents, we would need $2 \times n \times m$ frame axioms. This is too many!!!

Representing frame axioms in a compact way is a challenge for quite sometime. To date, there are many solutions to the frame problems. Under reasonable assumptions, we can write, for each fluent F, one *successor state axiom* in the following form

$$holds(F,do(A,S)) \leftrightarrow \gamma_f^+(A,S) \vee (holds(F,S) \wedge \gamma_f^-(A,S)) \tag{37}$$

where $\gamma_f^+(A,S)$ (resp. $\gamma_f^-(A,S)$) summarizes the conditions for the fluent F to become true (resp. false).

For example, in the block world we have

$$holds(on(X,Y),do(A,S)) \leftrightarrow \gamma^+_{on(X,Y)}(A,S) \vee (holds(on(X,Y),S) \wedge \gamma^-_{on(X,Y)}(A,S)) \tag{38}$$

where

$\gamma^+_{on(X,Y)}(A,S) = (A = put(X,Y)) \wedge poss(A,S)$ and

$\gamma^-_{on(X,Y)}(A,S) = (A \neq pickup(X)) \wedge poss(A,S)$.

Try to complete the description at home!

4.1 Projection Problem

Let us denote the situation calculus theory for the block world by $\mathcal{T}$. We can prove that

$$\mathcal{T} \models holds(on(a,c),s_0)$$

$$\mathcal{T} \models holds(\neg on(a,c),do(pickup(a),s_0))$$

In general, the problem of proving

$$\mathcal{T} \models holds(f,do(a_n,\ldots,do(a_1,s_0))) \tag{39}$$

for a fluent f and a sequence of actions $a_1,\ldots,a_n$ is called the projection problem.

We also write

$$\mathcal{T} \models holds(f,do(\alpha,s_0))) \tag{40}$$

$\alpha = [a_1,\ldots,a_n]$.

4.2 Planning in Situation Calculus

The projection shows that we could formulate a planning task by asking for a sequence of actions (α) that makes the goal (say, a fluent f) true in the resulting situation of executing α in s_0. That is, for a planning task of achieving the goal φ (a fluent formula), we ask for a sequence of actions α such that

$$\mathcal{T} \models holds(\varphi, do(\alpha, s_0))) \tag{41}$$

Example 4.1 *Find a sequence of actions that achieves the goal of having a on d. That is, we need to find* α *such that* $\mathcal{T} \models holds(on(a, d), do(\alpha, s_0))$.

It is easy to see that one of the possibility is

$\alpha = [put(a, e), put(c, table), put(a, d)]$. **Why?**

Can you find another sequence of actions?