

INTRODUCTION TO THE SITUATION CALCULUS

The Situation Calculus (McCarthy)

- A first order language for representing dynamically changing worlds; all changes are the result of named *actions*.
- A possible world history, which is simply a sequence of actions, is represented by a first order term called a *situation*.
- S_0 denotes the initial situation, where no actions have yet occurred.
- $do(\alpha, s)$ denotes the successor situation to s resulting from performing the action α .
- Actions may be parameterized:
 $put(x, y)$ might stand for the action of putting object x on object y ; $do(put(A, B), s)$ denotes that situation resulting from placing A on B when the world is in situation s .
- Actions are denoted by function symbols.
- *Fluents* = those relations or functions whose truth values may vary from situation to situation.
 - Denoted by predicate or function symbols taking a situation term as one of their arguments.
 - In a world in which it is possible to paint objects, we might have a functional fluent $colour(x, s)$, that denotes the colour of object x when the world is in situation s .

The Situation Calculus (Continued)

- Actions have *preconditions*: necessary conditions which a world situation must satisfy if the action can be performed in this situation.
 - If it is possible for a robot r to pick up an object x in the world situation s then the robot is not holding any object, it is next to x , and x is not heavy:
$$Poss(pickup(r, x), s) \supset [(\forall z)\neg holding(r, z, s)] \wedge \neg heavy(x) \wedge nexto(r, x, s).$$
 - Whenever it is possible for a robot to repair an object, then the object must be broken, and there must be glue available:

$$Poss(repair(r, x), s) \supset hasglue(r, s) \wedge broken(x, s).$$

- World dynamics are specified by *effect axioms* which specify the effect of a given action on the truth value of a given fluent.
 - The effect on the relational fluent *broken* of a robot dropping an object:

$$fragile(x) \supset broken(x, do(drop(r, x), s)).$$

- A robot repairing an object causes it not to be broken:

$$\neg broken(x, do(repair(r, x), s)).$$

- Painting an object with colour c :

$$colour(x, do(paint(x, c), s)) = c.$$

The Qualification Problem for Actions

- With only the axioms above, we can't prove anything interesting about when an action is possible.
pickup preconditions:
$$Poss(pickup(r, x), s) \supset [(\forall z)\neg holding(r, z, s)] \wedge \neg heavy(x) \wedge nexto(r, x, s).$$
- We can't infer when a *pickup* is possible!
- What about reversing the implication:

$$[(\forall z)\neg holding(r, z, s)] \wedge \neg heavy(x) \wedge nexto(r, x, s) \supset Poss(pickup(r, x), s).$$

- This sentence is *false*! We also need, in the antecedent of the implication:
$$\neg glued_to_floor(x, s) \wedge \neg arms_tried(r, s) \wedge \neg hit_by_10_ton_truck(r, s) \wedge \dots$$
- i.e. all the *qualifications* which must be true in order for a *pickup* to be possible!

- Imagine succeeding in enumerating all the qualifications for *pickup*. Would that help? Not really. Suppose all we know is
$$[(\forall z)\neg holding(R, z, S)] \wedge \neg heavy(A) \wedge nexto(R, A, S).$$
- We still can't infer $Poss(pickup(R, A), S)$ because we don't know that the qualifications are true!

The Qualification Problem (Continued)

- Intuitively, here's what we want:
 - When given only that the “important” qualifications are true:
$$[(\forall z)\neg holding(R, z, S)] \wedge \neg heavy(A) \wedge nexto(R, A, S).$$
and if we *don't know* that any of the “minor” qualifications (*glued_to_floor(A, S)*, *hit_by_10_ton_truck(R, S)*) is true, infer $Poss(pickup(R, A), S)$.
 - But if we happen to know that one of the “minor” qualifications is true, this will block the inference of $Poss(pickup(R, A), S)$.
- Historically, this has been seen to be a problem peculiar to reasoning about actions.
- Not so...
$$bird(x) \wedge \neg penguin(x) \wedge \neg ostrich(x) \wedge \neg peking_duck(x) \wedge \dots \\ \supset fly(x).$$
Given $bird(Tweety)$, want intuitively to infer $fly(Tweety)$.
- Formally, this is the same problem as action qualifications.
 - “Important” qualification: $bird(x)$.
 - “Minor” qualifications: $penguin(x), ostrich(x), \dots$
- This is the classic example of *nonmonotonic reasoning* in AI.

The Qualification Problem (Continued)

- Our approach (for now):
 - Assume that for each action $A(\vec{x})$, we have an axiom of the form
$$Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s),$$
where $\Pi_A(\vec{x}, s)$ is a first order formula with free variables \vec{x}, s which does not mention do .
We shall call these *action precondition axioms*.
 - *Example:*
$$Poss(pickup(r, x), s) \equiv [(\forall z)\neg holding(r, z, s)] \wedge \neg heavy(x) \wedge nexto(r, x, s)$$
- In other words, ignore all the “minor” qualifications.

The Frame Problem (McCarthy and Hayes)

- Axioms other than effect axioms are required for formalizing dynamic worlds. These are called *frame axioms*, and they specify the action *invariants* of the domain, i.e., those fluents unaffected by the performance of an action.
- A positive frame axiom – dropping things does not affect an object's colour:

$$colour(x, s) = c \supset colour(x, do(drop(r, y), s)) = c.$$

- A negative frame axiom – not breaking things:

$$\begin{aligned} &\neg broken(x, s) \wedge [x \neq y \vee \neg fragile(x)] \\ &\quad \supset \neg broken(x, do(drop(r, y), s)). \end{aligned}$$

- Problem: Vast number of frame axioms; only relatively few actions will affect the truth value of a given fluent. All other actions leave the fluent invariant.
- An object's colour remains unchanged as a result of picking things up, opening a door, turning on a light, electing a new prime minister of Canada, etc. etc.
- $\sim 2 \times \mathcal{A} \times \mathcal{F}$ frame axioms.
- The *frame problem*:
 - Axiomatizer must think of all these quadratically many frame axioms.
 - System must reason efficiently in the presence of so many axioms.

What Counts as a Solution to the FP?

- Suppose the person responsible for axiomatizing an application domain has succeeded in writing down **all** the effect axioms, i.e. for each relational fluent F and each action A which can cause F 's truth value to change, axioms of the form
$$R(\vec{x}, s) \supset (\neg)F(\vec{x}, do(A, s))$$
and for each functional fluent f and each action A that can cause f 's value to change, axioms of the form:
$$R(\vec{x}, y, s) \supset f(\vec{x}, do(A, s)) = y.$$
- Want a systematic procedure for generating, from these effect axioms, all the frame axioms.
- If possible, also want a *parsimonious* representation for these frame axioms (because in their simplest form, there are too many of them).

Why Do We Want a Solution to the FP?

- Frame axioms are necessary to reason about the domain being formalized.
- Convenience of the axiomatizer.
 - Modularity. The axiomatizer needs only to add new effect axioms.
 - Accuracy. No inadvertent omissions of frame axioms.
- For theorizing about actions.

Determinate Actions Only

- Recall the special syntactic form of effect axioms:

$$R(\vec{x}, s) \supset (\neg)F(\vec{x}, do(A, s)).$$

$$R(\vec{x}, y, s) \supset f(\vec{x}, do(A, s)) = y.$$

- These preclude *indeterminate* actions:

$$heads(do(flip, s)) \vee tails(do(flip, s)),$$

$$(\exists x)holding(x, do(pick_up_a_block, s)).$$

- *Determinate action* = action whose effect axioms all have the syntactic form above for every fluent.
- Indeterminate actions: later.
- Determinate actions: Intuitively, complete information about the initial situation



Complete information about the next situation.

Primitive Actions Only

- We as yet have no constructs for complex actions. All actions are *primitive*.
- Complex actions:
 - if *car_in_driveway* then *drive* else *walk*.
 - clear_table* \triangleq **while** $[(\exists \textit{block}) \textit{ontable}(\textit{block})]$ *remove_a_block*.
 - remove_a_block* \triangleq
 $(\pi \textit{block})[\textit{pickup}(\textit{block}); \textit{put_on_floor}(\textit{block})]$.
- Complex actions: later.

Limitations of This Version of the SC

- No time. So can't talk about how long actions take, when they occur.
- No concurrency.
- Discrete situation. No continuous actions like pushing an object from *A* to *B*.

A SOLUTION TO THE FRAME PROBLEM (SOMETIMES)

- “Sometimes” = determinate actions *without* ramifications (state constraints). Ramifications, later. Indeterminate actions, later.

Frame Axioms: Pednault’s Proposal

- Assume given a set of positive and negative effect axioms (one for each action A and fluent F):

$$\begin{aligned} \varepsilon_F^+(\vec{x}, \vec{y}, s) \supset F(\vec{x}, do(A(\vec{y}), s)), \\ \varepsilon_{\bar{F}}(\vec{x}, \vec{y}, s) \supset \neg F(\vec{x}, do(A(\vec{y}), s)). \end{aligned} \quad (1)$$

Here, $\varepsilon_F^+(\vec{x}, \vec{y}, s)$ and $\varepsilon_{\bar{F}}(\vec{x}, \vec{y}, s)$ are first order formulas whose free variables are among \vec{x}, \vec{y}, s . Notice that in these effect axioms the variable sequences \vec{x} and \vec{y} consist of distinct variables.

- **Completeness Assumption for Fluent Preconditions:**

The fluent precondition $\varepsilon_F^+(\vec{x}, \vec{y}, s)$ specifies all the conditions under which action $A(\vec{y})$, if performed, will lead to the truth of F for \vec{x} in A ’s successor situation. Similarly, $\varepsilon_{\bar{F}}(\vec{x}, \vec{y}, s)$ specifies all the conditions under which action $A(\vec{y})$, if performed, will lead to the falsity of F for \vec{x} in A ’s successor situation.

- By the completeness assumption, reason as follows: Suppose that both $F(\vec{x}, s)$ and $\neg F(\vec{x}, do(A(\vec{y}), s))$ hold. Then F , which was true in situation s , was made false by action A . By the completeness assumption, the only way F could become false is if $\varepsilon_{\bar{F}}(\vec{x}, \vec{y}, s)$ were true.

- This intuition can be expressed axiomatically by:

$$F(\vec{x}, s) \wedge \neg F(\vec{x}, do(A(\vec{y}), s)) \supset \varepsilon_{\bar{F}}(\vec{x}, \vec{y}, s).$$

This is logically equivalent to:

$$F(\vec{x}, s) \wedge \neg \varepsilon_{\bar{F}}(\vec{x}, \vec{y}, s) \supset F(\vec{x}, do(A(\vec{y}), s)).$$

- A symmetric argument yields the axiom:

$$\neg F(\vec{x}, s) \wedge \neg \varepsilon_F^+(\vec{x}, \vec{y}, s) \supset \neg F(\vec{x}, do(A(\vec{y}), s)).$$

- These have precisely the syntactic forms of positive and negative frame axioms and, by virtue of the argument leading to these, they play exactly the role of frame axioms.
- Provided the completeness assumption is true, there is a systematic way of obtaining the frame axioms from the effect axioms.

- **Example:**
Recall,

$$fragile(x) \supset broken(x, do(drop(r, x), s)).$$

- Rewrite this into a logically equivalent form to conform to the pattern (1):

$$x = y \wedge fragile(x) \supset broken(x, do(drop(r, y), s)).$$

- The completeness assumption for this setting is that the only precondition for x being broken as a result of dropping y is that x be fragile and the same as y . This assumption yields the negative frame axiom for the fluent *broken* wrt the action *drop*:

$$\begin{aligned} \neg broken(x, s) \wedge \neg [x = y \wedge fragile(x)] \\ \supset \neg broken(x, do(drop(r, y), s)). \end{aligned}$$

- To obtain the frame axioms in this way for all fluent-action pairs requires considering a large number of “vacuous” effect

axioms. For example, to obtain a frame axiom for the fluent *color* wrt action *drop*, consider the positive effect axiom for *color* wrt *drop*:

$$false \supset color(x, c, do(drop(r, y), s)).$$

From this we obtain the negative frame axiom for *color* wrt *drop*:

$$\neg color(x, c, s) \supset \neg color(x, c, do(drop(r, y), s)).$$

- This illustrates two problems with Pednault’s proposal:

- To systematically determine the frame axioms for all fluent-action pairs from their effect axioms, we must enumerate (or at least consciously consider) all these effect axioms, including the “vacuous” ones. In particular, we must enumerate all fluent-action pairs for which the action has no effect on the fluent’s truth value, which really amounts to enumerating most of the frame axioms directly.
- The number of frame axioms so obtained is $2 \times \mathcal{A} \times \mathcal{F}$, where \mathcal{A} is the number of actions, and \mathcal{F} the number of fluents. Some of these may be vacuously true (i.e., when the fluent precondition of the corresponding effect axiom is *true*), but in general, we are faced with the usual difficulty associated with the frame problem – too many frame axioms.
- Summary: Pednault’s proposal.

- It provides a systematic (and easily and efficiently implementable) mechanism for generating frame axioms from effect axioms.
- But it does not provide a parsimonious representation of the frame axioms.

Frame Axioms: Schubert's Proposal

- Schubert, elaborating on a proposal of Haas, argues in favor of what he calls *explanation closure axioms* for representing the usual frame axioms.
- Consider the fluent *holding*, and suppose that both *holding*(*r*, *x*) and \neg *holding*(*r*, *x*, *do*(*a*, *s*)) are true. How can we explain the fact that *holding* ceases to be true? If we assume that the only way this can happen is if the robot *r* put down or dropped *x*, we can express this with the explanation closure axiom:

$$\begin{aligned} & \text{holding}(r, x, s) \wedge \neg \text{holding}(r, x, \text{do}(a, s)) \\ & \supset a = \text{putdown}(r, x) \vee a = \text{drop}(r, x). \end{aligned}$$

- Notice that this sentence quantifies universally over *a* (actions).
- To see how this functions as a frame axiom, rewrite it in the logically equivalent form:

$$\begin{aligned} & \text{holding}(r, x, s) \wedge a \neq \text{putdown}(r, x) \wedge a \neq \text{drop}(r, x) \\ & \supset \text{holding}(r, x, \text{do}(a, s)). \end{aligned} \tag{2}$$

This says that all actions other than *putdown* and *drop* leave *holding* invariant,¹ which is the standard form of a frame axiom (actually, a set of frame axioms, one for each action distinct from *putdown* and *drop*).

¹To accomplish this, we shall require unique names axioms like *pickup*(*r*, *x*) \neq *drop*(*r'*, *x'*). We shall explicitly introduce these later.

- In general, an *explanation closure axiom* has one of the two forms:

$$F(\vec{x}, s) \wedge \neg F(\vec{x}, do(a, s)) \supset \alpha_F(\vec{x}, a, s),$$

$$\neg F(\vec{x}, s) \wedge F(\vec{x}, do(a, s)) \supset \beta_F(\vec{x}, a, s).$$

- In these, the action variable a is universally quantified. These say that if ever the fluent F changes truth value, then α_F or β_F provides an exhaustive explanation for that change.
- As before, to see how explanation closure axioms function like frame axioms, rewrite them in the logically equivalent form:

$$F(\vec{x}, s) \wedge \neg \alpha_F(\vec{x}, a, s) \supset F(\vec{x}, do(a, s)),$$

$$\neg F(\vec{x}, s) \wedge \neg \beta_F(\vec{x}, a, s) \supset \neg F(\vec{x}, do(a, s)).$$

and

- These have the same syntactic form as frame axioms with the important difference that action a is universally quantified. Whereas there would be $2 \times \mathcal{A} \times \mathcal{F}$ frame axioms, there are just $2 \times \mathcal{F}$ explanation closure axioms. This parsimonious representation is achieved by quantifying over actions in the explanation closure axioms.

- Schubert proposes that explanation closure axioms must be provided independently of the effect axioms. Like the effect axioms, these are domain-dependent. In particular, Schubert argues that they cannot be obtained from the effect axioms by any kind of systematic transformation. Thus, Schubert and Pednault entertain conflicting intuitions about the origins of frame axioms.

- Schubert's appeal to explanation closure as a substitute for frame axioms involves an assumption.

The Explanation Closure Assumption

The only way the fluent F 's truth value could have changed from true to false under action a is if α_F were true. This means, in particular, that α_F completely characterizes all those actions a that can lead to this change; similarly for β_F .

- We can see clearly the need for this assumption from the example explanation closure axiom (2). If, in the intended application, there were an action (say, $eat(r, x)$) that could lead to r no longer holding x , axiom (2) would be false.
- **Summary: Schubert's Proposal**
 - Explanation closure axioms provide a compact representation of frame axioms – $2 \times \mathcal{F}$ of them. (Aside: This assumes they aren't too long. See later for an argument why they are likely to be short.)
 - But Schubert provides no systematic way of automatically generating them from the effect axioms. In fact, he argues this is impossible.
- Can we combine the best of Pednault and Schubert?

A Simple Solution to the FP (Sometimes)

- **Example:** Suppose there are two positive effect axioms for the fluent *broken*:

$$\begin{aligned} fragile(x) &\supset broken(x, do(drop(r, x), s)), \\ nextto(b, x, s) &\supset broken(x, do(explode(b), s)). \end{aligned}$$

These can be rewritten in the logically equivalent form:

$$\begin{aligned} [(\exists r)\{a = drop(r, x) \wedge fragile(x)\} \\ \vee (\exists b)\{a = explode(b) \wedge nextto(b, x, s)\}] &\quad (3) \\ &\supset broken(x, do(a, s)). \end{aligned}$$

- Similarly, consider the negative effect axiom for *broken*:

$$\begin{aligned} \neg broken(x, do(repair(r, x), s)). \end{aligned}$$

In exactly the same way, this can be rewritten as:

$$(\exists r)a = repair(r, x) \supset \neg broken(x, do(a, s)). \quad (4)$$

- Now appeal to the following completeness assumption:
Axiom (3) characterizes all the conditions under which action a leads to y being broken.

- Then if $\neg broken(x, s)$, $broken(x, do(a, s))$ are both true, the truth value of *broken* must have changed because

$$\begin{aligned} (\exists r)\{a = drop(r, x) \wedge fragile(x)\} \\ \vee (\exists b)\{a = explode(b) \wedge nextto(b, x, s)\} \end{aligned}$$

was true.

- This intuition can be formalized, after some logical simplification, by the following explanation closure axiom:

$$\begin{aligned} \neg broken(x, s) \wedge broken(x, do(a, s)) \supset \\ (\exists r)a = drop(r, x) \wedge fragile(x) \vee \\ (\exists b)\{a = explode(b) \wedge nextto(b, x, s)\}. \end{aligned}$$

- Similarly, (4) yields the following explanation closure axiom:

$$\begin{aligned} broken(x, s) \wedge \neg broken(x, do(a, s)) \supset \\ (\exists r)a = repair(r, x). \end{aligned}$$

Aside: Normal Forms for Effect Axioms

- In the previous example, we rewrote one or more positive effect axioms as the single, logically equivalent positive effect axiom with the following syntactic normal form:

$$\gamma_F^+(\vec{x}, a, s) \supset F(\vec{x}, do(a, s)),$$

Similarly, we rewrote one or more negative effect axioms in the normal form:

$$\gamma_F^-(\vec{x}, a, s) \supset \neg F(\vec{x}, do(a, s)).$$

Here, $\gamma_F^+(\vec{x}, a, s)$ and $\gamma_F^-(\vec{x}, a, s)$ are first order formulas whose free variables are among \vec{x}, a, s .

The automatic generation of frame axioms appealed to these normal forms for the effect axioms.

- *Transformation of Effect Axioms to Normal Form:*

1. Each of the given positive effect axioms has the form:

$$\phi_F^+ \supset F(\vec{t}, do(a, s)).$$

Here α is an action term (e.g. $\text{pickup}(x)$, $\text{put}(A,y)$) and the \vec{t} are terms.

Write this in the following, logically equivalent form:

$$a = \alpha \wedge \vec{x} = \vec{t} \wedge \phi_F^+ \supset F(\vec{x}, do(a, s)). \quad (5)$$

Here, $\vec{x} = \vec{t}$ abbreviates $x_1 = t_1 \wedge \dots \wedge x_n = t_n$, and \vec{x} are new variables, distinct from one another and distinct from any occurring in the original effect axiom.

2. Suppose y_1, \dots, y_m are all the variables occurring in the original effect axiom. Then (5) is itself logically equivalent to:

$$(\exists y_1, \dots, y_m)[a = \alpha \wedge \vec{x} = \vec{t} \wedge \phi_F^+] \supset F(\vec{x}, do(a, s)).$$

3. So, each positive effect axiom for fluent F can be written in the logically equivalent form:

$$\Psi_F \supset F(\vec{x}, do(a, s)),$$

where Ψ_F is a formula whose free variables are among \vec{x}, a, s .

Do this for each of the k positive effect axiom for F , to get:

$$\begin{aligned} \Psi_F^{(1)} &\supset F(\vec{x}, do(a, s)), \\ &\vdots \\ \Psi_F^{(k)} &\supset F(\vec{x}, do(a, s)). \end{aligned}$$

4. Write these k sentences as the single, logically equivalent

$$[\Psi_F^{(1)} \vee \dots \vee \Psi_F^{(k)}] \supset F(\vec{x}, do(a, s)).$$

This is the normal form for the positive effect axioms for fluent F .

5. Similarly, compute the normal form for the negative effect axioms for fluent F .

- For those of you who know about semantics for logic programming: The above transformation to normal form as very similar (but not identical) to the preliminary transforming of logic program clauses, in preparation for computing a program's *completion*.

A Simple Solution: The General Case

- *Example:*

Suppose the following are all the positive effect axioms for fluent *tired*:

$$tired(Jack, do(walk(A, B), s)),$$

$$\neg marathonRunner(y) \wedge distance(u, v) > 2km \supset tired(y, do(run(u, v), s)).$$

Their normal form is:

$$\begin{aligned} \{[a = walk(A, B) \wedge x = Jack] \vee [(\exists u, v, y) . a = run(u, v) \wedge \\ \neg marathonRunner(y) \wedge distance(u, v) > 2km \wedge x = y]\} \\ \supset tired(x, do(a, s)). \end{aligned}$$

By properties of equality and existential quantification, this simplifies to:

$$\begin{aligned} \{[a = walk(A, B) \wedge x = Jack] \vee [(\exists u, v) . a = run(u, v) \wedge \\ \neg marathonRunner(x) \wedge distance(u, v) > 2km]\} \\ \supset tired(x, do(a, s)). \end{aligned}$$

- The previous example obviously generalizes. We suppose given, for each fluent F , the following two normal form effect axioms: *Positive Normal Form Effect Axiom for Fluent F*

$$\gamma_F^+(x, a, s) \supset F(x, do(a, s)). \quad (6)$$

Negative Normal Form Effect Axiom for Fluent F

$$\gamma_F^-(x, a, s) \supset \neg F(x, do(a, s)). \quad (7)$$

Here, $\gamma_F^+(x, a, s)$ and $\gamma_F^-(x, a, s)$ are first order formulas whose free variables are among x, a, s .

- *Causal Completeness Assumption:*

Axioms (6) and (7), respectively, characterize all the conditions under which action a can lead to F becoming true (respectively, false) in the successor situation. They are all the causal laws for the fluent F .

- Hence, if F 's truth value changes from *false* to *true* as a result of doing a , then $\gamma_F^+(x, a, s)$ must be *true*; similarly, if F 's truth value changes from *true* to *false*.
- This informally stated assumption can be represented axiomatically by the following:

Explanation Closure Axioms

$$F(x, s) \wedge \neg F(x, do(a, s)) \supset \gamma_F^-(x, a, s), \quad (8)$$

$$\neg F(x, s) \wedge F(x, do(a, s)) \supset \gamma_F^+(x, a, s). \quad (9)$$

- To make this work, we need *Unique Names Axioms* for *Actions*.

For distinct action names A and B ,

$$A(\vec{x}) \neq B(\vec{y}).$$

Identical actions have identical arguments:

$$A(x_1, \dots, x_n) = A(y_1, \dots, y_n) \supset x_1 = y_1 \wedge \dots \wedge x_n = y_n.$$

- **Result 1** Let \mathcal{T} be a *first-order theory* that entails

$$\neg(\exists \vec{x}, a, s). \gamma_F^+(\vec{x}, a, s) \wedge \gamma_F^-(\vec{x}, a, s).$$

Then \mathcal{T} entails that the *general effect axioms (6)* and *(7)*, together with the *explanation closure axioms (8)* and *(9)*, are *logically equivalent* to:

$$F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee F(\vec{x}, s) \wedge \neg \gamma_F^-(\vec{x}, a, s). \quad (10)$$

- The requirement that

$$\neg(\exists \vec{x}, a, s). \gamma_F^+(\vec{x}, a, s) \wedge \gamma_F^-(\vec{x}, a, s)$$

be entailed by the background theory \mathcal{T} simply guarantees the integrity of the effect axioms (6) and (7); under these circumstances, it will be impossible for both $F(\vec{x}, do(a, s))$ and $\neg F(\vec{x}, do(a, s))$ to be simultaneously derived. Notice that by the unique names axioms for actions, this condition is satisfied by the example treating the fluent *broken* above.

- Call formula (10) the *successor state axiom* for fluent F .

- For the above example, the successor state axiom for *broken* is:

$$broken(x, do(a, s)) \equiv$$

$$\begin{aligned} & (\exists r)\{a = drop(r, x) \wedge fragile(x)\} \vee \\ & (\exists b)\{a = explode(b) \wedge neat(b, x, s)\} \vee \\ & broken(x, s) \wedge \neg(\exists r)a = repair(r, x). \end{aligned}$$

A Simple Solution: Summary

2. The assumption that relatively few actions affect a given fluent.
3. The Generalized Completeness Assumption.

- Our proposed solution yields the following axioms:

1. Successor state axioms: for each fluent F ,

$$F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s).$$

2. For each action A , a single action precondition axiom of the form:

$$Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s).$$

3. Unique names axioms for actions.

- Ignoring the unique names axioms (whose effects can be compiled), this axiomatization requires $\mathcal{F} + \mathcal{A}$ axioms in total, compared with the $2 \times \mathcal{A} \times \mathcal{F}$ explicit frame axioms that would otherwise be required.
- But maybe we get fewer axioms at the expense of prohibitively long successor state axioms.
 - A successor state axiom’s length is roughly proportional to the number of actions which affect the truth value of the fluent F .
 - The intuition leading to the frame problem is that most actions don’t affect F . So few actions affect it. So its successor state axiom is short.
- The conciseness and perspicuity of this axiomatization relies on three things:
 1. Quantification over actions.

Deductive Planning with the Situation Calculus

- Planning: find a sequence of actions which, if performed in a world with an axiomatized initial situation, will lead to a world situation in which some goal statement will be true.

- **Example:**

Action precondition axioms:

$$Poss(pickup(r, x), s) \equiv robot(r) \wedge [(\forall z)\neg holding(r, z, s)] \wedge nexto(r, x, s), \quad (11)$$

$$Poss(walk(r, y), s) \equiv robot(r). \quad (12)$$

$$Poss(drop(r, x), s) \equiv robot(r) \wedge holding(r, x, s). \quad (13)$$

Effect axioms:

$$holding(r, x, do(pickup(r, x), s)),$$

$$\neg holding(r, x, do(drop(r, x), s)),$$

$$nexto(r, y, do(walk(r, y), s)),$$

$$nexto(r, y, s) \supset nexto(x, y, do(drop(r, x), s)),$$

$$y \neq x \supset \neg nexto(r, x, do(walk(r, y), s)),$$

$$on_floor(x, do(drop(r, x), s)),$$

$$\neg on_floor(x, do(pickup(r, x), s)).$$

Solve the frame problem:

Successor state axioms:

$$holding(r, x, do(a, s)) \equiv a = pickup(r, x) \vee holding(r, x, s) \wedge a \neq drop(r, x), \quad (14)$$

$$nexto(x, y, do(a, s)) \equiv a = walk(x, y) \vee$$

$$(\exists r)[nexto(r, y, s) \wedge a = drop(r, x)] \vee$$

$$nexto(x, y, s) \wedge \neg(\exists z)[a = walk(x, z) \wedge z \neq y], \quad (15)$$

$$on_floor(x, do(a, s)) \equiv (\exists r)a = drop(r, x) \vee$$

$$on_floor(x, s) \wedge \neg(\exists r)a = pickup(r, x). \quad (16)$$

Unique names axioms for actions:

$$pickup(r, x) \neq drop(r', y),$$

$$pickup(r, x) \neq walk(r', y),$$

$$walk(r, y) = walk(r', y') \supset r = r' \wedge y = y',$$

etc.

Initial situation:

$$chair(C), robot(R), nexto(R, A, S_0), \quad (17)$$

$$(\forall z)\neg holding(R, z, S_0).$$

Deductive Planning (Continued)

- Some derived facts:

$$holding(R, A, do(pickup(R, A), S_0)). \quad (18)$$

From (17), (18), (14) and unique names for actions,

$$holding(R, A, do(walk(R, y), do(pickup(R, A), S_0))). \quad (19)$$

From (17), (19) and (15),

$$nexto(A, y, do(drop(R, A), do(walk(R, y), do(pickup(R, A), S_0)))). \quad (20)$$

From (17), (16) and (19),

$$onfloor(A, do(drop(R, A), do(walk(R, y), do(pickup(R, A), S_0)))). \quad (21)$$

- Suppose we want to derive:

$$(\exists s).nexto(A, B, s) \wedge onfloor(A, s).$$

i.e. there is a world situation in which A is next to B and A is on the floor.

- The above is a constructive proof of this sentence, with

$$s = do(drop(R, A), do(walk(R, B), do(pickup(R, A), S_0))).$$

- A plan to get A onto the floor next to B .

Deductive Planning (Continued)

- Key idea: Plan synthesis as a side effect of theorem proving (Green, 1969, early versions of Shakey the SRI robot).

$$Axioms \models (\exists s)G(s).$$

- Any binding for s as a side effect of a proof is a plan guaranteed to yield a world situation satisfying the goal G .
- c.f. Prolog.
- Slight problem: Can also prove

$$(\exists s).nexto(A, B, s) \wedge onfloor(A, s),$$

with

$$s = do(drop(C, A), do(walk(C, B), do(pickup(C, A), S_0))).$$

- The first plan, in which the robot R does the work, conforms to the action precondition axioms, while the second plan does not; according to these axioms, robots can pick things up, and go for walks, but chairs cannot.
- The robot's plan is *executable* according to the action precondition axioms, meaning that one can prove, from the axioms, that:

$$Poss(pickup(R, A), S_0) \wedge$$

$$Poss(walk(R, B), do(pickup(R, A), S_0)) \wedge$$

$$Poss(drop(R, A), do(walk(R, B), do(pickup(R, A), S_0))).$$

There is no proof that this sequence of actions, as performed by the chair, is executable.

- **Official characterization of planning in the sitcalc:**

Relative to some background axioms, establish that

$$Axioms \vdash (\exists s).executable(s) \wedge G(s).$$

Motivation and Background

- Databases evolve over time as a result of *transactions*, whose purpose is to update the database with new information.
- Example: An educational database might have a transaction specifically designed to change a student's grade.
 - This would normally be a procedure which, when invoked on a specific student and grade, first checks that the database satisfies certain preconditions (e.g., that there is a record for the student, and that the new grade differs from the old), and if so, records the new grade.
- This is a *procedural* notion. Transactions also *physically modify* the database.
- Ideally, we want a *specification* of the entire evolution of a database.
- This section proposes such a specification of database transactions by appealing to various ideas from the AI planning literature:
 - The situation calculus.
 - The frame problem.
 - The projection problem in planning.
 - Goal regression for plan synthesis.
- Theory of database updates \cong AI planning in the situation calculus.

Database Updates: A Proposal

- Represent databases in the situation calculus. Updatable relations are fluents, i.e. they take a situation argument.
- Treat update transactions exactly like actions in the AI planning domain. Transactions are functions.
- For this to work, need a solution to the frame problem.

The Basic Approach: An Example

An Education Database

- **Relations**

1. $enrolled(st, course, s)$: st is enrolled in $course$ when the database is in situation s .
2. $grade(st, course, grade, s)$: The grade of st in $course$ is $grade$ when the database is in situation s .
3. $prerequ(pre, course)$: pre is a prerequisite course for $course$.

- **Initial Database State**

These will be arbitrary first order sentences, the only restriction being that fluents mention only the initial situation S_0 .

$$enrolled(Sue, C100, S_0) \vee enrolled(Sue, C200, S_0), \\ (\exists c)enrolled(Bill, c, S_0),$$

$$(\forall p).prerequ(p, P300) \equiv p = P100 \vee p = M100, \\ (\forall p)\neg prerequ(p, C100),$$

$$(\forall c).enrolled(Bill, c, S_0) \equiv c = M100 \vee c = C100 \vee c = P200, \\ enrolled(Mary, C100, S_0), \neg enrolled(John, M200, S_0), \dots \\ grade(Sue, P300, 75, S_0), \quad grade(Bill, M200, 70, S_0), \dots$$

Example: Continued

- **Database Transactions**

- Denote by function symbols.
 - Treat exactly like actions in situation calculus planning.
- For the example, there are three transactions:
 1. $register(st, c)$,
 2. $change(st, c, g)$,
 3. $drop(st, c)$.
 - A student can register in a course iff she has obtained a grade of at least 50 in all prerequisites for the course:

$$Poss(register(st, c), s) \equiv \{(\forall p).prerequ(p, c) \\ \supset (\exists g).grade(st, p, g, s) \wedge g \geq 50\}.$$

- It is possible to change a student's grade iff he has a grade which is different than the new grade:

$$Poss(change(st, c, g), s) \equiv (\exists g').grade(st, c, g', s) \wedge g' \neq g.$$

- A student may drop a course iff the student is currently enrolled in that course:

$$Poss(drop(st, c), s) \equiv enrolled(st, c, s).$$

Example: Continued

- **Transaction Effect Axioms:**

$$\begin{aligned} &\neg \text{enrolled}(st, c, \text{do}(\text{drop}(st, c), s)), \\ &\text{enrolled}(st, c, \text{do}(\text{register}(st, c), s)), \\ &\text{grade}(st, c, g, \text{do}(\text{change}(st, c, g), s)), \\ &g' \neq g \supset \neg \text{grade}(st, c, g', \text{do}(\text{change}(st, c, g), s)). \end{aligned}$$

Solve the frame problem.

- **Successor state axioms:**

$$\begin{aligned} \text{enrolled}(st, c, \text{do}(a, s)) &\equiv a = \text{register}(st, c) \vee \\ &\quad \text{enrolled}(st, c, s) \wedge a \neq \text{drop}(st, c), \\ \text{grade}(st, c, g, \text{do}(a, s)) &\equiv a = \text{change}(st, c, g) \vee \\ &\quad \text{grade}(st, c, g, s) \wedge (\forall g') a \neq \text{change}(st, c, g'). \end{aligned}$$

Queries

- All updates are *virtual*; the database is never physically changed.
- Querying the database resulting from a sequence of update transactions:
“Is John enrolled in any courses after the transaction sequence $\text{drop}(\text{John}, C100), \text{register}(\text{Mary}, C100)$ has been ‘executed’?”
 $\text{Database} \models (\exists c).\text{enrolled}(\text{John}, c, \text{do}(\text{register}(\text{Mary}, C100), \text{do}(\text{drop}(\text{John}, C100)$
 $\text{ed' ?}''$
- This is the *projection problem* in AI planning. More later.
- Such a sequence of update transactions is called a database *log* in database theory.
- Query evaluation wrt a database log: later.