Natural actions, concurrency and time in situation calculus

Based on the book “ knowledge in action: Logical foundations for specifying and implementing dynamical systems.”
by
Raymond Reiter.

April, 9 2004

Talk Outline

$\Rightarrow$ **Introduction to situation calculus**

- Syntax
- Foundational axioms
- Axiomatizing actions

$\Rightarrow$ **Time, concurrency and natural actions**

- Concurrency
- Sequential temporal situation calculus
- Concurrent non-temporal situation calculus
- Concurrent temporal situation calculus
- Natural actions

$\Rightarrow$ **Comments**

History

- Situation calculus was first introduced in 1963 by John McCarthy as a way of logically specifying dynamical systems in AI.
- A dynamical system that we want to control, simulate, analyze, can be axiomatized in situation calculus.
- Through logical entailment, all else will follow, including system control, simulation, and analysis.
- Once we get the logical specification right, the axioms are translated into Prolog code.

Definitions

- Situation calculus is a second-order language specifically designed for representing dynamically changing worlds.
- All changes in the world are result of named *actions*.
- A possible world history, which is simply a sequence of actions is represented by a first order term called *situation*.

Syntax

- The language $L_{sitcalc}$ has three disjoint sorts: *action* for actions, *situation* for situations, and *objects* for everything else that depends on the domain.
- Full set of connectives and quantifiers - $\wedge, \neg, \exists, \vee$.
- Countably infinite individual variable symbols of each sort. For variables of sort situation and action, *s* and *a* (with superscripts and subscripts) are used respectively.

Syntax (contd)

- Two function symbols of sort situation
  - Constant symbol,S_0 denoting the initial situation is an empty sequence of actions.
  - Binary function symbol,
    do: action $\times$ situation $\rightarrow$ situation.
    do(a,s) denotes the successor situation to s resulting from performing action a.
- Binary Predicate symbol $\sqsubset$: situation $\times$ situation.
  This defines an ordering relation on situations.
  $s \sqsubset s^{'}$ means that s is proper subsequence of $s^{'}$.

Syntax (contd)

- Binary Predicate symbol *poss:* action $\times$ situation. poss(a,s) denotes that it is possible to perform action a in situation s.
- Situation independent relations:
  For n$\geq$0, finite or countably infinite number of predicate symbols with arity n of sort $(action \cup object)^n$
  Example:
  human(joe), oddnumber(n), movingAction(run(L1,L2)).

Syntax (contd)

- Situation independent functions:
  For n$\geq$0, finite or countably infinite number of function symbols with arity n of sort
  $(action \ \cup \ object)^n \rightarrow object$
  Example:
  height(mount_everest), cost(pickup(person,object)).
- Action functions:
  For n$\geq$0, finite or countably infinite number of function symbols of sort $(action \cup object)^n \rightarrow action$.
  Example:
  pickup(X), move(A,B).

Syntax (contd)

- Relational fluents are relations whose truth values vary from situation to situation. For n$\geq$0, finite or countably infinite number of predicate symbols with arity n+1 and sorts $(action \cup object)^n \times situation$.
  Example : ontable(x,s) , husband(x,y,s).
- Functional fluents are functions whose values vary from situation to situation. For n$\geq$0, finite or countably infinite number of function symbols with arity n+1 and sorts $(action \cup object)^n \times situation \rightarrow action \cup object$.
  Example: age(mary,s), water_level(tank,s).

Foundational Axioms for situation calculus

The following axioms help us capture that situations are finite sequences of actions, and a certain sequence of actions precedes another.

- Unique names axiom for situations.
  $do(a_1, s_1) = do(a_2, s_2) \supset a_1 = a_2 \wedge s_1 = s_2$.
- Second order induction axiom.
  $(\forall P).P(S_0) \wedge (\forall a, s)[P(s) \supset P(do(a, s))] \supset (\forall s)\ P(s)$.
- $\neg s \sqsubset S_0$.
- $s \sqsubset do(a, s') \equiv s \sqsubseteq s'$.

All these axioms are domain independent and represented by Σ.

Axiomatizing actions in situation calculus

To represent the effects of actions the following axioms are proposed.

- Action precondition axioms
- Successor state Axioms
- Unique Names axioms for actions
  For distinct action names A and B,
  A($\vec{x}$) $\neq B(\vec{y})$.
  Identical actions have identical arguments:
  $A(x_1, ..., x_n) = A(y_1, ..., y_n) \supset x_1 = y_1 \wedge ... \wedge x_n = y_n$

Action precondition axioms

- These axioms specify conditions that must be satisfied in order for an action to be executed in any situation.
- For each action $A(\vec{x})$ there is an axiom
  $poss(A(\vec{x}), s) \equiv \pi_A(\vec{x}, s)$.
  $\pi_A(\vec{x}, s)$ is a first order formula with free variables $\vec{x}, s$.
  Example:
  $poss(pickup(r, x), s) \equiv [(\forall z)\neg holding(r, z, s)]$
  $\wedge \neg heavy(x) \wedge nextTo(r, x, s)$.

Successor state axioms

- These axioms define the value of a relational or functional fluent in the successor situation resulting from performing an action in the current situation.
- This axiom is logically constructed from the following axioms:
  - Effect Axioms
  - Explanation closure axioms

Effect Axioms

- These axioms specify how actions effect the values of fluents.
- Positive Normal form effect axiom for relational fluent F
  $\gamma_F^+(\vec{x}, a, s) \supset F(\vec{x}, do(a, s))$
- Negative Normal form effect axiom for relational fluent F
  $\gamma_F^-(\vec{x}, a, s) \supset \neg F(\vec{x}, do(a, s))$
- Effect axioms for functional fluents
  $\gamma_f(\vec{x}, y, a, s) \supset f(\vec{x}, do(a, s)) = y$

Examples :

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

$(\exists b')[a = move(b, b') \wedge y = height(b', s) + 1]$

$\supset height(b, do(a, s)) = y.$

Explanation closure axiom

This axiom captures the assumption that if F's truth value changes from false in current situation s to true in the next situation do(a,s) resulting from doing a then $\gamma_F^+(\vec{x}, a, s)$ must have been true. Similarly F's truth value from true to false.

- $F(\vec{x}, s) \wedge \neg F(\vec{x}, do(a, s)) \supset \gamma_F^-(\vec{x}, a, s)$
- $\neg F(\vec{x}, s) \wedge F(\vec{x}, do(a, s)) \supset \gamma_F^+(\vec{x}, a, s)$
- For functional fluents,
  $f(\vec{x}, do(a, s)) \neq f(\vec{x}, s) \supset \exists y \; \gamma_f(\vec{x}, y, a, s)$

Example:
$broken(x, s) \wedge \neg broken(x, do(a, s)) \supset$
$(\exists r) a = repair(r, x).$

Successor state axiom for fluent F

Suppose that T is 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 T entails that - effect axioms along with explanation closure axioms are logically equivalent to

- $F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s)$
  This is the successor state axiom for relational fluent F.
- For Functional fluents,
  $f(\vec{x}, do(a, s)) = y \equiv \gamma_f(\vec{x}, y, a, s) \vee y = f(\vec{x}, s) \wedge \neg(\exists y^{'})\gamma_f(\vec{x}, y^{'}, a, s)$

Example:
$broken(x, do(a, s)) \equiv (\exists r)(a = drop(r, x)) \wedge fragile(x, s) \vee broken(x, s) \wedge \neg(\exists r)a = repair(r, x)$

Executable situation

- Executable situations are those action histories in which it is actually possible to perform the actions one after the other.
- This definition allows us to know whether each action leading upto the current situation does not violate its precondition.

$$executable(s) \overset{def}{=} (\forall a, s^*).do(a, s^*) \sqsubseteq s \supset poss(a, s^*).$$

Introducing concurrency and continuous time in situation calculus

We now expand Situation calculus to include actions that occur in continuous time and also actions that occur simultaneously.

Dealing with concurrency

- A concurrent action is a collection of simple actions that occur at the same time.
- Modeling the possibility of concurrent action execution involves many formal and conceptual problems.
- If actions have duration, then what is a concurrent action
  - Do all actions have the same duration ?.
  - Is it that only their time segments overlap ?.

Overcoming problems

- Actions with duration are conceived as processes represented by relational fluents and
- Durationless (instantaneous) actions that initiate and terminate these processes are introduced.

Example:
The action walk(x,y) can be replaced by instantaneous actions startWalk(x,y) and endWalk(x,y) and process of walking from x to y by walking(x,y,s).

- With this device of instantaneous start and end actions, arbitrarily complex concurrency can be represented.

Example:
{startWalk(A,B), startChewgum}, {endChew,startSinging}, {endWalk(A,B)}

Sequential Temporal Situation Calculus

- Sequential temporal situation calculus is a result of adding explicit representation of time to sequential situation calculus.
- This addition helps us specify the exact times or range of times at which actions must occur.
- This is achieved by adding new temporal argument to all instantaneous actions.

Example:
startPushing(x,t) is the instantaneous action of pushing object x at time t.

Foundational Axioms for Sequential Temporal Situation calculus

- The foundational axioms are obtained by adding a new axiom to the existing foundational axioms of sequential situation calculus.
- To define this new axiom the function symbol, time: *action* → *reals* is introduced.
  time(a) denotes the time of occurrence of action a.
- Therefore, for every action $A(\vec{x}, t)$, $time(A(\vec{x}, t)) = t$, is the axiom for specifying time of occurrence of action A.

Foundational Axioms contd..

- A second function symbol, start: *situation* $\rightarrow$ *reals* is introduced. start(s) denotes the start time of situation s.
- Therefore, start(do(a,s)) = time(a) $\diamond$ denotes that the start time of a situation resulting from performing a in situation s is the time of occurrence of action a.
- $\diamond$ along with Σ are foundational axioms of sequential temporal situation calculus.

Executable Situation

- Since time is introduced , there is an addition to the definition of executable situation.
- This new definition ensures that the actions in the sequence do not have decreasing times of occurrences.
- $executable(s) \overset{def}{=} (\forall a, s^*).do(a, s^*) \sqsubseteq s \supset poss(a, s^*) \wedge start(s^*) \leq time(a)$
- $start(s^*) \leq time(a)$ permits action sequences in which time of an action is same as time of preceding action. This condition is useful when we have enabling actions.
  Example:
  do(startFalling(t), do(cutstring(t),S_0)).

Concurrent,Non-Temporal Situation Calculus

- The focus is on representing concurrent actions in situation calculus ignoring time for the moment.
- A concurrent action is a set of possibly infinite number of simple actions all having equal but unspecified duration.
- The variables $c, c^{'}$,... are used to represent concurrent actions.
- The notation a ϵ c means that simple action a is one of the actions of concurrent action c.
- A situation represents sequence of concurrent actions.

Foundational Axioms for concurrent non-temporal situation calculus

The first four are similar to those of sequential non-temporal situation calculus.The *a* is replaced by *c* here.

1. $do(c_1, s_1) = do(c_2, s_2) \supset c_1 = c_2 \wedge s_1 = s_2.$

2. Second order induction axiom.
   $(\forall P).P(S_0) \wedge (\forall c, s)[P(s) \supset P(do(c, s))] \supset (\forall s)\ P(s).$

3. $\neg s \sqsubset S_0.$

4. $s \sqsubset do(c, s') \equiv s \sqsubseteq s'.$

Foundational axioms contd..

5. In addition we have two more axioms,

$poss(a, s) \supset poss(\{a\}, s).$

6. $poss(c, s) \supset (\exists a) a \epsilon c \wedge (\forall a)[a \epsilon c \supset poss(a, s)].$

This axiom implies that a concurrent action is possible if it contains atleast one action and all its simple actions are possible.

Precondition Interaction problem

- The converse of 6 does not hold and therefore it is not biconditional.
  Example:
  $poss(startMoveLeft, s) \equiv \neg movingLeft(s).$
  $poss(startMoveRight, s) \equiv \neg movingRight(s).$
  But, $poss(\{startMoveLeft, startMoveRight\}, s)$ is false.
- Therefore two actions may each be possible, action preconditions jointly consistent, but they cannot occur concurrently.
- This problem is also called *Precondition interaction problem*.

Some more definitions

- The definition of executable situation is now extended for concurrent actions.
  $executable(s) \overset{def}{=} (\forall c, s^*).do(c, s^*) \sqsubseteq s \supset poss(c, s^*)$
- Consider two people shooting at each other. *shoot(x,y)* is the instantaneous action of person x shooting at person y.
  The following successor state axiom:
  $$\begin{aligned} dead(x, do(c, s)) \equiv\ & (\exists y)a = shoot(y, x) \epsilon\ c \\ & \vee\ dead(x, s). \end{aligned}$$
  entails the following:
  $dead(Tom, do(\{shoot(Tom, Harry),$
  $shoot(Harry, Tom)\}, S_0)).$

  $dead(Harry, do(\{shoot(Tom, Harry),$
  $shoot(Harry, Tom)\}, S_0))$

Concurrent Temporal situation calculus

- The focus is to accomodate time into concurrent non-temporal situation calculus
- The axiom for defining time of occurrence of an action is used again here.
  $time(A(\vec{x}, t)) = t.$
- Some new definitions are introduced.
  $$coherent(c) \overset{def}{=} (\exists a) a \epsilon c \wedge (\exists t) \forall a' [a' \epsilon c \supset time(a') = t]$$
  A coherent concurrent action is one in which there is atleast one action in the collection, and for which all of the instantaneous actions in the collection occur at the same time.

Foundational axioms for concurrent temporal situation calculus

1. The time for concurrent action is defined as
$$coherent(c) \supset [time(c) = t \equiv (\exists a)(a \epsilon c \wedge time(a) = t)]$$

2. The start time of a situation is defined as
$start(do(c, s)) = time(c)$

3. $poss(a, s) \supset poss(\{a\}, s)$

4. $poss(c, s) \supset coherent(c) \wedge (\forall a)[a \epsilon c \supset poss(a, s)]$.
   The converse of 4 does not hold.

The above four axioms along with foundational axioms 1 through 4 for concurrent non-temporal situation calculus are the foundational axioms for concurrent temporal situation calculus.

Executable situation

- Executable situation is defined as follows:
  $executable(s) \overset{def}{=} (\forall c, s^*).do(c, s^*) \sqsubseteq s \supset poss(c, s^*) \wedge start(s^*) \leq time(c)$
- The definition means that all concurrent actions in s are possible and times of action occurrences are non-decreasing.

Natural Actions

- Natural actions occur in response to known laws of physics like a ball bouncing at times predicted by Newtonian laws of motion.
- Fundamental property : They must occur at their predicted times, provided no earlier action(agent or natural) prevents them from occurring.
  Example: A catch action before the bounce prevents the bounce action.

What is needed ?

- Since several natural actions may occur simultaneously, a theory of concurrency is needed.
- Actions are modeled by equations of motion, therefore continuous time must be represented.
- Since concurrent temporal situation calculus has these properties, it provides the foundations for representing natural actions.

Representing physical laws

- Laws of physics are embodied in action precondition axioms.
  Example:
  $poss(bounce(t), s) \equiv isFalling(s) \wedge$
  $\{height(s) + vel(s)[t - start(s)] -$
  $\frac{1}{2}G[t - start(s)]^2 = 0\}$.
- height(s) and vel(s) are the height and velocity of the ball at start of situation s.
- poss(bounce(t),s) means that bounce is physically possible at time t during situation s.

Properties of natural actions

- Nature does not have free will to withhold her actions; if time and circumstances are right, the action must occur.
- Natural World Assumption.
  $\forall a\ natural(a).$
  restricts the domain of discourse of actions to natural actions only.
- $executable(do(c, s)) \wedge executable(do(c^{'}, s)) \wedge NWA \supset c = c^{'}$
  This lemma tells us that natural worlds are deterministic: if there is a executable successor situation, it is unique.

Least Natural Time Points

- Least natural time point is the earliest time during situation s at which a natural action can occur.
- This definition plays a central role in theorizing about natural actions.
- $lntp(s,t) \overset{def}{=} (\exists a)[natural(a) \land poss(a,s) \land time(a) = t] \land (\forall a)[natural(a) \land poss(a,s) \supset time(a) \geq t].$
- Least natural time point is unique.
  $lntp(s,t) \land lntp(s,t') \supset t = t'.$

Least Natural Time Point Condition

- In some cases, least natural time point may not exist
  Example:
  $(\forall a).natural(a) \equiv (\exists x, t)a = B(x, t)$,
  where x is non-zero natural number
  $poss(B(x, t), s) \equiv t = start(s) + 1/x$.
- The following sentence called LNTPC is introduced for such cases:
  $(\forall s).(\exists a)[natural(a) \wedge poss(a, s)] \supset (\exists t)lntp(s, t)$.

Why is LNTPC important?

- When LNTPC holds, the following theorem provides a complete characterization of executable situtations.
- Theorem: The foundational axioms for the concurrent, temporal situation calculus together with definitions of LNTPC and NWA entail:

$$
\begin{aligned}
& LNTPC \wedge NWA \supset \\
& executable(do(c,s)) \equiv \ \{executable(s) \wedge poss(c,s) \\
& \qquad \wedge start(s) \leq time(c) \wedge \\
& \qquad (\forall a)[a \epsilon c \equiv poss(a,s) \\
& \qquad \wedge lntp(s, time(a))]\}
\end{aligned}
$$

Example of a Natural world

- Consider two elastic balls B_1 and B_2 rolling along x-axis, between two walls W_1 and W_2 that are parallel to y-axis.
- The balls bounce indefinitely between the walls and occasionally collide with each other.
- The bounces and collisions change the velocities of the ball.
- Let wallLocation(w) denote the distance of wall w from y-axis.

Primitive actions and their precondition axioms

- $collide(b_1, b_2, t)$.
  $Balls\ b_1\ and\ b_2\ collide\ at\ time\ t.$
- $bounce(b, w, t)$.
  $Ball\ b\ bounces\ against\ wall\ w\ at\ time\ t.$

- $poss(collide(b_1, b_2, t), s) \equiv vel(b_1, s) \neq vel(b_2, s) \wedge t > start(s) \wedge t = start(s) - \frac{pos(b_1,s) - pos(b_2,s)}{vel(b_1,s) - vel(b_2,s)}$
- $poss(bounce(b, w, t), s) \equiv vel(b, s) \neq 0 \wedge t > start(s) \wedge t = start(s) + \frac{wallLocation(w) - pos(b,s)}{vel(b,s)}$

Fluents and their successor state axioms

- pos(b,s). A functional fluent denoting the position of ball b in situation s.
  $pos(b, do(c, s)) = pos(b, s) + vel(b, s) * (time(c) - start(s)).$
- vel(b,s). A functional fluent denoting the velocity of ball b in situation s.
  $vel(b, do(c, s)) = v \equiv$
  $(\exists w, t) bounce(b, w, t) \epsilon c \ \wedge v = -vel(b, s) \vee$
  $\neg(\exists w, t) bounce(b, w, t) \epsilon c \wedge (\exists b^{'}, t)[v = vel(b^{'}, s) \wedge$
  $(collide(b, b^{'}, t) \epsilon c \vee collide(b^{'}, b, t) \epsilon c)] \vee$
  $v = vel(b, s) \ \wedge \ \neg(\exists b^{'}, t)[collide(b, b^{'}, t) \ \epsilon \ c \vee$
  $collide(b, b^{'}, t) \epsilon c] \wedge \neg(\exists w, t) bounce(b, w, t) \epsilon c.$

Initial situation

- $B_1 \neq B_2, W_1 \neq W_2,$
- $pos(B_1, S_0) = 0, pos(B_2, S_0) = 120,$
- $vel(B_1, S_0) = 10, vel(B_2, S_0) = -5,$
- $wallLocation(W_1) = 0, wallLocation(W_2) = 120.$
- Domain closure axiom for natural actions:
$$\begin{aligned} natural(a) \equiv\ & (\exists b_1, b_2, t)[b_1 = B_1 \wedge b_2 = B_2 \\ & \wedge a = collide(b_1, b_2, t)] \vee \\ & (\exists b, w, t)[(b = B_1 \vee b = B_2) \\ & \wedge (w = W_1 \vee w = W_2) \wedge \\ & a = bounce(b, w, t)]. \end{aligned}$$

Results

The following is output received under ECLIPSE Prolog for simulating the first 10 concurrent actions for the previous natural world example.

- $[[collide(b1, b2, 8.0)], [bounce(b2, w2, 12.0)],$
  $[bounce(b1, w1, 24.0), bounce(b2, w1, 24.0),$
  $collide(b1, b2, 24.0)],$
  $[bounce(b2, w2, 36.0)], [collide(b1, b2, 40.0)],$
  $[bounce(b1, w1, 48.0), bounce(b2, w2, 48.0)],$
  $[collide(b1, b2, 56.0)], [bounce(b2, w2, 60.0)],$
  $[bounce(b1, w1, 72.0), bounce(b2, w1, 72.0),$
  $collide(b1, b2, 72.0)]$
  $[bounce(b2, w2, 84.0)]]$

Comments

- Unlike actions, the time argument is not added to the fluents. Therefore, the values of fluents are known only at the start of a situation.
- In some Dynamical systems, the values of functions such as temperature must be monitored continuously with time because they may trigger some events. The approach of situation calculus will not be useful to represent such functions.
- Situation calculus does not encourage the use of state constraints, therefore the language is less expressive.
- The axioms of situation calculus are applicable only for deterministic actions.