CHAPTER

Concurrent Events

8

The event calculus allows several events to occur at the same time, because we may have event occurrences $Happens(\alpha_1, \tau_1)$ and $Happens(\alpha_2, \tau_2)$ such that $\alpha_1 \neq \alpha_2$ and $\tau_1 = \tau_2$. In this case, we say that the event occurrences are concurrent or simultaneous. In this chapter, we first consider some concurrent event occurrences that we might want to prohibit and methods for prohibiting them. Then, we discuss concurrent event occurrences that have cumulative and canceling effects and how to handle them.

8.1 RESTRICTING CONCURRENCY

The commonsense world places certain restrictions on what events can occur simultaneously. This section presents two methods for representing these restrictions in the event calculus: (1) state constraints and (2) event occurrence constraints.

8.1.1 STATE CONSTRAINTS

The first method of restricting concurrency is to use state constraints. This method relies on the fact that, if the effects of two events are inconsistent with a state constraint, then those events cannot occur concurrently. Consider an agent that can walk from one room to another. We have the effect axioms:

$$Initiates(Walk(a, r_1, r_2), InRoom(a, r_2), t)$$

$$r_1 \neq r_2 \Rightarrow Terminates(Walk(a, r_1, r_2), InRoom(a, r_1), t)$$

Now, a commonsense fact is that an agent cannot walk into one room at the same time that the agent walks into some other room. This can be represented using a state constraint that says that an agent can be in one room at a time:

$$\begin{gathered} HoldsAt(InRoom(a, r_1), t) \wedge \\ HoldsAt(InRoom(a, r_2), t) \Rightarrow \\ r_1 = r_2 \end{gathered} \tag{8.1}$$

Given these axioms, we cannot, say, have both Nathan walking into the kitchen at timepoint 0:

$$Happens(Walk(Nathan, Hallway, Kitchen), 0)$$

and Nathan walking into the living room at timepoint 0:

$$Happens(Walk(Nathan, Hallway, LivingRoom), 0)$$

If we did, we would get inconsistency. We could show

$$HoldsAt(InRoom(Nathan, Kitchen), 1) \wedge$$
$$HoldsAt(InRoom(Nathan, LivingRoom), 1)$$

which contradicts (8.1).

Similarly, the fact that two agents cannot simultaneously pick up the same object can be represented using a state constraint that says that an object can be held by one agent at a time:

$$HoldsAt(Holding(a_1, o), t) \wedge HoldsAt(Holding(a_2, o), t) \Rightarrow$$
$$a_1 = a_2$$

8.1.2 EVENT OCCURRENCE CONSTRAINTS

The second method of restricting concurrency is to use *event occurrence constraints*, or formulas involving two or more atoms of the form $Happens(\alpha, \tau)$, where α is an event term and τ is a timepoint term. We now describe several common idioms.

Totally ordered plans

When solving abduction or planning problems, we may wish to insist that the plan be a *totally ordered plan*, or simple sequence of event occurrences. We can do this with an event occurrence constraint that says that at most one event can occur at a time:

$$Happens(e_1, t) \wedge Happens(e_2, t) \Rightarrow e_1 = e_2$$

Different conflicting actions of two agents

It may be desirable to restrict concurrency for different but conflicting actions of two agents. Consider the following effect axioms dealing with moving a jar and putting a penny in a jar:

$$Initiates(Move(a, j, l_1, l_2), At(j, l_2), t)$$

$$l_1 \neq l_2 \Rightarrow Terminates(Move(a, j, l_1, l_2), At(j, l_1), t)$$

$$HoldsAt(At(p, l), t) \wedge HoldsAt(At(j, l), t) \Rightarrow$$
$$Initiates(PutIn(a, p, j, l), In(p, j), t)$$

We may wish to state that an agent cannot place a penny in a jar at the same time that another agent moves the jar, even though this is permitted by these axioms. We can do this by adding the axiom

$$Happens(Move(a_1, j, l_1, l_2), t) \wedge a_1 \neq a_2 \Rightarrow$$
$$\neg Happens(PutIn(a_2, p, j, l_1), t)$$

Similar conflicting actions of two agents

It is sometimes desirable to restrict concurrency for similar actions of two agents. Suppose we have an effect axiom that says that, if a piece of food is not eaten and an agent eats the piece of food, then the piece of food will be eaten:

$$\neg HoldsAt(Eaten(p), t) \Rightarrow$$
$$Initiates(Eat(a, p), Eaten(p), t)$$

Nothing will prevent two agents from simultaneously eating the same piece of food. To prevent this, we may add the axiom

$$Happens(Eat(a_1, p), t) \wedge a_1 \neq a_2 \Rightarrow \neg Happens(Eat(a_2, p), t)$$

Different conflicting actions of one agent

We may wish to restrict concurrency for different but conflicting actions within a single agent. An agent may not be able to pat the agent's head and rub the agent's stomach at the same time:

$$Happens(PatHead(a), t) \Rightarrow \neg Happens(RubStomach(a), t)$$

Similar conflicting actions of one agent

We may also wish to restrict concurrency for similar actions within a single agent. Even if an agent can be holding several objects at the same time, the agent may not be able to pick up two objects simultaneously:

$$Happens(PickUp(a, o_1), t) \wedge Happens(PickUp(a, o_2), t) \Rightarrow o_1 = o_2$$

8.1.3 DISCUSSION

In order to recognize situations that might require restrictions on concurrency, we look for actions that use, consume, or provide the same resource at the same time.

Notice that EC and DEC disallow concurrent event occurrences with contradictory effects. If a fluent is simultaneously initiated and terminated by an occurring event, we get inconsistency. Consider the effect axioms

$$Initiates(Break(a, d), Broken(d), t)$$
$$Terminates(Fix(a, d), Broken(d), t)$$

and the event occurrence formulas

$$Happens(Break(Nathan, TV), 1)$$
$$Happens(Fix(Nathan, TV), 1)$$

From these we have both $HoldsAt(Broken(TV), 2)$ and $\neg HoldsAt(Broken(TV), 2)$.

8.2 CUMULATIVE AND CANCELING EFFECTS

When several events occur at the same time, the effects of those events might differ from what the effects would have been had the events occurred at different times. Specifically, two or more concurrent events can produce new, additional effects.

In this case, we say that the effects of the events are cumulative. Further, one event can cancel the normal effects of another event.

We represent cumulative effects using modified effect axioms.

Definition 8.1. If γ is a condition, α and $\alpha_1, \ldots, \alpha_n$ are event terms, β is a fluent term, and τ is a timepoint term, then

$$\gamma \ \wedge (\neg)Happens(\alpha_1, \tau) \wedge \cdots \wedge (\neg)Happens(\alpha_n, \tau) \Rightarrow Initiates(\alpha, \beta, \tau)$$

is a ***positive cumulative effect axiom***. This represents that, if γ is true and α and $\alpha_1, \ldots, \alpha_n$ all occur at τ, then β will be true after τ.

Definition 8.2. If γ is a condition, α and $\alpha_1, \ldots, \alpha_n$ are event terms, β is a fluent term, and τ is a timepoint term, then

$$\gamma \ \wedge (\neg)Happens(\alpha_1, \tau) \wedge \cdots \wedge (\neg)Happens(\alpha_n, \tau) \Rightarrow Terminates(\alpha, \beta, \tau)$$

is a ***negative cumulative effect axiom***. This represents that if γ is true and α and $\alpha_1, \ldots, \alpha_n$ all occur at τ, then β will be false after τ.

We represent that one event cancels the effect of another event as follows. Suppose that a positive effect axiom represents that, if γ is true, then α_1 initiates a fluent β:

$$\gamma \Rightarrow Initiates(\alpha_1, \beta, \tau)$$

In order to represent that α_2 cancels the effect β of α_1, we rewrite this axiom as:

$$\gamma \ \wedge \neg Happens(\alpha_2, \tau) \Rightarrow Initiates(\alpha_1, \beta, \tau)$$

We treat negative effect axioms in a similar fashion.

8.2.1 EXAMPLE: CAMERA WITH FLASH

Consider a camera with a flash unit. If a camera's shutter is released at the same instant that its flash is triggered, then we have the cumulative effect that a properly exposed picture is taken:

$$\begin{gathered} Happens(TriggerFlash(c), t) \Rightarrow \\ Initiates(ReleaseShutter(c), ProperlyExposedPicture(t), t) \end{gathered} \tag{8.2}$$

If a camera's shutter is released and its flash is not triggered at the same instant, then an improperly exposed picture is taken:

$$\begin{gathered} \neg Happens(TriggerFlash(c), t) \Rightarrow \\ Initiates(ReleaseShutter(c), ImproperlyExposedPicture(t), t) \end{gathered} \tag{8.3}$$

The shutter is released at timepoints 0 and 1, but only at timepoint 1 is the flash triggered:

$$\neg ReleasedAt(f, t) \tag{8.4}$$

$$Happens(ReleaseShutter(Camera), 0) \tag{8.5}$$

$$Happens(TriggerFlash(Camera), 1) \tag{8.6}$$

$$Happens(ReleaseShutter(Camera), 1) \tag{8.7}$$

We can show that an improperly exposed picture is taken at timepoint 0 and that a properly exposed picture is taken at timepoint 1.

Proposition 8.1. *Let* $\Sigma = (8.2) \wedge (8.3)$, $\Delta = (8.5) \wedge (8.6) \wedge (8.7)$, $\Omega = U[\mathit{TriggerFlash}, \mathit{ReleaseShutter}] \wedge U[\mathit{ProperlyExposedPicture}, \mathit{ImproperlyExposedPicture}]$, *and* $\Gamma = (8.4)$. *Then we have*

$$\begin{gathered} \mathit{CIRC}[\Sigma; \mathit{Initiates}, \mathit{Terminates}, \mathit{Releases}] \wedge \mathit{CIRC}[\Delta; \mathit{Happens}] \wedge \\ \Omega \wedge \Gamma \wedge \mathrm{DEC} \\ \vdash \mathit{HoldsAt}(\mathit{ImproperlyExposedPicture}(0), 1) \wedge \\ \mathit{HoldsAt}(\mathit{ProperlyExposedPicture}(1), 2) \end{gathered}$$

Proof. From $\mathit{CIRC}[\Sigma; \mathit{Initiates}, \mathit{Terminates}, \mathit{Releases}]$ and Theorems 2.1 and 2.2, we have

$$\begin{aligned} &\mathit{Initiates}(e, f, t) \Leftrightarrow \\ &\exists c\,(e = \mathit{ReleaseShutter}(c) \wedge \\ &f = \mathit{ProperlyExposedPicture}(t) \wedge \\ &\mathit{Happens}(\mathit{TriggerFlash}(c), t)) \vee \\ &\exists c\,(e = \mathit{ReleaseShutter}(c) \wedge \\ &f = \mathit{ImproperlyExposedPicture}(t) \wedge \\ &\neg \mathit{Happens}(\mathit{TriggerFlash}(c), t)) \end{aligned} \tag{8.8}$$

$$\neg \mathit{Terminates}(e, f, t) \tag{8.9}$$

$$\neg \mathit{Releases}(e, f, t) \tag{8.10}$$

From $\mathit{CIRC}[\Delta; \mathit{Happens}]$ and Theorem 2.1, we have

$$\begin{aligned} &\mathit{Happens}(e, t) \Leftrightarrow \\ &(e = \mathit{ReleaseShutter}(\mathit{Camera}) \wedge t = 0) \vee \\ &(e = \mathit{TriggerFlash}(\mathit{Camera}) \wedge t = 1) \vee \\ &(e = \mathit{ReleaseShutter}(\mathit{Camera}) \wedge t = 1) \end{aligned} \tag{8.11}$$

From (8.11), we have $\neg \mathit{Happens}(\mathit{TriggerFlash}(\mathit{Camera}), 0)$. From this, (8.5) (which follows from (8.11)), (8.3) (which follows from (8.8)), and DEC9, we have $\mathit{HoldsAt}(\mathit{ImproperlyExposedPicture}(0), 1)$. From (8.6) and (8.7) (which both follow from (8.11)), (8.2) (which follows from (8.8)), and DEC9, we have $\mathit{HoldsAt}(\mathit{ProperlyExposedPicture}(1), 2)$. ■

8.2.2 EXAMPLE: MOVING ROBOT

Another example is that of a simple robot. At any time, the robot has a unique location in two-dimensional space:

$$\mathit{HoldsAt}(\mathit{Location}(r, x_1, y_1), t) \wedge \tag{8.12}$$

$$\begin{gathered}HoldsAt(Location(r,x_2,y_2),t) \Rightarrow \\ x_1 = x_2 \wedge y_1 = y_2\end{gathered}$$

And at any time, a robot has a unique direction given by an angle in radians measured counterclockwise from the x axis:

$$\begin{gathered}HoldsAt(Direction(r,d_1),t) \wedge \\ HoldsAt(Direction(r,d_2),t) \Rightarrow \\ d_1 = d_2\end{gathered} \tag{8.13}$$

A robot has a left wheel and a right wheel. The wheels can be moved separately or at the same time. Moving the left wheel cancels the effect of moving the right wheel, and moving the right wheel cancels the effect of moving the left wheel. Thus, if a robot moves only its left wheel, then the robot will rotate clockwise by $\pi/180$ radians (1 degree):

$$\begin{gathered}\neg Happens(MoveRightWheel(r),t) \wedge \\ HoldsAt(Direction(r,d),t) \Rightarrow \\ Initiates(MoveLeftWheel(r), Direction(r,d-\pi/180),t)\end{gathered} \tag{8.14}$$

$$\begin{gathered}\neg Happens(MoveRightWheel(r),t) \wedge \\ HoldsAt(Direction(r,d),t) \Rightarrow \\ Terminates(MoveLeftWheel(r), Direction(r,d),t)\end{gathered} \tag{8.15}$$

If a robot moves its only right wheel, then the robot will rotate counterclockwise by $\pi/180$ radians:

$$\begin{gathered}\neg Happens(MoveLeftWheel(r),t) \wedge \\ HoldsAt(Direction(r,d),t) \Rightarrow \\ Initiates(MoveRightWheel(r), Direction(r,d+\pi/180),t)\end{gathered} \tag{8.16}$$

$$\begin{gathered}\neg Happens(MoveLeftWheel(r),t) \wedge \\ HoldsAt(Direction(r,d),t) \Rightarrow \\ Terminates(MoveRightWheel(r), Direction(r,d),t)\end{gathered} \tag{8.17}$$

Moving the left and right wheels simultaneously has a cumulative effect. Namely, if a robot moves its left and right wheels at the same time, then the robot will move forward by one unit:

$$\begin{gathered}Happens(MoveLeftWheel(r),t) \wedge \\ HoldsAt(Location(r,x,y),t) \wedge \\ HoldsAt(Direction(r,d),t) \Rightarrow \\ Initiates(MoveRightWheel(r), Location(r,x+Cos(d),y+Sin(d)),t)\end{gathered} \tag{8.18}$$

$$\begin{gathered}Happens(MoveLeftWheel(r),t) \wedge \\ HoldsAt(Location(r,x,y),t) \Rightarrow \\ Terminates(MoveRightWheel(r), Location(r,x,y),t)\end{gathered} \tag{8.19}$$

Consider the following observations and narrative. The robot is initially at location $\langle 0, 0\rangle$, and its direction is 0 (facing right):

$$\neg ReleasedAt(f, t) \tag{8.20}$$

$$HoldsAt(Location(Robot, 0, 0), 0) \tag{8.21}$$

$$HoldsAt(Direction(Robot, 0), 0) \tag{8.22}$$

At timepoint 0, the robot moves its right wheel; at timepoint 1, the robot moves both its left and right wheels:

$$Happens(MoveRightWheel(Robot), 0) \tag{8.23}$$

$$Happens(MoveLeftWheel(Robot), 1) \tag{8.24}$$

$$Happens(MoveRightWheel(Robot), 1) \tag{8.25}$$

We can show that at timepoint 2, the robot will be at location $\langle Cos(\pi/180), Sin(\pi/180)\rangle$.

Proposition 8.2. *Let* $\Sigma = (8.14) \wedge (8.15) \wedge (8.16) \wedge (8.17) \wedge (8.18) \wedge (8.19)$, $\Delta = (8.23) \wedge (8.24) \wedge (8.25)$, $\Omega = U[MoveLeftWheel, MoveRightWheel] \wedge U[Direction, Location]$, $\Psi = (8.12) \wedge (8.13)$, *and* $\Gamma = (8.20) \wedge (8.21) \wedge (8.22)$. *Then we have*

$$\begin{gathered} CIRC[\Sigma; Initiates, Terminates, Releases] \wedge CIRC[\Delta; Happens] \wedge \\ \Omega \wedge \Psi \wedge \Gamma \wedge \text{DEC} \\ \vdash HoldsAt(Location(Robot, Cos(\pi/180), Sin(\pi/180)), 2) \end{gathered}$$

Proof. From $CIRC[\Sigma; Initiates, Terminates, Releases]$ and Theorems 2.1 and 2.2, we have

$$\begin{aligned} & Initiates(e, f, t) \Leftrightarrow \\ & \exists r, d\, (e = MoveLeftWheel(r) \wedge \\ & f = Direction(r, d - \pi/180) \wedge \\ & \neg Happens(MoveRightWheel(r), t) \wedge \\ & HoldsAt(Direction(r, d), t)) \vee \\ & \exists r, d\, (e = MoveRightWheel(r) \wedge \\ & f = Direction(r, d + \pi/180) \wedge \\ & \neg Happens(MoveLeftWheel(r), t) \wedge \\ & HoldsAt(Direction(r, d), t)) \vee \\ & \exists r, x, y, d\, (e = MoveRightWheel(r) \wedge \\ & f = Location(r, x + Cos(d), y + Sin(d)) \wedge \\ & Happens(MoveLeftWheel(r), t) \wedge \\ & HoldsAt(Location(r, x, y), t) \wedge \\ & HoldsAt(Direction(r, d), t)) \end{aligned} \tag{8.26}$$

$$\begin{aligned} & Terminates(e, f, t) \Leftrightarrow \\ & \exists r, d\, (e = MoveLeftWheel(r) \wedge \end{aligned} \tag{8.27}$$

$$\begin{aligned}
& f = Direction(r, d) \wedge \\
& \neg Happens(MoveRightWheel(r), t) \wedge \\
& HoldsAt(Direction(r, d), t)) \vee \\
& \exists r, d\ (e = MoveRightWheel(r) \wedge \\
& f = Direction(r, d) \wedge \\
& \neg Happens(MoveLeftWheel(r), t) \wedge \\
& HoldsAt(Direction(r, d), t)) \vee \\
& \exists r, x, y, d\ (e = MoveRightWheel(r) \wedge \\
& f = Location(r, x, y) \wedge \\
& Happens(MoveLeftWheel(r), t) \wedge \\
& HoldsAt(Location(r, x, y), t)) \\
& \neg Releases(e, f, t) \qquad (8.28)
\end{aligned}$$

From $CIRC[\Delta; Happens]$ and Theorem 2.1, we have

$$\begin{aligned}
& Happens(e, t) \Leftrightarrow \qquad (8.29) \\
& (e = MoveRightWheel(Robot) \wedge t = 0) \vee \\
& (e = MoveLeftWheel(Robot) \wedge t = 1) \vee \\
& (e = MoveRightWheel(Robot) \wedge t = 1)
\end{aligned}$$

From (8.29), we have $\neg Happens(MoveLeftWheel(Robot), 0)$. From this, (8.23) (which follows from (8.29)), (8.22), (8.16) (which follows from (8.26)), and DEC9, we have

$$HoldsAt(Direction(Robot, \pi/180), 1) \qquad (8.30)$$

From (8.29) and (8.27), we have $\neg\exists e\ (Happens(e, 0) \wedge Terminates(e, Location(Robot, 0, 0), 0))$. From this, (8.21), (8.20), and DEC5, we have $HoldsAt(Location(Robot, 0, 0), 1)$. From this, (8.24) and (8.25) (which both follow from (8.29)), (8.30), (8.18) (which follows from (8.26)), and DEC9, we have $HoldsAt(Location(Robot, Cos(\pi/180), Sin(\pi/180)), 2)$. ■

BIBLIOGRAPHIC NOTES

Our discussion of restricting concurrency draws on the notions of "exclusive and shared resources" of Pinto (1994, p. 59), the "complete exclusion" and "conflict exclusion" of Ernst, Millstein, and Weld (1997, p. 1171), and the "interference" and "competing needs" of Blum and Furst (1997, p. 286). The technique of preventing more than one event from occurring at a time is used in the event calculus by Shanahan and Witkowski (2004). The example of putting a penny in a jar is based on the example of loading cargo into a rocket of Veloso (1989).

Concurrent events were originally considered in the situation calculus. Schubert (1990, pp. 56-61) introduces a function $\mathit{Costart}(a_1, a_2)$ that maps two actions a_1 and a_2 that start at the same time into a new action that is composed of a_1 and a_2 and that stops when a_1 and a_2 have both stopped. He gives the example of two people lifting a sofa (p. 56), but does not treat cumulative and canceling effects. Gelfond, Lifschitz, and Rabinov (1991) do treat cumulative and canceling effects. They describe the soup bowl scenario (p. 174), based on Pednault's (1987) scenario of two people lifting a table from opposite sides: If a soup bowl is lifted with one hand, the soup spills. If the bowl is lifted with both hands, the soup does not spill. Gelfond et al. use the predicate $\mathit{Cancels}(a, f)$ to represent that an action a (such as *Lift*) cancels the effect of a on a fluent f (such as *Spilled*). F. Lin and Shoham (1992, 1995) introduce "global actions" and "primitive actions" (F. Lin & Shoham, 1995, p. 304). A global action consists of a set of primitive actions. The predicate $\mathit{Canceled}(g_1, g_2, s)$ represents that a primitive action in g_2 cancels the global action g_1.

Pinto (1994, pp. 57-66) introduces a function + that maps two actions a_1 and a_2 into the action of performing a_1 and a_2 simultaneously. He discusses the "precondition interaction problem," which is that, in a given situation it may be possible to perform a_1 and it may be possible to perform a_2, yet it may not be possible to perform $a_1 + a_2$. He uses the example of an agent that can paint wall A or paint wall B but cannot paint both walls A and B simultaneously. He proposes a predicate $\mathit{precInt}(a_1, a_2)$, which represents that the preconditions of a_1 interact with the preconditions of a_2, and the axiom (p. 59):

$$\mathit{Poss}(a_1 + a_2, s) \equiv \mathit{Poss}(a_1, s) \land \mathit{Poss}(a_2, s) \land \neg \mathit{precInt}(a_1, a_2)$$

The predicate $\mathit{Poss}(a, s)$ represents that it is possible to perform action a in situation s. Pinto proposes that $\mathit{precInt}(a_1, a_2)$ could be determined on the basis of which resources are required to perform a_1 and a_2 and whether the use of those resources must be exclusive or can be shared. For example, a brush, wall, paint canister, and light are required to paint a wall. The use of the brush, wall, and paint canister must be exclusive, but the use of the light can be shared.

Pinto (1994) also discusses the "effect interaction problem," which is that concurrent actions may have cumulative and canceling effects, called "synergy" and "cancellation" (p. 60). He proposes to deal with this problem by breaking effects into "primary" and "secondary" effects (p. 61). Primary effects are the effects that must always occur when an action is performed; secondary effects are the effects that are subject to synergy and cancellation. He represents primary effects using effect axioms and secondary effects using state constraints. He treats the soup bowl scenario by using state constraints that determine when the bowl of soup is lifted or spilled. The analog in the event calculus consists of two effect axioms and two state constraints:

$$\mathit{Initiates}(\mathit{LiftLeft}, \mathit{LiftedLeft}, t)$$

$$\mathit{Initiates}(\mathit{LiftRight}, \mathit{LiftedRight}, t)$$

$$\mathit{HoldsAt}(\mathit{Lifted}, t) \Leftrightarrow \mathit{HoldsAt}(\mathit{LiftedLeft}, t) \land \mathit{HoldsAt}(\mathit{LiftedRight}, t)$$

$$\mathit{HoldsAt}(\mathit{Spilled}, t) \Leftrightarrow \mathit{HoldsAt}(\mathit{LiftedLeft}, t) \dot{\lor} \mathit{HoldsAt}(\mathit{LiftedRight}, t)$$

Reiter (1996) presents a variation on Pinto's (1994) approach. Reiter adds a sort called "concurrent" (p. 3) for sets of actions, redefines *do* as a function from sets of actions and situations to situations, and discusses successor state axioms that take concurrency into account. See also the review of Reiter (2001, chap. 7). Baral and Gelfond (1997) extend action language $\mathcal{A}$ for concurrent actions.

Following the work in the situation calculus, Shanahan (1995a, pp. 265-268; 1997b, pp. 301-304) introduces a method for handling concurrent events with cumulative and canceling effects into the event calculus. The notation $e_1 \& e_2$ (also $e_1 + e_2$) represents a compound event that consists of e_1 and e_2. The predicate $Cancels(e_1, e_2)$ represents that, if e_1 occurs concurrently with e_2, then e_1 cancels the effects of e_2. Shanahan extends the circumscriptive event calculus for concurrent events and shows that it can handle a problem involving a shopping cart. If the cart is pushed, then it moves forward; if it is pulled, then it moves backward. But if it is pushed and pulled at the same time, then it spins around. Shanahan (1999a, pp. 424-426) incorporates concurrent events into the forced separation version of the event calculus. In this case, the predicate is $Cancels(e_1, e_2, f)$, which represents that e_1 cancels the effect of e_2 on fluent f. Finally, R. Miller and Shanahan (1999, pp. 7-8; 2002, pp. 460-461) show that concurrent events with cumulative and canceling effects can be represented using effect axioms with *Happens* preconditions, which we call cumulative effect axioms. R. Miller and Shanahan (1999, p. 8; 2002, pp. 460-461) show how to handle the soup bowl scenario using their approach. Shanahan (1996) uses the example of a robot but not to demonstrate concurrent events with cumulative and canceling effects. R. Miller and Shanahan (1996) treat the case of a robot that can vary its bearing continuously as it moves.

EXERCISES

8.1 Write an algorithm that takes an axiomatization (list of axioms) as input and produces suggestions for possible event occurrence constraints as output. The idea is that these constraints are examined by a person, who decides whether they should be added to the axiomatization. The algorithm should work by looking for events that initiate or terminate fluents that are the preconditions of other events. Specifically, whenever it is the case that

- $Initiates(\alpha_1, \beta, \tau)$, and $\neg HoldsAt(\beta, \tau)$ is an action or fluent precondition of α_2, or
- $Terminates(\alpha_1, \beta, \tau)$, and $HoldsAt(\beta, \tau)$ is an action or fluent precondition of α_2,

(where α_1 and α_2 are event terms, β is a fluent term, and τ is a timepoint term), the algorithm should produce as output an axiom that states that α_1 and α_2 cannot occur simultaneously.

8.2 Extend the algorithm in Exercise 8.1 to suggest event occurrence constraints for similar actions of one or two agents.

8.3 Formalize the mixing of two colors of paint together as follows. If yellow paint is placed on a palette, the result is yellow. If blue paint is placed on a palette, the result is blue. But if yellow and blue paint are placed on a palette, the result is green.

8.4 Formalize the ripping of a piece of paper as follows. If the paper is pulled left, it moves left. If it is pulled right, it moves right. But if it is simultaneously pulled left and right, it rips. Discuss what a more detailed axiomatization of ripping a paper would look like.

8.5 Formalize the clapping of hands.