Lecture 4: Temporal Planning with Continuous Processes

Mikhail Soutchanski

(Acknowledgement: Vitaliy Batusov, Giuseppe De Giacomo, Mikhail Soutchanski, “Hybrid Temporal Situation Calculus", pages 11-13, `https://arxiv.org/abs/1807.04861`
Shaun Mathew and M.Soutchanski "Heuristic Planning for Hybrid Dynamical Systems with Constraint Logic Programming", available at `https://ceur-ws.org/Vol-3585/`)

TMU, Department of Computer Science
https://www.cs.torontomu.ca/mes

July 10, 2025

Motivation and Abstract

Since situation terms are discrete, it might seem that the situation calculus cannot represent continuous processes and their evolution in time, like an object falling under the influence of gravity.

However, one can view a process as a fluent – $falling(s)$ – which becomes true at the time t when the instantaneous action $startFalling(t)$ occurs, and becomes false at the time t of occurrence of the instantaneous action $endFalling(t)$. One can then write axioms that describe the evolution in time of the falling object quantities such as its velocity.

Before we can talk about continuous change, we need representation of time. So far, we talked about situations and sequences of actions that imply there is an order between actions. But we did not talk about physical time.

Suppose actions are paired with moments of time. When we solve the planning problem, we encounter a new problem: at what moments of time the actions have to be executed ? To schedule plan actions we can choose to execute them as earlier as possible. This means we have to solve an optimization problem for the moments of time.

We'll use an external Non-Linear Programming (NLP) software to solve the related minimum makespan optimization problem.

Assumption: each action either initiates or terminates a continuous process.

Prerequisites: basic understanding of Linear Programming at the level of an introductory CS course on algorithms.

The Situation Tree with a Timeline.

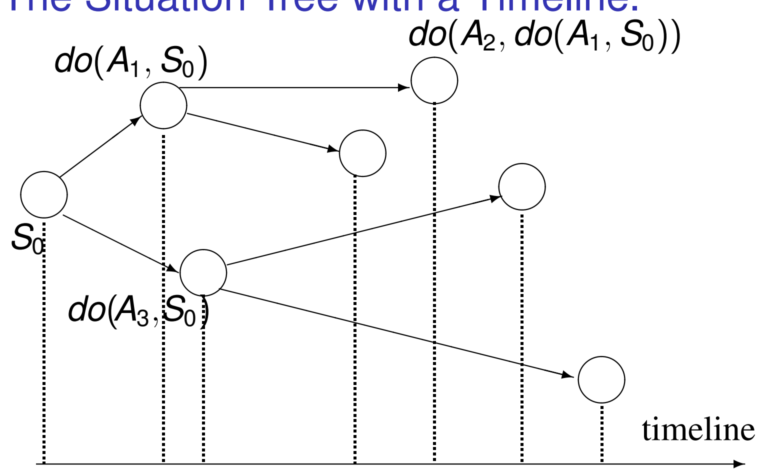

Figure: Suppose that actions A_1, A_3 are possible in S_0, and they result in situations $do(A_1, S_0)$ and $do(A_3, S_0)$. Also, shown is situation $do(A_2, do(A_1, S_0))$ that results from doing A_2 in $do(A_1, S_0)$. To avoid clutter other actions and the resulting situations are not named, and branching rightwards is omitted. Note each circle maps to a point on the timeline representing the unique moment when situation starts. Each situation may last for an interval from the moment it starts until next action occurs.

The sequential, temporal situation calculus.

We consider only sequencies of actions. No “concurrency" here, but see R.Reiter's book (Chapter 7) for details. Now: add an explicit representation for time to the sequential situation calculus. How?

Add a new temporal argument to all instantaneous actions, denoting the actual time at which that action occurs. Thus, $startMeeting(Susan, t)$ might be the instantaneous action of Susan starting a meeting at time t. Extend the foundational axioms for the situation calculus to accommodate time. How?

Recall the situation tree with the root in S_0. Imagine this tree with the time-line underneath. Relate action occurrences with time points since actions are instantaneous. Recall that $(s_1 \sqsubseteq s_2)$ means that the situation s_1 precedes s_2, i.e., s_1 occurs earlier than s_2 on the branch from S_0 to s_2. Our task: *write axioms relating this sequential order with temporal order.*

(1) We introduce a new function symbol $time : action \mapsto reals$. The number $time(a)$ denotes the time of occurrence of action a. This means that in any application involving a particular action $A(x, t)$, we shall need an axiom telling us the time of the action A, e.g., $time(A(x, t)) = t$. Example:

$$\forall t \forall person.\ \ time(startMeeting(person, t)) = t.$$

Foundational axioms for time

(2) It is convenient to have a new function symbol $start: situation \mapsto time$, where $start(s)$ denotes the start time of situation s. This requires the new foundational axiom:

$$start(do(a,s)) = time(a).$$

No axiomatization for rational numbers. (We could borrow the axioms for the dense linear order with the left end point.) The numbers in the subsequent axioms always have the standard interpretation and handled outside of the reasoner. This approach is related to semantic attachment [Weyhrauch] and Constraint Logic Programming [Colmerauer,Jaffar,Maher,Stuckey,Wallace].

(3) Next, we have to reconsider the abbreviation $executable(s)$ on situations. Recall it means intuitively that all actions in a sequence s should be consecutively possible. But now we do not want to consider

$$do(bounce(Ball, Floor, \mathbf{4}), do(startMeeting(Susan, \mathbf{6}), S_0))$$

as a possible action sequence. To coordinate occurrences of actions with time we suitably amend $executable(s)$:

$$executable(s) \stackrel{def}{=} \forall a \forall s'. do(a,s') \sqsubseteq s \rightarrow (poss(a,s') \wedge start(s') \leq time(a)),$$

Now, $executable(s)$ means that all the actions in s are possible, and moreover, the times of those action occurrences are nondecreasing.

Notice the constraint $start(s') \leq time(a)$ permits sequences in which the time of an action a may be the same as the time $start(s')$ of a preceding action.

Example: Bouncing Balls

Consider a finite number of balls that can be dropped and that can elastically bounce from the floor.
Agent actions: $drop(b, time)$ and $catch(b, time)$.
Natural events: $bounce(b, time)$ - ball b hits the floor, and $atPeak(b, time)$ - ball is at the top point of its trajectory.

The atemporal fluent $falling(b,s)$ means the ball b is falling down and accelerating under the Earth gravity. The atemporal fluent $flying(b,s)$ means the ball b bounced, it is flying up in situation s and decelerating due to gravity. The vertical axis is oriented downwards, i.e., if a ball is falling down, then its speed is positive and increases. But when the ball bounces, its speed is negative and decreases.

Consider functional temporal fluent $distance(b,t,s)$ that represents the ball b's height at the moment of time t within s, and functional temporal fluent $velocity(b,t,s)$ that characterizes instantaneous velocity of b at the moment t within the time interval as long as situation s lasts.

These temporal fluents describe time dependent change within situation, in between two occurrences of the agent actions and/or the natural events.

Atemporal Fluents vs Temporal Fluents

Previously, we had only *atemporal* fluents: they do not mention time. We will not use them to model continuously varying physical quantities. Atemporal fluents serve to specify the context in which continuous processes operate.

For example, the fluent $Falling(b,s)$ holds if in situation s a ball b is in the process of falling down and accelerating under the Earth gravity (9.81 m/s^2. For the duration of s, the position of the ball (and its derivatives) change as a function of time according to the equations of free fall.

Fluent $Falling(b,s)$ is directly affected by instantaneous actions $drop(b,t)$ (ball begins to fall at time t) and $catch(b,t)$ (ball stops at t). But these actions only change the context, thereby switching the continuous trajectory that the ball can follow. Namely, a falling ball is one context and a ball at rest is another.

In a general case, a context expression is a boolean combination of atemporal logical fluents. Numerical fluents in a context: future work.

In a general case, there are finitely many (parameterized) context types which are pairwise mutually exclusive.

To model continuously varying physical quantities, we introduce new functional *temporal fluents* with time as an argument. Values of these fluents change with time, and not only as a direct effect of instantaneous actions.

Important: even if situation does not change, temporal fluents change with time within situation that can last for an interval of time.

Preconditions in Logic with Temporal Constraints

$$\forall s \forall t \forall b.\ poss(drop(b,t),s) \leftrightarrow ball(b) \wedge \neg falling(b,s) \wedge \neg flying(b,s) \wedge \underline{t \geq start(s)}.$$

The agent action $drop(b,t)$ is possible in s at the moment of time t, if a ball b is neither falling, nor flying in s, and the moment of time $start(s)$ when s started is $\leq t$. (Due to $\forall t$ the branching factor is infinite for a planner.)

In an implementation, the temporal constraint $t \geq start(s)$ is added to a special data structure for a constraint store to be evaluated later at run time when the planner checks whether the goal logical conditions are satisfied.

$$\forall s \forall t \forall b.\ poss(catch(b,t),s) \leftrightarrow ball(b) \wedge (falling(b,s) \vee flying(b,s)) \wedge \underline{t \geq start(s)}.$$

$$\forall s \forall t \forall b.\ poss(bounce(b,t),s) \leftrightarrow ball(b) \wedge falling(b,s) \wedge \underline{distance(b,t,s) = 0} \wedge \underline{velocity(b,t,s) \geq \epsilon} \wedge \underline{t \geq start(s)}.$$

In an implementation, use external *eplex* LP solver to deal with the numerical constraints. An action can be possible only if the constraints are feasible.

$$\forall s \forall t \forall b.\ poss(atPeak(b,t),s) \leftrightarrow ball(b) \wedge \underline{distance(b,t,s) \geq 0} \wedge \underline{velocity(b,t,s) = 0} \wedge flying(b,s) \wedge \underline{t \geq start(s)}.$$

The last axiom is saying that a natural event $atPeak(b,t)$ can occur in s at the moment of time t if the ball b is flying up in s so that it reached its highest point at which its velocity is 0, but its hight is positive. An implementation of this axiom adds several more numerical constraints on the variables to the constraint store.

SSAs for Atemporal Fluents

There are 3 possible contexts:

1. one where the ball is at rest,
2. one where it is falling down, and
3. one where the ball is flying up after it bounced.

$$(\forall a \forall s \forall b).\ falling(b, do(a,s)) \leftrightarrow \exists t(a = drop(b,t)) \vee \exists t(a = atPeak(b,t)) \vee falling(b,s)) \wedge (\neg \exists t(a = catch(b,t)) \wedge \neg \exists t(a = bounce(b,t))$$

$$(\forall a \forall s \forall b).\ flying(b, do(a,s)) \leftrightarrow \exists t(a = bounce(b,t)) \vee flying(b,s)) \wedge \neg \exists t(a = catch(b,t)) \wedge \neg \exists t(a = atPeak(b,t))$$

SSAs to Initialize Temporal Fluents

When actions occur, the temporal change can be either continuous, or there might be jumps or resets in the values of temporal fluents. To describe these transitions in temporal fluents due to actions when new situation starts, we use auxiliary functional fluents $init_{dist}(b,d,s)$ and $init_{vel}(b,v,s)$.

Fluent $init_{dist}$

$$\forall s \forall y \forall a \forall b.\ init_{dist}(b, do(a,s)) = y \ \leftrightarrow\ distance(time(a), s) = y$$

The height of the ball changes continuously, no matter what actions happen.

The velocity y of the ball b resets to 0, when the agent catches the ball. When the ball bounces, its velocity jumps to the quantity with the opposite sign. All other actions with any other balls have no effect on these physical quantities at the moment when new situation starts.

$$\forall s \forall y \forall a \forall b.\ init_{vel}(b, do(a,s)) = y \leftrightarrow \exists y_0 . y_0 = velocity(time(a), s) \wedge \exists t(a = catch(b,t) \wedge y = 0) \ \vee\ \exists t(a = bounce(b,t) \wedge y = -y_0) \vee (\neg \exists t(a = bounce(b,t)) \wedge \neg \exists t(a = catch(b,t)) \wedge y = y_0).$$

State Evolution Axioms (SEA) in FOL

Each SEA characterizes how temporal fluent changes with time within a context determined by atemporal fluents. Each temporal fluent evolves from its initial value, determined by the corresponding *init* atemporal fluent, at the moment when situation starts. (The acceleration due to gravity is 9.81)

$$(\forall s \forall t \forall b).\ distance(b,t,s) = y \leftrightarrow\ \exists y_0 . y_0 = init_{dist}(b,s) \wedge (\neg falling(b,s) \wedge \neg flying(b,s) \wedge y = y_0) \vee (falling(b,s) \wedge \underline{y = y_0 - \int_{start(s)}^{t} (9.81 \cdot x)\, dx}) \vee (flying(b,s) \wedge \underline{y = y_0 + \int_{start(s)}^{t} (9.81 \cdot x)\, dx}).$$

$$(\forall s \forall t \forall b). velocity(b,t,s) = y \leftrightarrow\ \exists y_0 . y_0 = init_{vel}(b,s) \wedge (\neg falling(b,s) \wedge \neg flying(b,s) \wedge y = y_0) \vee (falling(b,s) \wedge \underline{y = y_0 + \int_{start(s)}^{t} 9.81 dx}) \vee (flying(b,s) \wedge \underline{y = y_0 - \int_{start(s)}^{t} 9.81 dx}).$$

In an implementation, collect all (underlined) numerical constraints in a data structure. Postpone evaluation until the planner checks if s is a goal state.

Since our preliminary implementation relied on the *eplex* library, we assumed that the balls move along straight lines instead of physically correct quadratic trajectories. Then, equations for both height and velocity are linear wrt time.

If an associated objective is also a linear function of its arguments, then optimization reduces to solving the Linear Programming (LP) problem.

Foundational Axioms for Situations and Time

All the following axioms have straightforward implementation in PROLOG.

Foundational Axioms from Chapters 4 and 7 of Ray Reiter's book [2001].

$\forall a_1 \forall a_2 \forall s_1 \forall s_2 . do(a_1, s_1) = do(a_2, s_2) \rightarrow a_1 = a_2 \wedge s_1 = s_2$

$\forall s . \neg (s \sqsubset S_0)$

$\forall a \forall s \forall s' . s \sqsubset do(a, s') \leftrightarrow s \sqsubseteq s'$, where $s \sqsubseteq s'$ means $(s \sqsubset s' \vee s = s')$

$\forall P . (P(S_0) \wedge \forall a \forall s (P(s) \rightarrow P(do(a,s)))) \rightarrow \forall s P(s)$

$\forall a, s' . do(a, s') \sqsubseteq s \rightarrow (poss(a, s') \wedge start(s') \le time(a)) \wedge \forall a' (poss(a', s) \wedge natural(a') \wedge a \ne a' \rightarrow time(a') \le time(a))$

$\forall a.\ start(do(a,s)) = time(a) \quad start(S_0) = 0$

Domain Specific Axioms for the Bouncing Ball Example

$\forall t, \forall b. time(drop(b,t)) = t$	$\forall t, \forall b. time(catch(b,t)) = t$
$\forall t, \forall b. time(bounce(b,t)) = t$	$\forall t, \forall b. time(atPeak(b,t)) = t.$
$\forall t \forall b. natural(atPeak(b,t))$	$\forall t \forall b. natural(bounce(b,t)).$
$\forall t \forall b. agent(drop(b,t))$	$\forall t \forall b. agent(catch(b,t)).$

Initial Theory

$ball(b1)$	$ball(b2)$
$velocity(b1, 0, S_0) = 0$	$velocity(b2, 0, S_0) = 0$
$distance(b1, 0, S_0) = 100$	$distance(b2, 0, S_0) = 150$

In a PROLOG program, functional fluent $distance(b,t,s)$ is implemented as predicate $dist(b,d,t,s)$, and functional fluent $velocity(b,t,s)$ is implemented as predicate $vel(b,v,t,s)$. Program is available at *https://www.cs.torontomu.ca/mes/publications/*

Planning Problem for the Two Balls

Objective: find a plan that satisfies a goal in minimal time wrt constraints.

Find *the earliest moment of time* such that each ball reached its peak at least once, both balls are falling, the velocities of the two balls are equal, and their heights are also equal. Minimization wrt constraints collected so far.

Checking these goal conditions reduces to the linear programming problem that can be solved using an external *eplex* LP solver interfaced with our program. Solved this planning instance with an uninformed iterative deepening depth-first search (DFS) planner.

The program found a correct 8 step plan in 0.18 seconds: *[drop(b2, 0), bounce(b2, 30.581039755351682), drop(b1, 50.9683995922528), atPeak(b2, 61.162079510703364), catch(b2, 71.35575942915392), bounce(b1, 71.35575942915392), drop(b2, 91.743119266055047), atPeak(b1, 91.743119266055047)]*.

Notice that this plan must be clever since the two balls had different initial heights: *dist(b1,100,0,[]). dist(b2,150,0,[]).*
But in the goal state their heights and velocities must be equal.

In an implementation, precondition axioms for nature's actions appear before preconditions for agent's actions. In a general case, need extra efforts to make sure nature's actions are executed as soon as they are possible.

Deriving State Evolution Axioms

Having combined all laws which govern the evolution of f with time into a single axiom (3), we can make a causal completeness assumption (Explanation Closure): *there are no other conditions under which the value of f can change in s from its initial value at $start(s)$ as a function of t*, i.e.,

$$f(\bar{x}, t, s) \neq f(\bar{x}, start(s), s) \to \exists y\, \Phi(\bar{x}, y, t, s). \quad (5)$$

Theorem

Let for each formula of the form (1) the background theory $\mathcal{D}$ entail $\forall(\gamma(\bar{x}, s) \to \exists y\, \delta(\bar{x}, y, t, s))$. Then the conjunction of axioms (1), (3), (4), (5) is logically equivalent to

$$f(\bar{x}, t, s) = y \leftrightarrow [\Phi(\bar{x}, y, t, s) \vee y = f(\bar{x}, start(s), s) \wedge \neg\Psi(\bar{x}, y, t, s)], \quad (6)$$

where $\Psi(\bar{x}, s)$ denotes $\bigvee_{1 \le i \le k} \gamma_i(\bar{x}, s)$.

We call the formula (6) a *state evolution axiom* (SEA) for the fluent f. Note what the SEA says: f evolves with time during s according to some law whose context is realized in s or stays constant if no context is realized.

See proof in the paper Vitaliy Batusov, Giuseppe De Giacomo, Mikhail Soutchanski, "Hybrid Temporal Situation Calculus", pages 11-13, `https://arxiv.org/abs/1807.04861`

Temporal Change Axiom (TCA) in a General Case

Our starting point is a *temporal change axiom* (TCA) which describes the evolution of a particular temporal fluent due to the passage of time in a particular context of an arbitrary situation: Similar to $vel(b, t, s)$.

$$\gamma(\bar{x}, s) \wedge \delta(\bar{x}, y, t, s) \to f(\bar{x}, t, s) = y, \quad (1)$$

where $t, s, \bar{x}, y$ are variables and $\gamma(\bar{x}, s)$, $\delta(\bar{x}, y, t, s)$ are formulas uniform in s. We call $\gamma(\bar{x}, s)$ the *context* as it specifies the condition under which formula $\delta(\bar{x}, y, t, s)$ provides (may be implicitly) the value y to fluent f at time t.

$$\gamma(\bar{x}, s) \to \exists y\, \delta(\bar{x}, y, t, s). \quad (2)$$

For each TCA, we require that whatever the circumstance, the axiom supplies a value for the quantity modelled by f if its context is satisfied.

A finite set of k temporal change axioms for fluent f can be equivalently expressed as follows, where $\Phi(\bar{x}, y, t, s)$ is $\bigvee_{1 \le i \le k}(\gamma_i(\bar{x}, s) \wedge \delta_i(\bar{x}, y, t, s))$.

$$\Phi(\bar{x}, y, t, s) \to f(\bar{x}, t, s) = y \quad (3)$$

$$\Phi(\bar{x}, y, t, s) \wedge \Phi(\bar{x}, y', t, s) \to y = y'. \quad (4)$$

Condition (4) guarantees the consistency of the axiom (3) by preventing a continuous quantity from having more than one value at any moment of time. With condition (4), all contexts in the given set of TCA are pairwise mutually exclusive wrt a BAT $\mathcal{D}$. Note that contexts $\gamma(\bar{x}, s)$ are *time-independent*.

ODEs in Temporal Change Axioms

$\gamma(\bar{x}, s) \wedge \delta(\bar{x}, y, t, s) \to f(\bar{x}, t, s) = y$ /*Why no condition ($t > start(s)$) on LHS?*/
Let us discuss more specifically what can be inside the formula δ.

Science, engineering and PDDL+ describe continuous effects in dynamical systems in terms of an explicit ordinary differential equation (ODE) of the form

$$\frac{df(\bar{x}, t, s)}{dt} = RHS(t, f)$$

where $RHS(t, f)$ is a continuously differentiable (or more generally, a Lipschitz continuous function). An initial value $f(\bar{x}, t_0, s)$ is implicitly given, and together with the ODE it defines the *initial value problem* that has a unique solution.

To avoid introducing additional fluents for derivatives, we can encode process effects using the equivalent integral form $f(\bar{x}, t, s) = \int_{t_0}^{t} RHS(\tau, f)\, d\tau + f(t_0)$, where we use notation t_0 instead of $start(s)$. We require that all FOL structures interpret the definite integral symbol in the standard way.

Let $\mathcal{M}$ be an arbitrary situation calculus structure, σ be an object assignment, $h(\bar{x}, t, s)$ a SC term of sort $\mathbb{R}$ whose free variables are among $\bar{x}, t, s$, and let τ_1, τ_2 be terms of sort $\mathbb{R}$. Then we require that

$$\left(\int_{\tau_1}^{\tau_2} h(\bar{x}, t, s)\, dt\right)^{\mathcal{M}}[\sigma] = \int_{\tau_1^{\mathcal{M}}[\sigma]}^{\tau_2^{\mathcal{M}}[\sigma]} h^{\mathcal{M}}(\bar{x}^{\mathcal{M}}[\sigma], t, s^{\mathcal{M}}[\sigma])\, dt.$$

The modeller must ensure that $h^{\mathcal{M}}(\bar{x}^{\mathcal{M}}[\sigma], t, s^{\mathcal{M}}[\sigma])$ is a continuous real-valued function defined on the interval $[\tau_1^{\mathcal{M}}[\sigma], \tau_2^{\mathcal{M}}[\sigma]]$.

For several related temporal fluents $f_1, f_2, \ldots, f_n$ assume a system of ODEs, in a context $\gamma(\bar{x}, s)$ where all these fluents change in s simultaneously.

Temporal Basic Action Theory (BAT)

The SEA for a temporal fluent f does not completely specify the behaviour of f because it talks only about change within s. Need a SSA describing how the initial value of f changes (or does not change) when an action is performed.

How to relate $f(\bar{x}, time(a), do(a, s))$ with $f(\bar{x}, time(a), s)$? Enforce "=" or not? Transition is not always continuous, e.g., object's acceleration changes from 0 to $-9.8m/s^2$ when an object is dropped. Need ability to model action-induced discontinuous jumps in the values of the continuously varying quantities.

For each temporal functional fluent $f(\bar{x}, t, s)$, we introduce an *auxiliary atemporal* functional fluent $f_{init}(\bar{x}, s)$ whose value in s represents the value of the temporal fluent f in s at the time instant $start(s)$. Add new SSA for f_{init}:

$$f_{init}(\bar{x}, do(a, s)) = y \leftrightarrow \exists y'. f(\bar{x}, time(a), s) = y' \wedge Init(\bar{x}, y', y, a, s), \qquad (7)$$

where $Init(\bar{x}, y', y, a, s)$ is a formula uniform in s whose purpose is to describe how the initial value y of f_{init} in $do(a, s)$ relates to the temporal fluent f value y' at the same time instant in s (i.e., prior to execution of action a).

To establish the relationship between temporal fluents and their atemporal *init*-counterparts, we require $\mathcal{D}_{ss} \wedge \mathcal{D}_{se} \models f(\bar{x}, start(s), s) = f_{init}(\bar{x}, s)$.

A *temporal basic action theory* is $\mathcal{D} = \Sigma \cup \mathcal{D}_{ss} \cup \mathcal{D}_{ap} \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0} \cup \mathcal{D}_{se}$ such that $\Sigma \cup \mathcal{D}_{ss} \cup \mathcal{D}_{ap} \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0}$ constitutes a BAT, and $\mathcal{D}_{se}$ is a set of state evolution axioms.

Example 2: a Water Reservoir

inflow ——→ | *water* | max volume = tank capacity is C
rate R_{in} | *volume* | min volume = 0
| v | ——→ outflow rate R_{out}

Consider a water reservoir with an adjustable inflow and an adjustable outflow. Let the temporal functional fluent $vol(t, s)$ represent the volume of water in the tank at time t. The maximum capacity of the tank is C. Let R_{in} and R_{out} be inflow and outflow rates (volume per unit time), which are for simplicity are constant, i.e., rates are time-invariant.

Let actions $startIn(t)$, $endIn(t)$ represent opening/closing of the inflow valve. These actions initiate/terminate the process represented as the fluent $inflow(s)$. Let actions $startOut(t)$, $endOut(t)$ represent opening/closing of the outflow valve. These actions initiate/terminate the process $outflow(s)$.

$Poss(startIn(t), s) \leftrightarrow \neg inflow(s) \wedge t \geq start(s) \wedge (vol(t, s) < C)$.
$Poss(endIn(t), s) \leftrightarrow inflow(s) \wedge t \geq start(s)$.
$Poss(startOut(t), s) \leftrightarrow \neg outflow(s) \wedge t \geq start(s) \wedge (vol(t, s) > 0)$.
$Poss(endOut(t), s) \leftrightarrow outflow(s) \wedge t \geq start(s)$.

$inflow(do(a, s)) \leftrightarrow \exists t(a = startIn(t)) \vee inflow(s) \wedge \neg\exists t(a = endIn(t))$
$outflow(do(a, s)) \leftrightarrow \exists t(a = startOut(t)) \vee outflow(s) \wedge \neg\exists t(a = endOut(t))$

Satisfiability of a Temporal BAT

A set $\mathcal{D}_{se}$ of SEA is *stratified* iff there are no temporal fluents $f_1, \ldots, f_n$ such that $f_1 \succ f_2 \succ \ldots \succ f_n \succ f_1$ where $f \succ f'$ holds iff there is a SEA in $\mathcal{D}_{se}$ where f appears on the left-hand side and f' on the right-hand side. A temporal BAT is *stratified* iff its $\mathcal{D}_{se}$ is.

Example: distance varies with time and depends on velocity that in turn depends on acceleration.

Similarly to Reiter's BATs, temporal BATs have a relative satisfiability property.

Theorem

A stratified temporal BAT $\mathcal{D}$ is satisfiable iff $\mathcal{D}_{una} \cup \mathcal{D}_{S_0}$ is satisfiable.

The proof extends that of Theorem 1 in [Pirri&Reiter,JACM,1999]. This important result provides means of checking satisfiability of the theory without having to reason about situations using the 2nd-order induction axiom of Σ.

There are simple practical examples of temporal BAT such that their temporal fluents are not stratified. However, it turns out one can prove for them as well that such temporal BATs have a model under reasonable conditions.

Reservoir: a State Evolution Axiom (SEA) for Volume

An obvious initial value SSA asserts the continuity of volume (no leaks):
$vol_{init}(do(a, s)) = v \leftrightarrow vol(time(a), s) = v$.

To write a State Evolution Axiom for the fluent $vol(t, s)$ consider all possible combinations of inflow and outflow: either the inflow valve is open while outflow is present or not, or the inflow valve is closed while the outflow valve can be open or closed. Each combination is a separate context. For each context, $vol(t, s)$ evolves according to a different function of flow rates and time. Inflow cannot exceed capacity C, and outflow cannot yield $vol(t, s) < 0$.

$$\begin{aligned} vol(t, s) = v \leftrightarrow \exists v_0 \exists t_0 (\ & vol_{init}(s) = v_0 \wedge start(s) = t_0 \wedge \\ & (inflow(s) \wedge \neg outflow(s) \wedge v = \min\{v_0 + R_{in} \cdot (t - t_0), C\} \vee \\ & inflow(s) \wedge outflow(s) \wedge v = \max\{\min\{v_0 + R_{in} \cdot (t - t_0) - R_{out} \cdot (t - t_0), C\}, 0\} \vee \\ & \neg inflow(s) \wedge outflow(s) \wedge v = \max\{v_0 - R_{out} \cdot (t - t_0), 0\} \vee \\ & \neg inflow(s) \wedge \neg outflow(s) \wedge v = v_0)\ ). \end{aligned}$$

The expression $\neg inflow(s) \wedge \neg outflow(s)$ on the last line of the SEA is logically equivalent to the negated disjunction of the three contexts on preceding lines. Notice that in this axiom there are four different pairwise exclusive contexts, and for each context there is its own function describing how the fluent evolves with time.