Best-practice time point ontology for event calculus-based temporal reasoning

Robert C. Schrag
Digital Sandbox, Inc.
McLean, VA USA
bschrag@dsbox.com

***Abstract*—We argue for time points with zero real-world duration as a best ontological practice in point- and interval-based temporal representation and reasoning. We demonstrate anomalies that unavoidably arise in the event calculus when real-world time intervals corresponding to finest anticipated calendar units (e.g., days or seconds, per application granularity) are taken (naively or for implementation convenience) to be time "points." Our approach to eliminating the undesirable anomalies admits durations of infinitesimal extent as the lower and/or upper bounds that may constrain two time points' juxtaposition. Following Dean and McDermott, we exhibit axioms for temporal constraint propagation that generalize corresponding naïve axioms by treating infinitesimals as orthogonal first-class quantities and we appeal to complex number arithmetic (supported by programming languages such as Lisp) for straightforward implementation. The resulting anomaly-free operation is critical to effective event calculus application in commonsense understanding applications, like machine reading.**

***Index Terms*—temporal knowledge representation and reasoning, event calculus, temporal ontology best practices, temporal constraint propagation**

I. Introduction

Machine reading technology recently has been applied to extract temporal knowledge from text. The event calculus [8] presents appropriate near-term targets for formal statements about events, time-varying properties (i.e., fluents), and time points and intervals. While at least one implemented event calculus-based temporal logic [2] also has included calendar dates and clock times, most classical event calculus treatments address real-world time only abstractly. None so far has adopted the carefully crafted formulation of points (instants), intervals, dates, and times in Hobbs' and Pan's RDF temporal ontology [4]—which correctly treats all time units as intervals. We say, "correctly," because the casual treatment of a calendar or clock unit as a time point unavoidably leads to undesirable anomalies. This point may be subtle—ISO standard 8601 [3] pertaining to representation of dates and times states, "On a time scale consisting of successive steps, two distinct instants may be expressed by the same time point," and also (unfortunately, apparently circularly) defines an instant as a "point on the time axis." We hope, by demonstrating anomalies resulting from incorrect time point treatment and by presenting effective correct implementation techniques, to motivate future best-practice event calculus-based applications.

II. Event Calculus Ontology and Axioms

We have implemented a temporal reasoning engine for an event calculus variant including the following ontological elements.

- Time intervals are convex collections of time points—intuitively, unbroken segments along a time axis.
- The ontological status of time points is an issue contended here. We argue that in the best practice they are taken to be instants with no real-world temporal extent, while naïvely (we argue incorrectly) finest anticipated calendar or clock units—which actually are intervals—have been taken as time "points." We take a time point to be a degenerate time interval—one whose beginning and ending points both are the time point itself.
- Fluents are statements representing time-varying properties—e.g., the number of living children a person has.
- The events of interest occur at individual time points and may cause one or more fluents to change truth value. E.g., the event of adopting an only child will cause the fluent hasChildren(Person, 0) to become false and the fluent hasChildren(Person, 1) to become true.

Figure 1 exhibits axioms defining the predicates we use to say when fluents "hold" (are true) and when events "occur" (happen).

holdsThroughout(fluent, interval) ↔ ∀(point): pointInInterval(point, interval) → holdsAt(fluent, point)
holdsThroughout(fluent, interval) ↔ ∀(sub): hasSubInterval(interval, sub) ∧ holdsThroughout(fluent, sub)

holdsAt(fluent, point) ↔ ∃(interval): intervalIsPoint(interval, point) ∧ holdsThroughout(fluent, interval)

holdsWithin(fluent, interval) ↔ ∃(sub): hasSubInterval(interval, sub) ∧ holdsThroughout(fluent, sub)

occursWithin(event, interval) ↔ ∃(point): pointInInterval(point, interval) ∧ occursAt(event, point)

Figure 1. Axioms relating holds and occurs predicates. Variables appearing on the left-hand side of an initial implication are universally quantified. Variables introduced on the right-hand side are quantified as indicated. The predicates relating time points and intervals are defined in the appendix.

Informally, a fluent holds throughout an interval *I* iff it holds at every point and throughout every subinterval contained by *I*. It holds (or occurs) within *I* iff it holds (or occurs) within some subinterval (or point) contained by *I*.

In the naïve approach, it's perfectly acceptable to assert that a fluent holds or that an event occurs "at" a specific "point" on the calendar or clock. We believe that under the preferred approach, in which the only (true) points directly accessible delimit the boundaries of measured time units, such assertions (or even queries) should be rare—perhaps limited to issues of legal status (e.g., one reaches the age of majority at exactly 12:00 midnight on one's 21st birthday). Thus, we commend preferred use of holdsWithin and occursWithin to replace naïve use of holdsAt and occursAt.

Besides being correct, the preferred approach is also more robust. In the naïve approach, supposing an enterprise decides to enhance its represented granularity from days to hours, it will need to replace all existing occurrences of holdsAt with holdsWithin (because its working definition of a "point" will have changed). As such, naïve approach users might as well avoid holdsAt and just use holdsWithin, which has equivalent semantics when its interval argument is a time point.

A given event calculus application also will include axioms to indicate which transition events initiate or terminate which fluents, as summarized by Schrag [7]. We don't need that much detail here, however, to demonstrate our concerns about undesirable anomalies arising from the naïve approach.

III. Anomalies Arising from the Naïve Time Point Approach

We discuss the following anomalies.

A. Inability to order time points within a finest represented time unit (e.g., a calendar day—see section A)
B. Inability to avoid inferred logical contradiction when contradictory statements hold at different real-world times within a finest represented time unit (see section B)
C. Inability to order real-world events occurring within a finest represented time unit (see section C)
D. Inability to avoid inferred logical contradiction when real-world events occur within a finest represented time unit and initiate contradictory fluents (see section D)

The time map in Figure 2 illustrates these anomalies, as discussed in the following subsections.

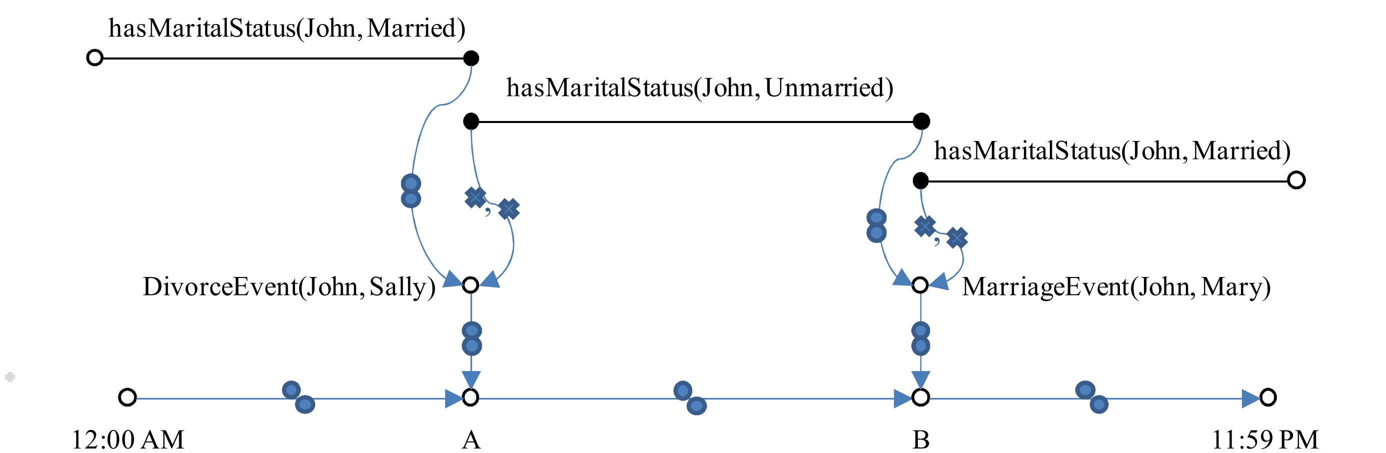

Figure 2. Time map illustrating naïve approach anomalies. Fluent observations (top) include fluents and the intervals throughout which they hold. Dark-filled points indicate that associated fluents are known not to hold beyond their intervals' beginning or ending. Constraint graphics (with arrows) are defined in Figure 9, in the appendix. Transition event occurrences (middle) include the events and points where these occur. Contradictory fluents cannot overlap temporally, and, per event calculus convention, initiated fluent observations begin immediately after triggering transition events. The calendar (bottom) shows the initial and final minutes of a given day, plus two included time points, ordered as shown.

A. Inability to order time points

As is apparent in Figure 2, this basic problem underlies the other three listed above. In the naïve approach, the only way to order time points is to associate them with distinct finest calendar or clock units. Suppose days are the finest time unit represented. We'd like to assert the point-wise temporal relations (i.e., constraints) Figure 2 indicates, but in the naïve approach such constraints would be contradictory—all the points shown would resolve to the same calendar day's time "point," which cannot precede itself. This anomaly can be particularly troubling in the representation of statements extracted by machine reading from news articles, which frequently exhibit only calendar dates but cover sequences of events occurring within single days. The option of discarding such fine ordering information—and treating all within-day events as if they were simultaneous—is equally problematic. Rendering event orderings correctly is critical to representing causality—just one fundamental element of a true commonsense understanding that machine reading is hoped ultimately to support.

Even when our representation isn't fine enough to specify absolutely when during a given day (e.g.) a time point occurs, when we can order the points, we can avoid contradictions resulting from an incorrect presumption of simultaneity. Absent total (or even partial) ordering, we also can still hypothesize orders that might not lead to contradictions.

B. Inability to order contradictory holds statements

A person can't be both married and unmarried at the same time, as would be required if all the constraint-linked points in Figure 2 were collapsed onto a single day "point." In the naïve approach, it is (from a real-world perspective) as if we forced every marriage or divorce (indeed, every event) to occur at the stroke of midnight.

C. Inability to order events

In the naïve approach, we can say that a person divorced one spouse and married another on the same day, but we can't say in what order these events occurred.

D. Inability to order occurs statements initiating contradictory fluents

Without the ability to order events, we don't know whether any axiom proscribing polygamy has been violated or not. An implementation might take one position or another, depending on the order in which it happened to visit the transition events and to apply its rules for initiating and terminating fluents, detecting contradictions, and propagating constraints.

IV. Temporal Constraint Representation and Propagation

Compared to an application's finest represented calendar or clock unit, available real-world information may be more or less precise. E.g., we may know the year that a given event occurred but not the month or the day. If our finest represented units are days, this gives us an earliest and a latest possible date on which the event could have occurred (the first and last days of the year given). We use the notation distance(a, b, [x, y]) to indicate that the number of finest time units along a path from time point a to time point b has as a lower bound x and as an upper bound y.

Rather than expose our system-internal time units, we provide a user interface in terms of calendar and clock units—affording users source code-level robustness against future granularity enhancements. A distinguished calendar/clock point (e.g., the beginning point of the interval for 12:00 midnight, January 1, 1900) affords a reference against which the distance to other dates/times is calculated.

We refer to an asserted distance statement (or to a user-provided statement from which it is derived) as a temporal constraint.

Real-world information also may give us only qualitative information about the relationship between two time points—e.g., one is before or one is after the other. The following two figures exhibit axioms to define qualitative relations among time points—Figure 3 following the naïve approach, Figure 4 the preferred one. (See also Figure 9 in the appendix for graphical definitions of these relations.) Notice that the only difference between these two axiom sets is in their representation of the smallest possible distance between any two time points. In the naïve approach, it is one finest time unit. In the preferred approach, it is arbitrarily small—taken to be infinitesimal.

timePointEqualTo(a, b) ↔ distance(a, b, [0, 0])
timePointLessThan(a, b) ↔ distance(a, b, [1, ∞])
timePointGreaterThan(a, b) ↔ distance(a, b, [−∞, −1])
timePointGreaterThanOrEqualTo(a, b) ↔ distance(a, b, [0, ∞])
timePointLessThanOrEqualTo(a, b) ↔ distance(a, b, [−∞, 0])
hasNextTimePoint(a, b) ↔ distance(a, b, [1, 1])
hasPreviousTimePoint(a, b) ↔ distance(a, b, [−1, −1])
timePointTouching(a, b) ↔ distance(a, b, [−1, 1])
timePointGreaterThanOrTouching(a, b) ↔ distance(a, b, [−1, ∞])
timePointLessThanOrTouching(a, b) ↔ distance(a, b, [−∞, 1])

Figure 3. Axioms defining qualitative relations between time points in the naïve approach, where finest time units are treated as "points" and the smallest possible distance is one such time unit

timePointEqualTo(a, b) ↔ distance(a, b, [0, 0])
timePointLessThan(a, b) ↔ distance(a, b, [ϵ, ∞])
timePointGreaterThan(a, b) ↔ distance(a, b, [–∞, −ϵ])
timePointGreaterThanOrEqualTo(a, b) ↔ distance(a, b, [0, ∞])
timePointLessThanOrEqualTo(a, b) ↔ distance(a, b, [–∞, 0])
hasNextTimePoint(a, b) ↔ distance(a, b, [ϵ, ϵ])
hasPreviousTimePoint(a, b) ↔ distance(a, b, [−ϵ, −ϵ])
timePointTouching(a, b) ↔ distance(a, b, [−ϵ, ϵ])
timePointGreaterThanOrTouching(a, b) ↔ distance(a, b, [−ϵ, ∞])
timePointLessThanOrTouching(a, b) ↔ distance(a, b, [–∞, ϵ])

Figure 4. Axioms defining qualitative relations between time points in the preferred approach, where all time units are treated as intervals and we use an infinitesimal (denoted ϵ) to separate points that are (in the limit) "adjacent"

Both approaches use infinity (denoted ∞) to represent the largest possible distance between time points. Handling this in temporal constraint propagation (computing tightest distance bounds, considering all constraints) requires axioms defining non-standard arithmetic, as in Figure 5. Figure 6 exhibits axioms for the constraint propagation process in which Figure 5's arithmetic axioms are applied. Note that all but the last of Figure 5's axioms handle only the infinities specially. By treating the positive infinitesimal denoted ϵ as the imaginary number *i* (as in [2][5][6]) and by appealing to complex arithmetic, we can use the same axioms to support propagation in both approaches.

Note that in the naïve approach using only real numbers all the imaginary parts will be zero. The only substantive difference between the two approaches' computational complexity for constraint propagation is that the preferred approach enables finer (and thus more numerous unique) constraints.

Implementation is straightforward for addition and arithmetic negation in a programming language such as Lisp that supports complex numbers and arithmetic. While complex numbers with unequal real and/or imaginary parts are incomparable with respect to magnitude, in our imaginary-as-infinitesimal interpretation the real parts always dominate and the imaginary parts are compared only when the real parts are equal—per the last axiom defining finite>, in which the predicates real>, real=, and imaginary> invoke the indicated comparisons on the real and imaginary parts of their arguments.

infinite(–∞)
infinite(∞)

infinite+(–∞, –∞, –∞)
infinite+(∞, ∞, ∞)
infinite+(a, –∞, –∞) ← ¬infinite(a)
infinite+(–∞, b, –∞) ←¬infinite(b)
infinite+(a, ∞, ∞)← ¬infinite(a)
infinite+(∞, b, ∞) ← ¬infinite(b)
infinite+(a, b, a + b) ← ¬infinite(a) ∧ ¬infinite(b)

infinite–(–∞, ∞)
infinite–(∞, –∞)
infinite–(a, –a) ← ¬infinite(a)

infinite>(∞, –∞)
infinite>(a, –∞) ← ¬infinite(a)
infinite>(∞, b) ← ¬infinite(b)
infinite>(a, b) ← ¬infinite(a) ∧ ¬infinite(b) ∧ finite>(a, b)

finite>(a, b) ← real>(a, b) ∨ (real=(a, b) ∧ imaginary>(a, b))

Figure 5. Axioms supporting constraint propagation arithmetic (addition, subtraction, and comparison) over temporal duration bounds of infinite extent

distance(b, a, [–y, –x]) ↔ distance(a, b, [x, y]) ∧ infinite–(x, –x) ∧ infinite–(y, –y)
distance(a, b, [w, y]) ← distance(a, b, [x, y]) ∧ distance(a, b, [w, z]) ∧ infinite>(w, x)
distance(a, b, [x, z]) ← distance(a, b, [x, y]) ∧ distance(a, b, [w, z]) ∧ infinite>(y, z)
distance(a, c, [mo, np]) ← distance(a, b, [m, n]) ∧ distance(b, c, [o, p]) ∧ infinite+(m, o, mo) ∧ infinite+(n, p, np)

Figure 6. Axioms for propagating lower and upper temporal bounds to infer tightest bounds considering all constraints

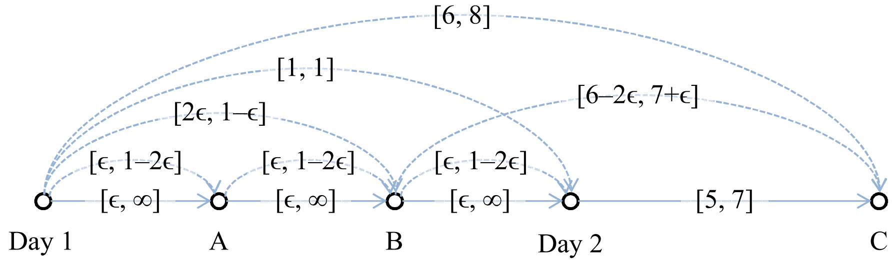

Figure 7. Raw (solid arrow) and inferred/propagated (dashed arrow) constraints, with lower and upper bounds, in the preferred approach. Constraints have directions indicated by arrows (all oriented from left to right)

V. How the Preferred Approach Avoids Anomalies

To see how constraint propagation works—and avoids anomalies—in the preferred approach, see Figure 7, which supposes days are our finest time unit.

By way of raw constraints, we know that points A and B both fall between Day 1 and Day 2, that A follows B, and that point C is between five and seven days after Day 2. For clarity, Figure 7 omits the [ϵ, ∞] constraint from Day 1 to B and from A to Day 2, as well as many inferred constraints relating pairs of points not connected in the figure. The two-dimensional (in the implementation, complex) arithmetic treating infinitesimal and non-infinitesimal quantities orthogonally effectively maintains qualitative point ordering—both within finest represented calendar or clock unit boundaries (e.g., relating points A and B) and across them (relating B and C). See Figure 8.

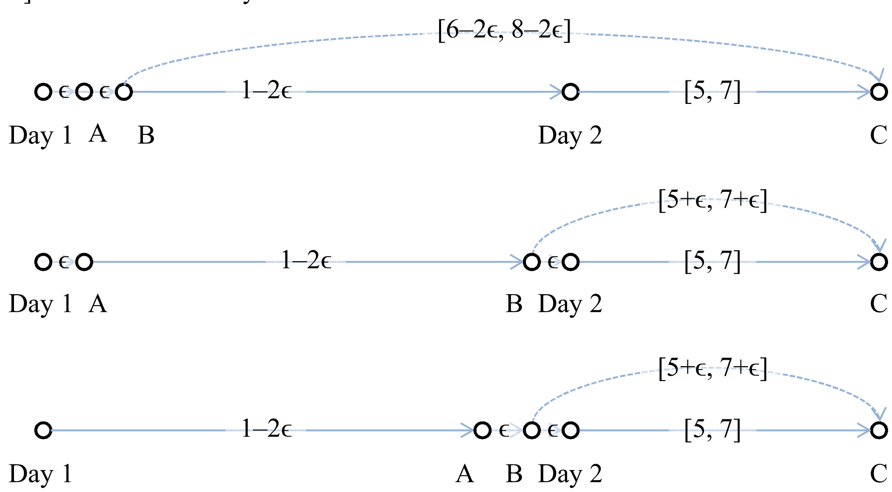

Figure 8. Extreme cases for the time points A and B in Figure 7, including (at the extremes) greatest lower and least upper bounds in the inferred constraints shown there

As we explained in section III, resolving this time point ordering anomaly simultaneously resolves the other three anomalies described there as well. Now, we also can order the events that occur at time points and avoid spurious contradictions that arise from the naïve approach's inability to order events and fluent observations. When our finest time units are days, we no longer have to pretend that all events occur at the stroke of midnight. With appropriate ordering of events, we'll be able to put machine reading in a better position to support commonsense understanding of causality.

VI. Summary

We have demonstrated temporal reasoning anomalies that arise when implementation of the event calculus naively follows classical treatments that casually treat finest represented calendar or clock time intervals as "points." We have presented axioms and described implementation techniques to resolve these anomalies when all time intervals are correctly treated as time intervals and when time points are taken to be instants with zero real-world duration extent. We argue that this preferred approach, rather than the naïve one, is needed for the event calculus to be useful in applications, like machine reading, intended to support commonsense understanding including causality.

Acknowledgment

Thanks to other participants in DARPA's Machine Reading research program that supported our temporal reasoning implementation—especially to other members of the SAIC evaluation team, including Global InfoTek (the author's former employer).

References

[1] J. Allen, "Maintaining knowledge about temporal intervals," in Communications of the ACM. 26, pp. 832–843, November 1983.

[2] T. Dean and D. McDermott, "Temporal data base management," Artificial Intelligence, vol. 32, pp. 1–55, 1987.

[3] International Standards Organization, "Data elements and interchange formats—information interchange—representation of dates and times," international standard ISO 8601:2004(E), third edition, 2004.

[4] J. Hobbs and F. Pan, "An ontology of time for the semantic web," ACM Transactions on Asian Language Information Processing, Vol. 3, No. 1, pp. 66–85, March 2004.

[5] R. Schrag, J. Carciofini, and M. Boddy, "Beta-TMM Manual (version b19)," Technical Report CS-R92-012, Honeywell SRC, 1992.

[6] R. Schrag, M. Boddy, and J. Carciofini. "Managing disjunction for practical temporal reasoning," in Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR-92), pp 36–46, 1992.

[7] R. Schrag, "Exploiting inference to improve temporal RDF annotations and queries for machine reading," 7th International Conference on Semantic Technologies for Intelligence, Defense, and Security (STIDS), 2012.

[8] M. Shanahan, "The event calculus explained," in Artificial Intelligence Today, ed. M. Wooldridge and M. Veloso, Springer Lecture Notes in Artificial Intelligence no. 1600, pp.409–430, 1999.

Appendix: Time point and interval relations

The set of predicates illustrated in Figure 9 (repeated from Figure 4) supports every qualitative binary time point relation over the time point distance landmark values indicating equality, adjacency, and lack of constraint above or below. (A user also may specify arbitrary bounds on the number of time units intervening between any two points.) As illustrated in Figure 10 selected examples, this point orientation yields a much broader set of qualitative interval relations than does Allen's classical formalism [1], which is purely interval oriented, without points.

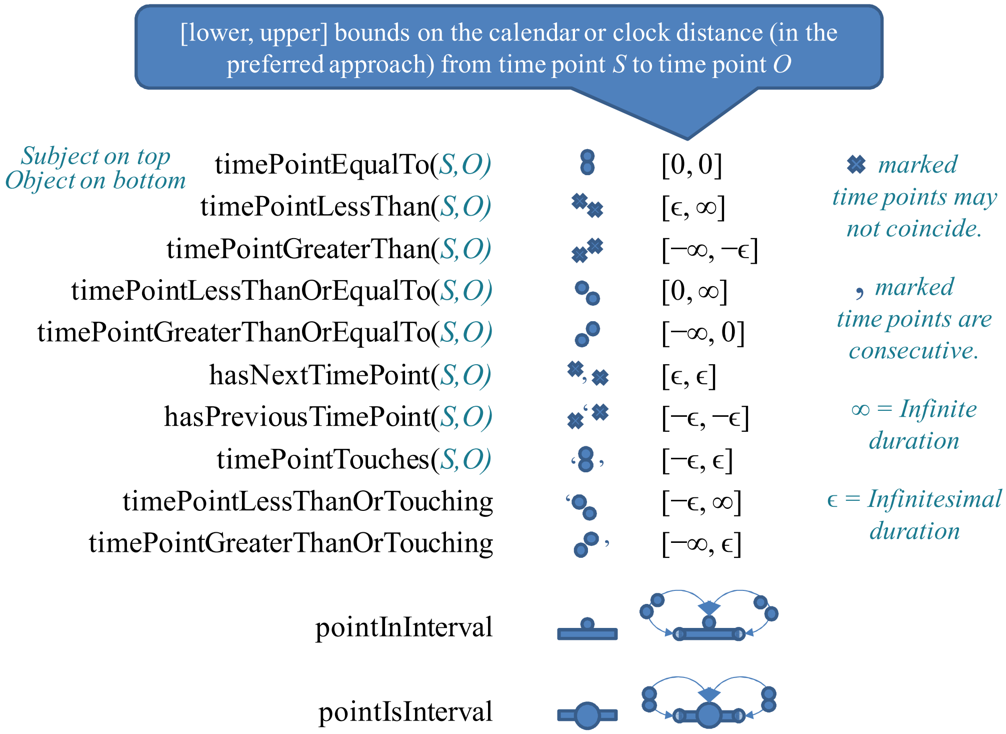

Figure 9. Qualitative relations over time points, with graphical icons that we use to illustrate the definitions of point-and-interval relations (here) and interval-interval relations (in Figure 10). Such illustrated definitions include beginning and ending points super-imposed on interval icons, to elucidate the constraints.

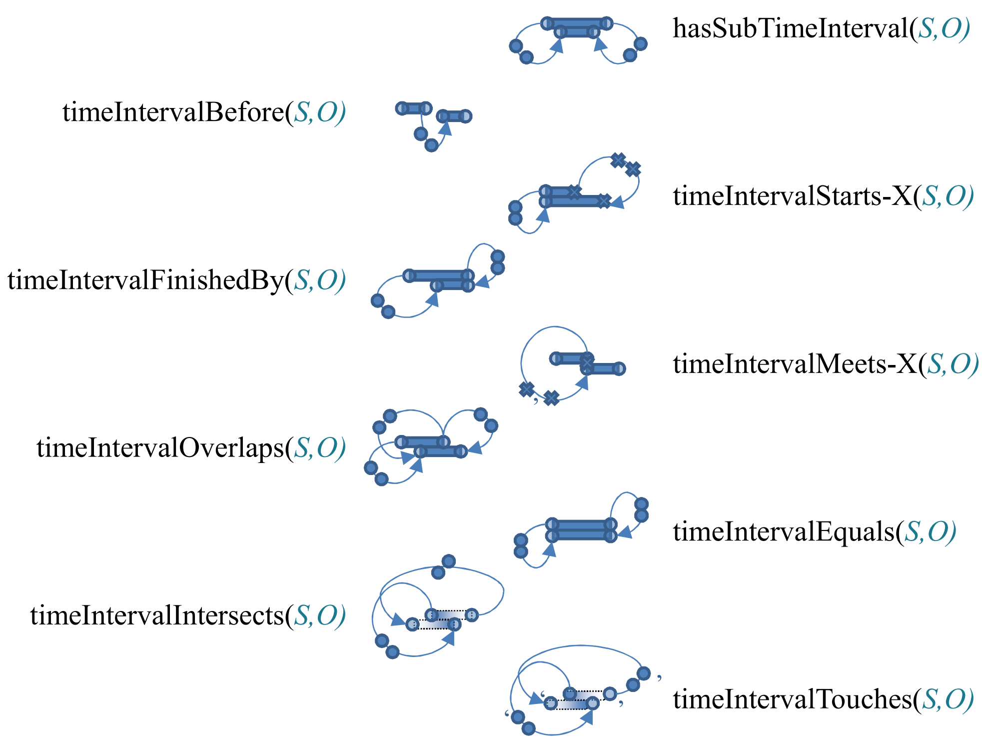

Figure 10. Selected relations over time intervals (with defined time point relations indicated)