Reconciling Point-Based and Interval-Based Semantics in Temporal Relational Databases: A Treatment of the Telic/Atelic Distinction

Paolo Terenziani and Richard T. Snodgrass, *Senior Member*, *IEEE*

Abstract—The analysis of the semantics of temporal data and queries plays a central role in the area of temporal databases. Although many different algebræ and models have been proposed, almost all of them are based on a point-based (snapshot) semantics for data. On the other hand, in the areas of linguistics, philosophy, and, recently, artificial intelligence, an oft-debated issue concerns the use of an interval-based versus a point-based semantics. In this paper, we first show some problems inherent in the adoption of a point-based semantics for data, then argue that these problems arise because there is no distinction drawn in the data between *telic* and *atelic facts*. We then introduce a three-sorted temporal model and algebra including coercion functions for transforming relations of one sort into relations of the other at query time which properly copes with these issues.

Index Terms—Temporal databases, database semantics, data models.

1 Introduction

Time plays an important role in real-world phenomena. There has been much work over the last two decades in incorporating time into data models, query languages, and database management system (DBMS) implementations. In particular, many extensions to the standard relational model have been proposed and more than 2,000 papers on temporal databases (TDBs) have been published over the last two decades (cf., a cumulative bibliography [48], four surveys [27], [38], [28], [24], and several workshop proceedings [20], [29], [39]).

An important issue about the treatment of temporal information in temporal databases is the semantics of data and queries. Many researchers realized the richness of the semantics of temporal data [15], [31] and provided various operators in different query languages [4], [34], [38]. Much work dealing with the semantics of temporal data modeling has appeared in the temporal database literature (e.g., [9], [16], [25], [30], [23]). Toman [41] has pointed out some problems connected with the definition of a clear semantics for the approaches where the validity times of tuples and attributes are encoded using time intervals. In fact, in most temporal models, time intervals (or sets of time intervals) are associated with tuples (or with attributes) instead of time points, but this is only a matter of efficient and compact implementation [12]. Many works (e.g., [16], [23], [43]) showed that, for instance, in SQL/Temporal, TSQL2, TSQL, HQL, and TQuel, data could be seen as a sequence of states indexed by time points. Thus, such approaches adopt a *point-based semantics for data*, in the sense discussed in Section 2 below.

A point-based semantics has limitations which have been studied in the areas of philosophy, linguistics, and artificial intelligence (henceforth: AI). However, the distinction between point-based and interval-based approaches in TDBs (see, e.g., [41], [43], [12], [7], [8]) is an entirely different one than the distinction in the above areas. In this paper, we show how the latter distinction can be profitably incorporated into temporal models and query languages and, in fact, we argue that this distinction is in some ways more central than the similarly named but orthogonal distinction in temporal databases. In [40], we sketch the relevance of such distinction in different fields of research, including philosophy, linguistics, cognitive science, and AI.

2 Preliminaries

The goal of this paper is to augment the usual point-based semantics with an interval-based semantics for data into temporal relational databases, motivated by analogous linguistic and philosophical distinctions. We chose to operate at the *algebraic level*, to adopt *tuple timestamping*, and to focus on *valid time*. Moreover, for the sake of clarity, we do not consider valid-time event relations [34] and we assume discrete time and a single granularity for all relations. In Section 3, we introduce a relational model and algebra, an adaptation of that of SQL/Temporal, which is taken as a paradigmatic example of algebraic relational approaches using tuple timestamping and point-based semantics for data. However, our goal in this paper is not that of providing a minimal nor complete set of algebraic operators, in the sense of [37, p. 524]. The modifications we made to SQL/Temporal's algebraic operators are primarily motivated by our goal of demonstrating relative weaknesses and strengths of point-based and interval-based

- *P. Terenziani is with the Dipartimento di Informatica, Universitá del Piemonte Orientale "Amedeo Avogadro," Corso Borsalino 54, 15100 Alessandria, Italy. E-mail: terenz@mfn.unipmn.it.*
- *R.T. Snodgrass is with the Department of Computer Science, University of Arizona. Tucson, AZ 85721. E-mail: rts@cs.arizona.edu.*

Manuscript received 22 June 2001; revised 18 Nov. 2002; accepted 8 Apr. 2003.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 114406.

semantics. We adopt the BCDM (Bitemporal Conceptual Data Model) [23]. In particular, we associate a set of time points with tuples. Our approach could also be adapted to cover attribute timestamping, as, e.g., in [19], because the distinctions between attribute and tuple timestamping is not one of semantics [23]. In the BCDM, time is seen as a finite sequence of *chronons* [21], isomorphic to a sequence of natural numbers. The sequence of chronons can be thought of as representing a partitioning of the time line into indivisible segments. Using chronons instead of discrete time points does not imply a significant change in our approach.

2.1 Distinctions

To set the stage, we provide some central definitions. Before stating these definitions, it is important to point out three distinctions, which are at the core of our approach:

1. *Representation versus semantics of the language*—The approach we propose concerns the temporal semantics of data and queries, independent of the representation one uses for time. This distinction is analogous to the distinction between *concrete* and *abstract* databases emphasized by Chomicki [11].
2. *Data language versus query language*—The two should be differentiated [12]. ATSQL2 [8], SQL/Temporal [35], [36], and SQL/TP [41], [43], [44] support time intervals in their data representation language; however, while the query languages of ATSQL2 [13] and SQL/Temporal are based on time intervals, that of SQL/TP is based on time points.
3. *Data semantics versus query semantics*—In most cases, within the database community, the semantics of *data* is not distinguished from the semantics of the *query*. On the other hand, data have their own semantics, independently of any query language and query operators. This is the usual approach in AI and logics: Logical formulæ have an intrinsic meaning, which can be formally defined in model-theoretic terms. Of course, queries are an operational way of making such a semantics explicit. However, a set of logical formulæ has a semantics *per se*, even if no query is asked. Analogously, we will say that data in a database have a semantics, which we will call "semantics for data" (or semantics, for short).

These distinctions will be emphasized and explicated throughout this paper.

2.2 Telic versus Atelic Facts

A very basic issue underlying many approaches in philosophy and linguistics is the fact that the usual human way of "capturing" reality (i.e., of representing it, or of describing it through natural language expressions) involves a *distinction between different classes of facts*. For the sake of brevity, in the following, we will just sketch such a distinction, considering only the linguistic viewpoint (see [40] for a more detailed discussion). Within the linguistic community, it is commonly agreed that natural language sentences can be classified within different *aktionsart classes* (e.g., *activities*, *accomplishments*, *achievements*, and *states* in [46]; also called *aspectual classes* [14]) depending on their linguistic behavior or their semantic properties. These semantic properties demonstrate that *the semantics of the association of facts to time depends on the classes of facts* being considered. For example, Dowty [18] proposed the following semantic criteria to distinguish between *states* and *accomplishments*:

1. A sentence ϕ is stative iff it follows from the truth of ϕ at an interval I that ϕ is true at all subintervals of I (e.g., if John was asleep from 1:00 to 2:00 p.m., then he was asleep at all subintervals of this interval: *be asleep* is a stative).
2. A sentence ϕ is an accomplishment/achievement (or *kinesis*) iff it follows from the truth of ϕ at an interval I that ϕ is false at all subintervals of I (e.g., if John built a house in exactly the interval from 1 September until 1 June, then it is false that he built a house in any subinterval of this interval: *build a house* is an accomplishment/achievement) [18, p. 42].

Property 1 for states has been often called *downward hereditary* in the TDB and AI literature (e.g., [4], [32]). Notice that also *upward hereditary* holds over states: If John was asleep from 1:00 to 2:00 and from 2:00 to 3:00, then he was asleep from 1:00 to 3:00.

Obviously, the aktionsart distinctions above have a deep impact on the semantic framework one has to adopt to model the meaning of sentences and of the facts they describe. *Point-based semantics* evaluate the truth of sentences over time points (see also the following section). These semantics perfectly work on stative facts: "John was asleep" in item 1 above is true exactly for all time points within 1:00 and 2:00 p.m. On the other hand, point-based semantics seems to be inadequate to deal with accomplishments. For instance, given 2, there is no specific time point p such that "John built a house" is true in p. "John built a house" is true (or, in other words, occurred) exactly in the *time interval* from 1 September to 1 June. This and analogous observations led most researchers in linguistics, starting from the works in [3], [17], [18], to criticize point-based semantics which *is not adequate to deal with the semantics of accomplishments* (while it works well for *states* and *activities*) *for which an interval-based semantics is needed*.

Different authors used different terminologies and models to deal with this phenomenon. For instance, Moens and Steedman [26] based their explanation on the fact that accomplishments are ***telic*** (from the Greek: "telos" meaning "goal") in the sense that they are characterized by the fact that they reach a *culmination* (goal or telos), while states (and activities) are ***atelic*** (from the Greek: "a" as a prefix indicates negation), i.e., do not have an intrinsic culmination.[1]

2.3 Basic Definitions

The fundamental tension examined in this paper is point-based versus interval-based; this characterization may be applied, somewhat orthogonally, to the semantics for data and to the semantics for queries. We first consider data; query semantics will be examined in later sections.

1. The telic/atelic dichotomy dates back to Aristotle (who first introduced such a terminology) [2]; cognitive science approaches have recently shown that it has strong cognitive evidence (see, e.g., [5]).

Definition (Point-based semantics for data). *The data in a temporal relation is interpreted as a sequence of states (with each state a conventional relation: a set of tuples) indexed by points in time. Each state is independent of every other state. Such temporal relations can be encoded in many different ways (data language). For example, the following are three different encodings of the same information, within a point-based semantics, of John being married to Mary in the states indexed by the times 1, 2, 7, 8, and 9:*

1. $< \text{John}, \text{Mary} \mid\mid 1, 2, 7, 8, 9 > \in R.$
2. $< \text{John}, \text{Mary} \mid\mid \{[1-2], [7-9]\} > \in R.$
3. $< \text{John}, \text{Mary} \mid\mid [1-2] > \in R$ *and*

$$< \text{John}, \text{Mary} \mid\mid [7-9] > \in R.$$

The fact denoted by <John, Mary> is in five individual states, as follows:

$$1 \rightarrow \{< \text{John}, \text{Mary} >]\} \quad 2 \rightarrow \{< \text{John}, \text{Mary} >\}$$
$$7 \rightarrow \{< \text{John}, \text{Mary} >\} \quad 8 \rightarrow \{< \text{John}, \text{Mary} >\}$$
$$9 \rightarrow \{< \text{John}, \text{Mary} >\}.$$

Definition (Interval-based semantics for data). *Each tuple in a temporal relation is associated with a set of time intervals which are the temporal extents in which the fact described by the tuple occur. In this semantics, the index is a time interval. Time intervals are atomic primitive entities in the sense that they cannot be decomposed. Note, however, that time intervals can overlap; there is no total order on time intervals, unlike time points.*

For example, let $< \text{John} \mid\mid [10-20] >$ represent the fact that John started to build a house at time 10 and finished at time 20. If an interval-based semantics is adopted, the interval [10-20] is interpreted as an atomic (indivisible) one.

1. $[10, 20] \rightarrow \{< \text{John} >\}.$

Note that, if an interval-based semantics is used, this tuple would *not* imply that John built the house in [12-15] or at the time point 12 or at any other time interval different than [10-20].[2]

3 Atelic Temporal Model and Algebra

Here, we first introduce a paradigmatic example of an approach based on point-based (snapshot) semantics (and on tuple timestamping), which is a simplification and an adaptation of SQL/Temporal's model (i.e., the BCDM). However, this choice is not critical as most (if not all) extant *temporal models* are point-based in their data, though we hasten to add that many temporal *query languages* are a mixed of point-based and interval-based (atelic and telic).

2. Again, our definition of interval-based semantics for data is also independent of the representation formalism. For instance, one could choose to represent time intervals as a set of points and nevertheless adopt the interval-based semantics. If an interval-based semantics is adopted, <John | | {{10, 11, 12, 13, 14, 15, 6, 17, 18, 19, 20}}> denotes exactly the same content as <John | | [10-20]}> above.

TABLE 1
STOCK^A Relation

Name	Category	Price	VT
IBM	High-tech	63	{14,15,16,17}
IBM	High-tech	61	{12,13}
GM	Industrial	49	{10,11,12,16,17,18}

3.1 Atelic Temporal Model

We assume that time is a linear, ordered, and discrete set of time points $p \in \mathcal{P}$ (e.g., isomorphic to integers) [22]. We adopt the term *fact* for any statement that can be meaningfully assigned a truth value (i.e., that is either true or false), we represent facts with *tuples* and we use the *valid time* of a fact (tuple) to represent the collected time when the fact is true. As in most TDB approaches (see below), we adopt a point-based semantics for data. Thus, we associate with each tuple an *atelic element*, which is the set of time points when the fact holds.

Concerning the data model, in this paper, we only deal with *valid time state relations* [22]. For the sake of simplicity, we disallow *value-equivalent* tuples (tuples with mutually identical nontemporal attributes), as in TSQL2 and BCDM.

We term these relations *atelic relations* and term tuples in such relations *atelic tuples*, since they represent atelic facts. For instance, the STOCK^A relation shown in Table 1 is an atelic relation representing stocks, their category, and their price over time (here, and in the following tables, we give the time in minutes for a specific hour). For example, the first tuple represents the fact that IBM had a price of \$63 for four minutes (from 14 through 17).

3.2 Impact of Aktionsart Distinctions

The treatment of telic facts in a temporal relational algebra based on atelic elements (i.e., on a point-based semantics for data; see Section 2.1) encounters several problems. We illustrate these problems with an example, but we stress that the same problem arises whenever an atelic relation (i.e., a relation based on point-based semantics) is used to represent telic facts.

A phone call starting at time t_1 and ending at time t_2 is a durative telic fact. For instance, if John made a call to Mary from time 10 to time 12, he didn't make it from 10 to 11. Similarly, two consecutive calls, one from 10 to 12 (inclusive) and the other from 13 to 15 (inclusive), are clearly different from a single call from 10 to 15. For example, in the current Italian phone system, the cost of a call is the sum of a fixed amount, which has to be paid for each call (independently of its duration) plus a variable amount, depending on many factors, including duration and distance. Thus, one call from John to Mary starting at 10 and ending at 15 is cheaper than two consecutive calls taking the same amount of time. This implies that phone calls are telic facts.

Suppose that we use an *atelic* relation to represent telic facts such as phone calls. In particular, let us model the fact that John called Mary twice: once from 10 to 12 and once from 13 to 15 (inclusive). The linguistic analysis (see Section 2.2) tells us that telic facts cannot be correctly captured using models that are based on a point-based semantics for data since such models are not sufficiently

TABLE 2
PHONE^A Relation

Caller	Called	VT
John	Mary	{10,11,12,13,14,15}

TABLE 3
$PHONE2^A$ Relation

Caller	Called	VT
John	Mary	{[10–12],[13–15]}

expressive. Consider the atelic PHONE^A relation shown in Table 2.

The point-based semantics of this relation is given in Fig. 1.

This atelic PHONE^A relation captures the fact that John was calling Mary at 10, 11, 12, 13, 14, and 15. This relation does not capture relevant information, namely, the fact that there were two distinct calls, one from 10 to 12 and the other from 13 to 15.

3.3 Representation Language versus Data Semantics

It is important to emphasize that the above loss of information is *not* due to the fact that the *language* we use to represent data does not support time intervals, but to the fact that the underlying *data semantics* is point-based. In other words, whenever the data semantics is point-based (independently of how the data language looks), we risk a loss of information when dealing with telic facts. This is true for all approaches that use temporal elements to timestamp tuples; considerSQL/Temporal, TSQL2, TSQL, HQL, and TQuel, which utilize temporal elements to timestamp tuples, and Gadia's [19] Homogeneous Relational Model, which uses temporal elements to timestamp attributes.

While temporal elements are sets of intervals, this is only a matter of data *representation language* since the underlying data *semantics* is point-based. In other words, temporal elements are merely a notational representation for a set of time points. In such approaches, one can represent the above phone example as in relation $\mathrm{PHONE2}^A$ (see Table 3).

However, while $\mathrm{PHONE2}^A$ is different from PHONE^A at the level of the representation language, it is identical to PHONE^A at the data semantics level since both relations represent the identical content shown in Fig. 1, namely, that John was calling Mary at 10, 11, 12, 13, 14, and 15 (formally, PHONE^A and $\mathrm{PHONE2}^A$ are *snapshot equivalent* [23]). Thus, the loss of information is the *same* even if we adopt *temporal elements* [19] instead of atelic elements.

To summarize, one central message of this paper is that, independently of whether the data *language* supports or not time intervals, if the underlying data *semantics* is point-based, we do not have sufficient expressive power to properly deal with the semantics of telic facts. Obviously, this lack of expressive power also has a dramatic impact on the results one may obtain when querying relations based on the point-based semantics, as we will now examine.

$10 \rightarrow$ {<John, Mary>} $11 \rightarrow$ {<John, Mary>} $12 \rightarrow$ {<John, Mary>}

$13 \rightarrow$ {<John, Mary>} $14 \rightarrow$ {<John, Mary>} $15 \rightarrow$ {<John, Mary>}

Fig. 1. Point-based semantics for the table PHONE^A.

3.4 Atelic Relational Algebra

We use a slight adaptation of the algebra of SQL/Temporal to exemplify atelic operators. Our operators σ, π^A, $\times^A$, $-^A$, and $\cup^A$ generalize the standard set operators (selection, projection, Cartesian product, difference, and union) to apply over atelic data. Since our definitions are quite standard with respect to the ones in the (temporal algebræ in the) literature, in the following, we only describe Cartesian Product (see [40] for the other definitions). We denote with $Sch(R^A)$ the data attributes of an atelic relation R^A. Given a tuple t of a relation R, we denote by $t(R)$ the value of the data attributes in t and, by $t(\mathrm{VT})$, its validity time.

3.4.1 Cartesian Product

The state-by-state interpretation above for atelic operators dictates that we adopt the intersection semantics for the atelic Cartesian product: The validity time of the resulting relation is the intersection of the validity times of the tuples.

$$Sch(R_1^A \times^A R_2^A) \equiv Sch(R_1^A) \cup Sch(R_2^A)$$
$$R_1^A \times^A R_2^A \equiv \{s | \exists t_1 \in R_1^A \; \exists t_2 \in R_2^A \; (s(R_1^A) = t_1(R_1^A) \wedge s(R_2^A) = t_2(R_2^A) \wedge s(\mathrm{VT}) = t_1(\mathrm{VT}) \cap t_2(\mathrm{VT})) \wedge s(VT) \neq \emptyset\}.$$

We also define an additional atelic operator, an adaptation of Gadia's temporal selection operator [19] (henceforth termed temporal restriction) to tuple timestamping.

3.4.2 Temporal Restriction

This operator restricts the validity time of the tuple to the time interval I. Note that the notation $\{I\}$ indicates the set of points contained in the interval I.

$$Sch(Restr_{[I]}^A(R^A)) \equiv Sch(R^A)$$
$$Restr_{[I]}^A(R^A) \equiv \{s | (\exists t \in R^A (s(R^A) = t(R^A) \wedge s(\mathrm{VT}) = t(\mathrm{VT}) \cap \{I\}) \wedge s(\mathrm{VT}) \neq \emptyset.\}$$

Temporal operators such as temporal selection that have explicit predicates that treat the timestamps as intervals are *not* included within our atelic algebra since they essentially have an interval-based semantics. Consider, e.g., temporal selection based on duration (e.g., "select all phone calls that lasted at least five units"). To determine the duration of a fact in an atelic model, one has to collect all the consecutive time points in which the fact holds in order to determine the maximal convex time interval(s) covering exactly such points. Then, the durations of such intervals are considered. Thus, there is an (implicit) shift from time points to time intervals. In our approach, we make such a shift explicit by supporting temporal selection operators only in the telic algebra (which operates on a data model based on interval-based semantics; see Section 4).

TABLE 4
PHONE^T Relation

Caller	Called	VT
John	Mary	{ [10–12], [13–15] }

[10–12] → {<John, Mary>} [13–15] → {<John, Mary>}

Fig. 2. Interval-based semantics for the relation PHONE^T.

4 Telic Relational Model and Algebra

4.1 Telic Elements

In order to cope with the temporal issues discussed in Section 3.2, we need to explicitly introduce the notion of *time intervals*. A time interval i is a convex set of time points between a starting point i^- and an ending point i^+, that is, $\forall p\ p \in \mathcal{P}(i^- \leq p \leq i^+ \Rightarrow\ p \in I\ )$, where $i^- \leq i^+$.

We indicate the domain of time intervals by I. Moreover, we consider sets of time intervals (i.e., subsets of 2^I), termed *telic elements*. It is important to notice that, although we define time intervals as sets of time points, we regard them as *atomic* objects. Thus, a telic element may also contain meeting or overlapping intervals (e.g., {[10-15], [13-20]} is a reasonable telic element).

Two functions map atelic elements to telic elements and vice versa. *to-atelic* takes in input a telic element TE (i.e., a set of time intervals) and gives as output the atelic elements containing all the points belonging to the intervals in TE. *to-telic*, in contrast, takes as input an atelic element (a set of time points) and gives as output the telic elements containing the maximal convex intervals that cover exactly the time points in the atelic element.

$$\textit{to-atelic}(\{[12-16],[15-17],[20-21]\}) = \{12,13,14,15,16,17,20,21\}$$
$$\textit{to-telic}(\{12,13,14,15,16,17,20,21\}) = \{[12-17],[20-21]\}.$$

Note that, given any atelic element $A \in 2^P$,

$$\textit{to-atelic}(\textit{to-telic}(A)) = A.$$

However, given a telic element $T \in 2^I$, it may be the case that $\textit{to-telic}(\textit{to-atelic}(T)) \neq T$.

We regard time intervals as indivisible primitive entities and we use the operations of union, complement, and intersection over telic elements as the *application of standard set operators on the domain of time intervals*. For example,

$$\{[10-15],[20-30]\} \cup \{[10-25],[35-40]\} = \{[10-15],[20-30],[10-25],[35-40]\}$$
$$\{[10-15],[20-30] \cap \{[10-25],[20-30],[35-40]\} = \{[20-30]\}.$$

Notice that temporal coercion is not performed over time intervals by union so that upward hereditary is not forced.

4.2 Telic Model

As discussed in Section 3, associating a tuple t with an *atelic* element $\{p_1,\ldots,p_k\}$ means that the fact represented by the tuple *t holds* over all time points $p_1,\ldots,p_k$, thus utilizing the point-based semantics. On the other hand, time intervals (and *telic* elements) are introduced in order to represent that the fact described by a tuple t is *accomplished* in a given time interval i, i.e., t starts at i^- and finishes (reaches its *culmination* or *telos*) at i^+. For instance, in the phone call example, a tuple $< \text{John}, \text{Mary}, \{[10-12]\} >$ means that John's call to Mary started at time 10 and finished at 12. Notice that, although it is true that John *was calling* Mary within any time point contained in [10-12], it would not be correct to state that John *accomplished* such a call at 11; downward hereditary does not hold. Analogously, upward hereditary does not hold when a telic interval is associated with a tuple. For instance, the tuple $< \text{John}, \text{Mary}, \{[10-12],[13-15]\}$ represents two different episodes of John calling Mary, not to be confused or merged together.

In our telic relational model, a *telic relation* is a set of *telic tuples*, each with a validity time, which is a *telic element*. For the sake of simplicity, we do not admit value-equivalent tuples, similarly to the atelic model.

4.3 Impact of Aktionsart on Telic Relations

We now show that, adopting an interval-based semantics (i.e., associating tuples with telic elements), one can correctly capture the meaning of telic facts into relational relations. Let us consider again, e.g., the phone example discussed in Section 3.2. Instead of using an atelic relation (cf., the PHONE^A relation), let us now use a telic relation (say PHONE^T) shown in Table 4.

PHONE^T is a telic relation so that the validity time is a telic element, and interval-based semantics is used. This means that the time intervals [10-12] and [13-15] are interpreted as atomic temporal entities, and the semantics of PHONE^T can be expressed as in Fig. 2 (contrast with Fig. 1).

Thus, the telic relation PHONE^T correctly models the information that there were two episodes of John calling Mary, one that occurred from 10 to 12 and the other from 13 to 15.

More generally, telic elements (and interval-based semantics) are sufficiently expressive to model the semantics of telic facts since they do not lose information concerning the different episodes, even in case such episodes overlap or meet in time. This fact is also evident when asking queries to telic relations.

4.4 Telic Algebra

Now, we can define the operators on telic relations. The rationale underlying all the definitions is the following: In the interval semantics, each tuple occurs exactly in the time intervals in its validity time, and nowhere else. Notice that, in the definitions of the telic algebraic operators, set operators apply to telic elements (i.e., set of time intervals), while, in the atelic algebra, they operate on atelic elements (i.e., set of points).

Some of the telic operators have an identical definition as their atelic counterpart. In particular, (nontemporal) selection, which applies to atelic relations, also works fine on telic relations. The formal definition of telic union ($\cup^T$) is similar to the definition of atelic union ($\cup$), with the added behavior that the telic union of two value-equivalent telic

John made two calls to Mary, one from 10 to 12, and the other from 13 to 15;

Sue made two calls to Ann, one from 12 to 14, the other from 15 to 16;

Eric made a call to Paul from 14 to 16.

Fig. 3. Sample temporal data.

tuples t_1 and t_2 is a single tuple with a valid timestamp of the set union ($\cup$) of the two underlying telic elements $t_1(\mathrm{VT})$ and $t_2(\mathrm{VT})$. This similarity also applies to atelic (π^A_B) and telic (π^T_B) projection operators.

4.4.1 Cartesian Product

Atelic Cartesian product necessarily requires that the two timestamps overlap (equivalently, that the two sets of time points are not disjoint), for each pair of tuples, effecting a point-based interpretation of facts. From a theoretical view, we could define telic Cartesian product similarly, but this would retain intervals only if they matched exactly, which seems artificial. The atelic Cartesian product can be seen as the counterpart of the "while" adverb in the temporal algebra and "while" involve an atelic view of the facts to which it applies [26]. Since forcing intersection is unnatural, we instead allow any of the Allen [1] predicates and make this predicate explicit in the operator, which can then be seen as the counterpart of the "before" or "meets," etc. adverbs.

There is also the issue of what timestamp to associate with the resulting tuples. For atelic Cartesian product, this decision necessarily is to intersect the two timestamps to determine the resulting timestamp, which doesn't work here. So, we simply chose to return the timestamp of the left tuple. In the following, ϕ is a binary Allen predicate [1]:

$$
\begin{aligned}
&Sch(R^A_1 \times^A_\phi R^A_2) \equiv Sch(R^A_1) \cup Sch(R^A_2)\\
&R^T_1 \times^T_\phi R^T_2 \equiv \{s | \exists t_1 \in R^T_1\ \exists t_2 \in R^T_2 (s(R^T_1) =\\
&\quad t_1(R^T_1) \wedge s(R^T_2) = t_2(R^T_2) \wedge\\
&\quad \phi(t_1(\mathrm{VT}), t_2(\mathrm{VT})) \wedge s(\mathrm{VT}) = t_1(\mathrm{VT})\}.
\end{aligned}
$$

Analogously, set difference between telic elements is artificial since only repeated intervals will be affected. For this reason, we do not include a telic difference operator. Temporal restriction on telic relations also does not make sense because the entire interval must be retained.

4.4.2 Temporal Selection

Unlike conventional selection, temporal selection takes a temporal predicate, ϕ, on telic intervals; examples of such predicates include duration and comparison with constants (see Section 4.5).

$$
\begin{aligned}
&Sch(\sigma^T_\phi(R^T)) \equiv Sch(R^T)\\
&\sigma^T_\phi(R^T) \equiv \{s | \exists t \in R^T (s(R^T) = t(R^T) \wedge s(\mathrm{VT}) = \mathrm{VT}') \wedge\\
&s(\mathrm{VT}) \neq \emptyset\},
\end{aligned}
$$

where $\mathrm{VT}' = \{i \in t(\mathrm{VT}) |\ \phi(i)\}$.

4.5 Impact of Aktionsart Distinctions

We again consider the phone example. Suppose that we want to store the data in Fig. 3.

We use a telic relation PHONE^T to represent such data, and also consider the corresponding atelic relation PHONE^A (see Table 5). Once again, notice that the distinction between telic and atelic relations is not due to the representation *language*, but to the (interval-based versus point-based) *data semantics*.

4.5.1 Downward Hereditary

Our first query is, "What are the complete phone calls within times 10 to 11?"

- (Q1): $Restr^A_{[10,11]}(\mathrm{PHONE}^A)$.

The answer to Q1 is the tuple $< \mathrm{John}, \mathrm{Mary} \,||\, \{10, 11\} >$. However, this is not a correct answer since downward hereditary do not hold for telic facts: From the fact that John made a *complete* phone call to Mary from 10 to 12, it is wrong to conclude that he makes such a *complete* call from 10 to 11 (at most, one could conclude that John was calling Mary at 10 and 11, but not that he did a *complete* call at that time). This fact is captured if a telic relation is used since the restriction operator cannot be applied to a telic relation.

4.5.2 Upward Hereditary

The treatment of upward hereditary causes even more severe problems to the point-based semantics. To illustrate, we provisionally apply temporal selection also to atelic relations, even if, in our approach, temporal selection only applies to telic ones. Notice that, actually, temporal selection is used in most temporal algebræ in the literature, even if they employ a point-based semantics in their data model. In Section 5, we will propose a clean solution to this problem by introducing *coercion functions*.

Let us consider, for instance, the following query, asking for all information regarding phone calls lasting at least five units in the atelic relation PHONE^A.

- (Q2): $\sigma^T_{\mathrm{duration(VT)} \geq 5}(\mathrm{PHONE}^A)$.

The tuples $< \mathrm{John}, \mathrm{Mary} \,||\, \{10, 11, 12, 13, 14, 15\} >$ and

TABLE 5
PHONE^T and PHONE^A Relation

PHONE^T

Caller	Called	VT
John	Mary	{ [10–12], [13–15] }
Sue	Ann	{ [12–14], [15–16] }
Eric	Paul	{ [14–16] }

PHONE^A

Caller	Called	VT
John	Mary	{10,11,12,13,14,15}
Sue	Ann	{12,13,14,15,16}
Eric	Paul	{14,15,16}

$$< \text{Sue}, \text{Ann}, \{12, 13, 14, 15, 16\} >$$

are given as output. This is due to the fact that, in the atelic relation, we have a loss of information concerning the start and end points of calls since consecutive calls are "coalesced together." On the other hand, none of the calls in Fig. 3 lasts at least five units.

Let us suppose now that the above query is applied to the telic relation PHONE^T.

- (Q2′): $\sigma^T_{\text{duration(VT)}\geq 5}(\text{PHONE}^T)$.

In such a case, no tuple is given as output, consistent with the data in Fig. 3.

We have an analogous situation if we ask for information concerning phone calls that, e.g., follow one call from John to Mary.

- (Q3):

$$\text{PHONE}^A \times^T_{\text{After}} (\sigma_{\text{caller=John,called=Mary}}(\text{PHONE}^A)).$$

- (Q3′):

$$\text{PHONE}^T \times^T_{\text{After}} (\sigma_{\text{caller=John,called=Mary}}(\text{PHONE}^T)).$$

The answer to (Q3) is the empty relation, while the answer to (Q3′) is the relation containing the tuples $< \text{Sue}, \text{Ann} \,||\, \{[15-16]\} >$ and $< \text{Eric}, \text{Paul} \,||\, \{[14-16]]\} >$, as desired.

In general, considering the benchmark for temporal query languages in [21], we see that analogous problems arise for all *interval queries* (i.e., queries asking about durations, endpoints, and relative positions of the validity times of tuples) whenever relations using point-based semantics for data are used in order to represent some telic fact.

5 An Integrated, Three-Sorted Algebra

We now show some examples motivating the fact that both telic models and operators and atelic ones are needed, as well as a flexible way of coercing telic relations to and from atelic ones.

5.1 The Telic Model and Algebra Are Not Adequate for Atelic Facts

The telic model and algebra in Section 4, taken in isolation, are not powerful enough to deal with all types of facts, in particular, atelic facts. In Section 3, we argued that most data models are atelic and that a telic data model is needed. Here, we argue the reverse, that an *atelic* data model is also needed. In fact, both kinds of data must be expressible in a temporal model.

Using a solely telic model and algebra to deal with atelic facts such as earning a given salary, owning a house, and so on, generate exactly the dual of the problems discussed in Sections 3.2 and 4.5. Both downward and upward hereditary properties hold for atelic facts; not considering them causes loss of information. Consider, e.g., the telic relation STOCK^T (see Table 6), which is the telic "counterpart" of the atelic relation STOCK^A in Section 3.1, using *telic elements*.

TABLE 6
STOCK^T Relation

Name	Category	Price	VT
IBM	High-tech	63	{ [14–17] }
IBM	High-tech	61	{ [12–13] }
GM	Industrial	49	{ [10–12], [16–18] }

Notice that it is part of the intended meaning of "stock prices" that stating that IBM price was more than 60 on [12-13], and then on [14-17] implies that it was more than 60 from 12 to 17 (upward hereditary); moreover, from the fact that the price of IBM was 63 from 14 to 17, one may correctly infer that it was 60 from 15 to 16 (downward hereditary). These semantic assumptions are automatically captured if the stock data are represented by an atelic relation (i.e., by a relation based on a point-based semantics for data). On the other hand, such assumptions no longer hold in case a telic relation (i.e., a relation based on interval-based semantics for data) such as STOCK^T is used to represent the same data. This loss of information becomes even more evident if we ask queries on STOCK^T. For example, restriction cannot be applied to telic relations, so that we cannot enforce downward hereditary. However, restriction on the atelic relation STOCK^A gives the (desired) result $\{< \text{IBM}, \text{high-tech}, 63 \,||\, \{15, 16\} >\}$.

- (Q4): $Restr^A_{[15,16]}(\sigma_{\text{Name=IBM}}(\text{STOCK}^A))$.

5.2 The Algebra

We have three sorts of temporal relations: atelic relations, in which the validity times of tuples are atelic elements (cf. Section 3.1), telic relations, in which the validity times of tuples are telic elements (cf. Section 4), plus standard nontemporal relations. For example, both the atelic relation STOCK^A in Section 3.1 and the telic relation PHONE^T in Section 4.5 may be present in a temporal database. Our algebra is a three-sorted one consisting of the atelic operators in Section 3.4, the telic operators in Section 4.4, plus the standard operators for the nontemporal algebra.

We introduce three temporal functions, τ^A_p, $Transform^T_I$, and $Transform^A_p$, which allow one to obtain the conventional (i.e., nontemporal) relation corresponding to a temporal relation, and vice versa. Notice that we chose to define τ^A_p only on atelic relations since, by definition, facts in telic relations only occur over time intervals.

Definition.

$$\tau^A_p(R) \equiv \{s | \exists t \in R \, \exists i \in t(\text{VT})(p \in i \wedge s(R) = t(R))\}.$$

The $Transform^A$ (respectively, $Transform^T$) function applies to an nontemporal relation R and a given time point p (respectively, a time interval I) and returns an atelic (respectively, telic) relation, with the timestamp of every tuple of p (respectively, I).

Definition.

$$Transform_I^T(R) \equiv \{s|\exists t \in R(s(R) = t(R) \wedge s(\mathrm{VT}) = \{I\})\}$$
$$Transform_p^A(R) \equiv \{s|\exists t \in R(s(R) = t(R) \wedge s(\mathrm{VT}) = \{p\})\}.$$

5.3 Need for Further Flexibility: Telic/Atelic Coercion Functions

Transformation functions that coerce telic relations into atelic ones, and vice versa, are needed, for two reasons. First, there should be a way to apply temporal selection (and, in general, any "interval query" [21]) to atelic relations, as most temporal algebræ do. Second, linguistics tells us that the distinction between telic and atelic facts is not at all a rigid one within natural languages [14], [40], [47]. The same flexibility would be greatly desirable in our temporal model and would significantly increase the expressive power of our approach.

Consider again, for instance, our phone call example (cf. relation PHONE^T) and the query

1. "Who made calls during John's calls to Mary?"

As shown in Section 4.3, phone calls are telic facts. However, when stating "*during* John's calls to Mary," we look inside the fact, *coercing* it into an atelic one. Thus, this query involves two different ways of looking at the relation PHONE^T. First, John's calls to Mary are interpreted as atelic facts since we are not looking for calls occurring during *one* of John's calls, but, more generally, *while John was calling Mary* (i.e., we ask for calls that have occurred during [10-15], and not during one of [10-12], [13-15]). Second, the calls we are asking for are interpreted as telic facts since we look for *each complete occurrence* of them which is fully contained in [10-15]. For example, we want Sue in our output since Sue made a call in [12-14], which is during [10-15], regardless of the fact that Sue also made another consecutive call from 15 to 16.

We thus need more flexibility: Although each base relation must be declared as telic or atelic, we need coercion functions to allow the temporal counterpart of linguistic aktionsart coercion [26]. These coercion functions $\alpha^T(\ )$ and $\alpha^A(\ )$ can be easily given in terms of the functions *to-telic* and *to-atelic* provided in Section 4.1. Given any telic relation R^T,

$$Sch(\alpha^T(R^T)) \equiv Sch(R^T)$$
$$\alpha^T(R^T) \equiv \{s|\exists t_1 \in R^T(s(R^T) = t_1(R_1^T) \wedge s(\mathrm{VT}) = to\text{-}atelic(t_1(\mathrm{VT})))\}$$

and any atelic relation R^A,

$$Sch(\alpha^A(R^A)) \equiv Sch(R^A)$$
$$\alpha^A(R^A) \equiv \{s|\exists t_1 \in R^A(s(R^A) = t_1(R_1^A) \wedge s(\mathrm{VT}) = to\text{-}telic(t_1(\mathrm{VT}))).\}$$

For example, in our approach, query 1 above can be expressed as shown in (Q5):

- (Q5):

$$\pi^T_{\mathrm{Caller}}(\mathrm{PHONE}^T \times^T_{\mathrm{During}} (\alpha^A(\sigma_{\mathrm{Caller=John,Called=Mary}}(\alpha^T(\mathrm{PHONE}^T))))).$$

5.4 Examples

In the following examples, we assume that the temporal database contains the atelic relation STOCK^A (Section 3.1) and the telic relation PHONE^T (Section 4.5).

The first two queries were given earlier. "Who made phone calls to whom during 10-11?"

- (Q1): $Restr^A_{[10,11]}(\alpha^T(\mathrm{PHONE}^T))$.

"Who made phone calls after John called Mary?" Here, "after" treats the phone calls as telic.

- (Q3′):

$$\mathrm{PHONE}^T \times^T_{\mathrm{After}} (\sigma_{\mathrm{caller=John,called=Mary}}(\mathrm{PHONE}^T)).$$

"Who made at least one complete call (during the time) when IBM's price was more than \$60?"

- (Q6):

$$\pi^T_{\mathrm{Caller}}(\mathrm{PHONE}^T \times^T_{\mathrm{During}} \alpha^A(\pi^A_{\mathrm{Name}}(\sigma^A_{\mathrm{Name=IBM,Price>60}}(\mathrm{STOCK}^A)))).$$

Here, PHONE^T is already telic, but STOCK^A needs to be converted, but only after the select and project operators. The *telic* result is:

$$\{< \mathrm{John} \,||\, \{[13-15]\} >, < \mathrm{Sue} \,||\, [12-14], [15-16] >, < \mathrm{Eric} \,||\, \{[14-16]\} >\}.$$

The intermediate steps in the evaluation of (Q6) are graphically shown in Fig. 4.

This example shows the importance of having both telic relations (in this case: PHONE^T), atelic ones (in this case: STOCK^A), and temporal coercion. Notice that, since STOCK^A is an atelic relation, the projection

$$\pi^A_{\mathrm{Name}}(\sigma_{\mathrm{Name=IBM,Price>60}}(\mathrm{STOCK}^A))$$

makes the point-union of the validity times of value-equivalent tuples. In other words, temporal coalescing [6], [10], [45] is applied so that the (atelic) tuple

$$< \mathrm{IBM}, \{12, 13, 14, 15, 16, 17\} >$$

is returned as the result of the projection. Thus, the tuple $< \mathrm{John} \,||\, \{[13-15]\} >$ is included in the final result, even though [13-15] is not included in any of the time intervals in which IBM was more than \$60 (i.e., [12-13], [14-17]) considered in isolation. On the other hand, since PHONE^T is telic, no temporal coercion is performed on the validity time of the tuple $< \mathrm{John, Mary} \,||\, \{[10-12], [13-15]\} >$. Thus, the tuple $< \mathrm{John} \,||\, \{[13-15]\} >$ is in the final result, as desired. Notice that coercion α^A must be used to apply the interval query "during" (i.e., $\times^T_{\mathrm{During}}$) to the atelic relation $\pi^A_{\mathrm{Name}}(\sigma_{\mathrm{Name=IBM,Price>60}}(\mathrm{STOCK}^A))$. Finally, using

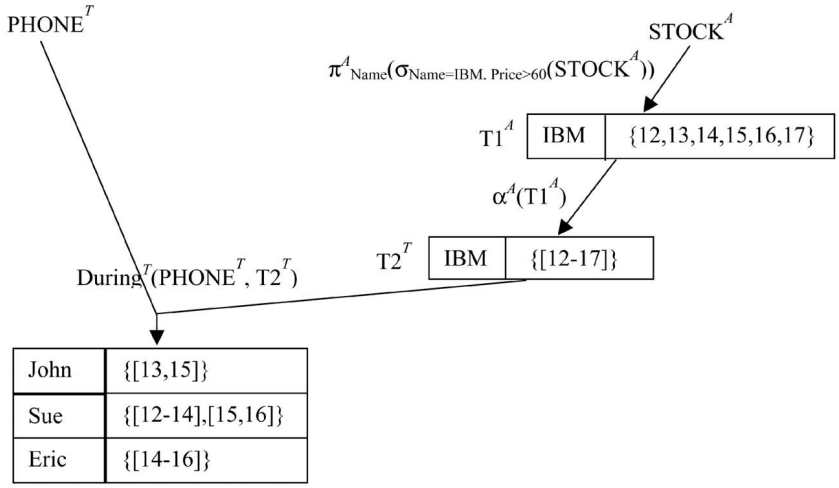


Fig. 4. Evaluation of the query Q6.

telic Cartesian product, we return the time intervals of phone calls as the validity time of the result.

"Tell me about stocks whose price was \$61 during the time when John was calling Mary."

- (Q7):

$$\alpha^A(\sigma_{\text{Price}=61}(\text{STOCK}^A)) \times^T_{\text{During}} \alpha^A(\alpha^T(\sigma_{\text{Caller}=\text{John},\text{Called}=\text{Mary}}(\text{PHONE}^T))).$$

This query shows the need of coercion from telic to atelic. In fact, we have an inner view ("was calling") of a telic relation (PHONE^T). Since PHONE^T is telic, the result of $\sigma_{\text{Caller}=\text{John},\text{Called}=\text{Mary}}(\text{PHONE}^T)$ is the telic relation

$$\{< \text{John}, \text{Mary} \parallel \{[10-12], [13-15]\} >\}.$$

However, in (Q7), we are not interested distinguishing different occurrences of John's calls to Mary, but we see calling Mary as an atelic fact, looking for the overall time in which John was calling her. We thus apply the α^T coercion operator, obtaining the atelic tuple

$$\{< \text{John}, \text{Mary}\{10, 11, 12, 13, 14, 15\} >\}.$$

Thus, the tuple $< \text{IBM}, \text{High-tech} \parallel \{[12-13]\} >$ is in the result even if IBM's price was \$61 in the time interval [12-13], which is not contained in any of John's calls taken in isolation. Notice that, since the $\times^T_{\text{During}}$ operator only applies to telic tuples, we need the coercions α^A to both sides of the operator. The telic result is thus:

$$\{< \text{IBM}, \text{High-tech} \parallel \{[12-13]\} >\}.$$

5.5 Properties of the Three-Sorted Algebra

Reduction and equivalence are important properties for a temporal algebra since they grant that a temporal algebra is a consistent extension of the nontemporal classical one. We state both properties for our three-sorted algebra here.

Property 1. *The atelic operators $\times^A$, π^A, and $-^A$ are* snapshot reducible *[33] to the analogous conventional relational algebraic operators. As for one case, π^A, snapshot reducibility is stated as $\forall R^A \forall p(\tau_p^A(\pi^A(R^A)) = \pi(\tau_p^A(R^A)))$.*

Property 2. *The equivalence property holds for the atelic operators $\times^A$, π^A, and $-^A$ and for the telic operators $\cup^T$ and π^T. As for one case, π^A, equivalence is stated as*

$$\forall R^A \forall p(Transform(\pi^A(R^A), p) = \pi^A(Transform(R^A, p))).$$

Moreover, our three-sorted algebra is more expressive than just the atelic one. In fact, Property 3 states that, if the database consists of both telic and atelic relations and one coerces all telic relations into atelic ones, the results obtained from queries can be different from those obtained distinguishing between telic and atelic relations.

Property 3. *Given a telic relation R^T and denoting by Op^A and Op^T the analogous (unary) atelic and telic operators, we have $\exists R^T(Op^A(\alpha^T(R^T)) \neq \alpha^T(Op^T(R^T)))$.*

6 Alternative and Related Approaches

Our approach provides a general solution to the difficulties of mixing point-based and interval-based data. However, another tack is to see if existing language facilities can be exploited to solve the same problem. In this section, we briefly examine some of these alternatives.

6.1 Alternative Approaches

Let us suppose to impose a *temporal first normal form* (1NF) [19] as, e.g., in TSQL and in HQuel, so that just one time interval is associated with each tuple, instead of a temporal element. In such approaches, e.g., relation PHONE^T could be represented by relation PHONE^{1NF} in Table 7.

However, if one adopts the point-based semantics for data, this transformation alone does not solve the problem. For example, the first two tuples of PHONE^{1NF} still carry on the content that John was calling Mary on 10, 11, 12, 13, 14, and 15. On the other hand, approaches that use 1NF as above and *never* perform coalescing of value-equivalent tuples, as in SQL/Temporal, exhibit the same kind of problems discussed in Section 5.1, since upward and downward hereditary

TABLE 7
PHONE^{1NF} and PHONE^{ATTR} Relation

PHONE^{1NF}

Caller	Called	VT
John	Mary	[10–12]
John	Mary	[13–15]
Sue	Ann	[12–14]
Sue	Ann	[15–16]
Eric	Paul	[14–16]

PHONE^{ATTR}

Caller	Called	Seq	VT
John	Mary	1	[10–12]
John	Mary	2	[13–15]
Sue	Ann	1	[12–14]
Sue	Ann	2	[15–16]
Eric	Paul	1	[14–16]

would never hold. Basically, any "homogeneous" approach in which upward and downward hereditary hold on all relations (as in TSQL2) or do not hold in any relation (as in SQL/Temporal) will not be satisfactory. Neither will approaches that have to fix a priori on which relations/attributes coalescing has to be performed and on which not, with no possibility of changing this property at query time (cf., e.g., [4]).

Another approach is to design database schemas in such a way that no value-equivalent tuples occur in the base relations so that at least the problem of dealing with upward hereditary (temporal coalescing) vanishes. For instance, in our phone-call example, one could add an additional attribute containing the sequence number of calls for each phone number to ensure that relations contain no value-equivalent tuples (see the table PHONE^{ATTR} in Table 7).

However, in the example above and in many practical cases, forcing that no value-equivalent tuples occur in base relations imposes a strong requirement on database designers, who need to introduce new attributes which are often scarcely useful and informative [7]. Moreover, the problems discussed in this paper would arise again for derived relations. For instance, if one projects away the additional Seq attribute from the relation PHONE^{ATTR}, one obtains again the original relation, with all the limitations discussed throughout this paper.

Finally, one could also introduce a surrogate attribute which is used to keep the identity of the fact (i.e., it is a key attribute) and is hidden to the user [23]. Notice that, in such a way, one models exactly the semantics of telic tuples since the surrogate attribute will represent explicitly the different "episodes" (occurrences) of a given telic fact. Thus, this solution can be simply conceived as a possible partial implementation of our telic model to avoid upward hereditary (or, in other words, to prevent temporal coalescing). However, such a partial implementation should be augmented to deal with the other aspects of our three-sorted algebra.

Temporal interpolation techniques, those that derive information for times for which no information is stored, on the basis of related information holding at different times [23], [31], could be useful. For example, Bettini et al. [4] proposed explicitly associating with each table a formal specification of the assumptions on the semantics of temporal attributes (e.g., persistence of data). At query time, such specifications are automatically merged with the user's query in order to provide the correct results. Bettini et al. also considered *interval assumptions*, including upward and downward hereditary which, however, are only studied in the context of evaluating the values of attributes whose validity time is expressed at different time granularities.

Finally, Chen and Zaniolo [10] use *aggregate functions*, such as `length` and `contains`, to perform telic operations on data assumed to be atelic. They also define the aggregate function `coales` to explicitly force upward inheritance on data assumed to be atelic.

In summary, this discussion of alternative solutions to the point/interval quandary examined five possibilities: INF, static declaration of point, or interval semantics for data, using additional attributes or surrogates, using temporal interpolation facilities, and using aggregate functions. In the first two cases, the problem was only partially solved. In the last three cases, it may be possible to express what is desired, but requires significant effort to fit this distinction into a formalism not designed with this purpose in mind. Instead, we feel that the atelic/telic distinction is so central that it should have a first-class status in both the data model and algebra.

6.2 The Point-Based versus Interval-Based Controversy Revisited

Recently, Toman [43, p. 212] pointed out some problems for the approaches where the validity times of tuples/attributes are encoded by time intervals (as, e.g., in TSQL2 and SQL/Temporal): "*this approach lead to a tension between the syntax of the query languages and their intended semantics: the data model and the semantics of the language are point-based, while temporal attributes refer to the actual encoding for sets of time intervals (e.g., interval endpoints)*." He proved that, for every point-based query, there is an equivalent interval-based query, and vice versa[3] [41], a statement that seems to fly in the face of the atelic/telic distinction. However, Toman always assumes a point-based semantics *for the data*. His "interval-based temporal database" is, in our terminology, a point-based semantics data model represented with an interval-based encoding. So, a rephrasing of that statement, in our terminology, is that, for every point-based query, there is an equivalent interval-based query over *point-based data* represented by an interval-based encoding, a statement which is consistent with the discussion in our paper.

Toman then proposed a new model and language which are more purely point-based; the only appearance of intervals is in the specific encoding used to implement this language; no notion of interval appears either in the data model nor in his SQL/TP (for time-point) query language [42], [43], [44]. We agree that having the interval-based

3. Specifically, Toman [41] in Theorem 5.5 restricted his attention to a particular form of interval query, which in our terminology correspond to atelic queries.

nature of the encoding appear in a restricted way in the query language raises problems; however, we feel that it is more appropriate to emphasize the distinction between atelic and telic facts, rather than jettisoning telic facts altogether, thereby allowing data with a semantics such as in Fig. 2 and queries such as atelic queries on telic data and telic queries on telic and atelic data (see Section 5.4).

Chomicki and Toman [12] clearly point out the distinction between abstract and concrete temporal databases and between data and query language. Furthermore, the framework for multidimensional time they introduce is a very general and powerful one and could be applied to model both atelic and telic elements. In particular, time points in our approach might correspond to their one-dimension points and our telic intervals to their two-dimensional points. On the other hand, they do not propose any specific treatment of the telic/atelic dichotomy so that, for instance, no counterpart of our coercion function is taken into account. Thus, we believe that the considerations we made in Section 6.1 also apply to their approach.

In ATSQL2 and when using temporal statement modifiers in general [8], the data semantics is purely point-based (atelic) and the query semantics is almost entirely atelic (except for duration and interval comparison predicates). However, there is some leeway in choosing a representation of the result as there are potentially many snapshot-equivalent representations of the result. Their notion of *interval preservation* selects among these representations that which best preserves the underlying intervals (this notion was first introduced by Böhlen et al. [7]). Through their *nonrestrictiveness* property, they allow the interval timestamps to be converted into values of an explicit attribute, thereby enabling interval-based queries to be simulated in conventional SQL, often with difficulty, as SQL has little notion of time.

7 Conclusions and Future Work

The analysis of the semantics of temporal data and queries plays a central role in the area of TDBs since "*data explicitly stored in a temporal database are often associated with certain semantic assumptions*" [4, p. 277]. Although many different models have been proposed in the TDB literature, almost all of them are based on a point-based (snapshot) semantics for the association of tuples (or attributes) to time. On the other hand, in the areas of linguistics and philosophy, many approaches showed that a point-based semantics is useful for atelic facts, but is not adequate to cope with telic facts for which an interval-based semantics is needed.

In this paper, we introduced a three-sorted sorted model and algebra which properly copes with both telic and atelic facts and which introduces coercion functions for transforming tables of one sort into tables of the other, at query time. Reduction and equivalence with respect to the classical nontemporal algebra hold for our temporal algebra.

Our approach makes the following main contributions:

1. We have clarified several subtle issues concerning the adoption of a point-based versus interval-based semantics, also making clearer the distinction between data language and data semantics and between query and data semantics.
2. We have proposed a data model and algebra that emphasizes the telic/atelic dichotomy, which is not dealt with in any other temporal database approach.
3. Elsewhere [40], we extend SQL/Temporal [35] in a minimal way to cope with the telic/atelic dichotomy by introducing the keyword "TELIC," which can be used for the creation of telic relations as well as in the queries to perform coercions.

We feel that the atelic/telic distinction is so central that it should be given first-class status in the data model and query language, especially as doing so requires so few changes to either. Moreover, we strongly agree with Moens and Steedman's claim that "*Effective exchange of information between people and machines is easier if the data structures that are used to organize the information in the machine correspond in a natural way to the conceptual structures people use to organize the same information*" [26, p. 26].

The approach in this paper can be easily extended in order to cope also with *validity time event relations* [34]. We also envision the possibility of extending our approach to deal with other relevant temporal properties, such as temporal persistence [4] and interpolation functions [31], [23], [4]. We want to extend these approaches to take into account the impact of aktionsart distinctions on the semantics of TDBs. Finally, we are also investigating the extensions needed in order to extend current temporal ER models to cope also with the telic/atelic distinction.

Acknowledgements

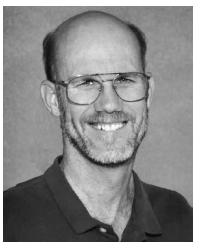
The authors would like to thank Christian S. Jensen for stimulating discussions. US National Science Foundation grants IIS-0100436 and EIA-008013 provided partial support of this research.

References

[1] J.F. Allen, "Maintaining Knowledge about Temporal Intervals," *Comm. ACM,* vol. 26, no. 11, pp. 832-843, 1983.
[2] Aristotle , *The Categories, on Interpretation. Prior Analytics.* Cambridge, Mass.: Harvard Univ. Press, 1938.
[3] M. Bennet and B. Partee, "Tense and Discourse Location in Situation Semantics," Indiana Univ. Linguistics Club, Bloomington, 1978.
[4] C. Bettini, X.S. Wang, and S. Jajodia, "Temporal Semantic Assumptions And Their Use in Databases," *IEEE Trans. Knowledge and Data Eng.,* vol. 10, no. 2, pp. 277-296, Mar./Apr. 1998.
[5] L. Bloom, L. Lifter, and J. Hafitz, "Semantics of Verbs and the Developments of Verb Inflection in Child Language," *Language,* vol. 52, no. 2, pp. 386-412, 1980.
[6] M. Böhlen, R.T. Snodgrass, and M. Soo, "Coalescing in Temporal Databases," *Proc. Int'l Conf. Very Large Databases,* pp. 180-191, 1996.
[7] M. Böhlen, R. Busatto, and C.S. Jensen, "Point- Versus Interval-Based Data Models," *Proc. IEEE Int'l Conf. Data Eng.,* pp. 192-200, 1998.
[8] M. Böhlen, C.S. Jensen, and R.T. Snodgrass, "Temporal Statement Modifiers," *ACM Trans. Database Systems,* vol. 25, no. 4, pp. 407-456, Dec. 2000.
[9] J.A. Bubenko, "The Temporal Dimension in Information Modelling," *Architecture and Models in Data Base Management Systems,* G. Nijssen, ed., North Holland, 1977.
[10] C.X. Chen and C. Zaniolo, "Universal Temporal Extensions for Database Languages," *Proc. IEEE Int'l Conf. Data Eng.,* pp. 428-437, 1999.
[11] J. Chomicki, "Temporal Query Languages: A Survey," *Proc. Int'l Conf. Temporal Logic,* pp. 506-534, 1994.
[12] J. Chomicki and D. Toman, "Temporal Logic in Information Systems," *Logics for Databases and Information Systems,* J. Chomicki and G. Saake, eds., chapter 3, pp. 31-70, 1998.

[13] J. Chomicki, D. Toman, and M.H. Böhlen, "Querying ATSQL Databases with Temporal Logic," *ACM Trans. Database Systems,* vol. 26, no. 2, pp. 145-178, 2001.
[14] *Computational Linguistics,* special issue on Tense and Aspect, B.L. Webber, guest ed., vol. 14, no. 2, 1988.
[15] J. Clifford and A.U. Tansel, "On an Algebra for Historical Relational Databases: Two Views," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* S. Navathe, ed., pp. 247-265, 1985.
[16] J. Clifford and D.S. Warren, "Formal Semantics for Time in Temporal Databases," *ACM Trans. Database Systems,* vol. 8, no. 2, pp. 214-254, 1983.
[17] D.R. Dowty, *World Meaning and Montague Grammar,* D. Riedel, ed. Dordrecht, 1979.
[18] D.R. Dowty, "The Effects of the Aspectual Class on the Temporal Structure of Discourse," *Tense and Aspect in Discourse, Linguistics and Philosophy,* vol. 9, no. 1, pp. 37-61, 1986.
[19] S.K. Gadia, "A Homogeneous Relational Model and Query Languages for Temporal Databases," *ACM Trans. Database Systems,* vol. 13, no. 4, pp. 418-448, 1988.
[20] *Proc. Int'l Workshop Infrastructure for Temporal Databases,* R.T. Snodgrass, ed., 1993.
[21] C.S. Jensen et al., "A Consensus Test Suite of Temporal Database Queries," *Proc. Int'l Workshop Infrastructure for Temporal Databases,* R.T. Snodgrass, ed., pp. QQ1-QQ28, June 1993.
[22] "A Consensus of Glossary of Temporal Database Concepts," *Temporal Databases: Research and Practice,* C.S. Jensen and C.E. Dyreson, eds., pp. 367-405, 1998.
[23] C.S. Jensen and R.T. Snodgrass, "Semantics of Time-Varying Information," *Information Systems,* vol. 21, no. 4, pp. 311-352, 1996.
[24] C.S. Jensen and R.T. Snodgrass, "Temporal Data Management," *IEEE Trans. Knowledge and Data Eng.,* vol. 11, no. 1, pp. 36-44, 1999.
[25] D.H.O. Ling and D.A. Bell, "Taxonomy of Time Models in Databases," *Information. Software Technology,* vol. 32, no. 3, pp. 215-224, 1990.
[26] M. Moens and M. Steedman, "Temporal Ontology and Temporal Reference," *Computational Linguistics,* vol. 14, no. 2, pp. 15-28, 1998.
[27] L.E. McKenzie and R.T. Snodgrass, "Evaluation of Relational Algebras Incorporating the Time Dimension in Databases," *ACM Computing Surveys,* vol. 23, no. 4, pp. 501-543, 1991.
[28] G. Özsoyoglu and R.T. Snodgrass, "Temporal and Real Time Databases: A Survey," *IEEE Trans. Data and Knowledge Eng.,* vol. 7, no. 4, pp. 513-532, Aug. 1995.
[29] "Recent Advances in Temporal Databases," *Proc. Int'l Workshop Temporal Databases,* J. Clifford and A. Tuzhilin, eds., 1995.
[30] F. Roddick and J.D. Patrick, "Temporal Semantics In Information Systems: A Survey," *Information Systems,* vol. 17, no. 3, pp. 249-267, 1992.
[31] A. Segev and A. Shoshani, "Logical Modelling of Temporal Data," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* U. Dayal and I. Traiger, eds., pp. 454-466, 1995.
[32] Y. Shoham, "Temporal Logics in AI: Semantical and Ontological Considerations," *Artificial Intelligence,* vol. 33, pp. 89-104, 1987.
[33] R.T. Snodgrass, "The Temporal Language TQuel," *ACM Trans. Database Systems,* vol. 12, no. 2, pp. 247-298, 1987.
[34] *The Temporal Query Language TSQL2,* R.T. Snodgrass, ed. Kluwer Academic, 1995.
[35] R.T. Snodgrass, M.H. Böhlen, C.S. Jensen, and A. Steiner, "Adding Valid Time—Part A," ANSI X3H2-95-485, ftp.cs.arizona.edu/tsql/tsql2/sql3, 1995.
[36] R.T. Snodgrass, M.H. Böhlen, C.S. Jensen, and A. Steiner, "Transitioning Temporal Support in TSQL2 to SQL3," *Temporal Databases: Research and Practice,* pp. 150-194, 1998.
[37] M.D. Soo, C.S. Jensen, and R.T. Snodgrass, "An Algebra for TSQL2," *The Temporal Query Language TSQL2,* R.T. Snodgrass, ed., chapter 27, Kluwer Academic, 1995.
[38] *Temporal Databases: Theory, Design and Implementation,* A. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R.T. Snodgrass, eds. Benjamin/Cummings, 1994.
[39] *Temporal Databases: Research and Practice,* O. Etzion, S. Jajodia, and S. Sripada, eds. Springer Verlag, 1998.
[40] P. Terenziani and R.T. Snodgrass, "Reconciling Point-Based and Interval-Based Semantics in Temporal Relational Databases: A Proper Treatment of the Telic/Atelic Distinction," TimeCenter Technical Report 60, June 2001.
[41] D. Toman, "Point vs. Interval-Based Query Languages for Temporal Databases," *Proc. ACM Symp. Principles of Database Systems,* pp. 58-67, June 1996.
[42] D. Toman, "Point-Based Temporal Extension of SQL," *Proc. Deductive and Object-Oriented Databases,* F. Bry, R. Ramakrishnan, and K. Ramamohanaroa, eds., pp. 103-121, Dec. 1997.
[43] D. Toman, "Point-Based Temporal Extensions of SQL and Their Efficient Implementation," *Temporal Databases: Research and Practice,* O. Etzion, S. Jajodia, and S. Sripada, eds., pp. 211-237, Springer Verlag, 1998.
[44] D. Toman, "SQL/TP: A Temporal Extension of SQL," *Constraint Databases,* G. Kuper, L. Libkin, and J. Paredaens, eds., chapter 19, pp. 391-399, 2000.
[45] C. Vassilakis, "An Optimisation Scheme for Coalesce/Valid Time Selection Operator Sequences," *SIGMOD Record,* vol. 29, no. 1, pp. 38-43, 2000.
[46] Z. Vendler, "Verbs and Times," *Linguistics in Philosophy,* pp. 97-121, New York: Cornell Univ. Press, 1967.
[47] H.J. Verkuyl, *On the Compositional Nature of the Aspects.* Dordhrect, Riedel, 1971.
[48] Y. Wu, S. Jajodia, and X.S. Wang, "Temporal Database Bibliography Update," *Temporal Databases: Research and Practice,* O. Etzion, S. Jajodia, and S. Sripada, eds., pp. 338-366, Springer Verlag, 1998.

Paolo Terenziani received both the Laurea (in 1987) and the PhD (in 1993) degrees in computer science from the Universitá di Torino, Italy. He is currently a full professor with the Dipartimento di Informatica at the Universitá del Piemonte Orientale "Amedeo Avogadro," Alessandria, Italy. His research interests mainly focus on the treatment of time and time-dependent phenomena in different areas, including databases (in the subareas of temporal relational databases, and of semantics) and artificial intelligence (natural language, knowledge representation, temporal reasoning, constraint propagation techniques). He has published more than 60 papers about these topics in refereed journals and conferences.



Richard T. Snodgrass received the BA degree in physics from Carleton College and the MS and PhD degrees in computer science from Carnegie Mellon University. He joined the University of Arizona in 1989, where he is a professor of computer science. He is an ACM fellow. He is editor-in-chief of the *ACM Transactions on Database Systems*. He chairs the ACM SIGMOD Advisory Board, serves on the editorial board of the *International Journal on Very Large Databases* and the Advisory Board of *ACM SIGMOD Digital Review* and has served on the editorial board of the *IEEE Transactions on Knowledge and Data Engineering*. He chaired the Americas program committee for the 2001 International Conference on Very Large Databases, chaired the program committees for the 1994 ACM SIGMOD Conference and the 1993 International Workshop on an Infrastructure for Temporal Databases, and was a vice-chair of the program committees for the 1993 and 1994 International Conferences on Data Engineering. He was ACM SIGMOD Chair from 1997 to 2001 and has previously chaired the ACM Publications Board and the ACM SIG Governing Board Portal Committee. He received the ACM SIGMOD Contributions Award in 2002. He chaired the TSQL2 Language Design Committee, edited the book, *The TSQL2 Temporal Query Language* (Kluwer Academic), authored *Developing Time-Oriented Database Applications in SQL* (Morgan Kaufmann), was a coauthor of *Advanced Database Systems* (Morgan Kaufmann), and was a coeditor of *Temporal Databases: Theory, Design, and Implementation* (Benjamin/Cummings). He codirects TimeCenter, an international center for the support of temporal database applications on traditional and emerging DBMS technologies. His research interests include temporal databases, query language design, query optimization and evaluation, storage structures, and database design. He is a senior member of the IEEE.