Temporal Specialization

Christian S. Jensen

Department of Mathematics and Computer Science
Aalborg University
Fredrik Bajers Vej 7E
DK-9220 Aalborg Ø, DENMARK
csj@iesd.auc.dk

Richard T. Snodgrass

Department of Computer Science
University of Arizona
Tucson, AZ 85721

rts@cs.arizona.edu

Abstract

In *temporal specialization*, the database designer restricts the relationship between the valid time-stamp (recording when something is true in the reality being modeled) and the transaction time-stamp (recording when a fact is stored in the database). An example is a *retroactive* temporal event relation, where the event must have occurred before it was stored, i.e., the valid time-stamp is restricted to be less than the transaction time-stamp. We discuss many useful restrictions, defining a large number of specialized types of temporal relations, and indicate some of their applications. We present a detailed taxonomy of specialized temporal relations. This taxonomy may be employed during database design to specify the particular time semantics of temporal relations.

1 Introduction

The time of validity of a fact in a temporal relation and the time the fact was recorded in the relation are ostensibly independent. Yet, in many applications of temporal relations, the two times interact in restricted ways. For example, in the monitoring of temperatures during a chemical experiment, temperature measurements are recorded in the temporal relation *after* they are valid, due to transmission delays. The resulting relation is termed *retroactive*. Alternatively, salary payments recorded in the temporal relation of a bank are recorded *before* the time the funds become accessible to employees, resulting in a *predictive* relation.

We explore a variety of temporal relations with specialized relationships between transaction and valid time. Such *specialized* temporal relations occur in many practical applications, and the framework presented here is a means of capturing more of the semantics of temporal relations, with two primary benefits. Used by designers and researchers, the framework conveys a more detailed understanding of temporal relations. The additional semantics, when captured by an appropriately extended database system, may be used for selecting appropriate storage structures, indexing techniques, and query processing strategies.

The paper extends a previously presented taxonomy on time in databases [SA85, SA86]. The previous taxonomy defined three kinds of time that could be associated with facts: *user-defined time* (with no database system-interpreted semantics), *valid time* (when a fact is true in reality), and *transaction time* (when a fact is stored in the database). A fact in a *temporal relation* has both a valid and a transaction time. A temporal relation records both the previous states of the relation and the history of reality. Temporal relations support three kinds of queries: (1) *current* queries, queries on the current state of the database; indeed, conventional database systems support only this kind of query; (2) *historical* queries, which extract facts about the history of objects from the modeled reality; and (3) *rollback* queries which extract facts as stored in the database at some point in the past. Though we use relational terminology throughout this paper, most of the analysis applies analogously to other data models.

The original taxonomy falls short in its characterization of temporal relations in three ways. First, the taxonomy fails to give an adequate understanding of some time-extended relations. Many proposals for adding time to databases advocate storing a single time-stamp per fact (e.g., [JMR91, SR85, SS87]), yet it appears that both rollback and historical queries are possible in these schemes. However, the taxonomy explicitly forbids both kinds of queries on a relation with only one time-stamp per tuple. Second, because the taxonomy focuses on the orthogonality of the three kinds of time, it ignores restricted interrelationships between the valid and transaction times of facts in temporal relations. In many practical applications, valid and transaction times of facts exhibit interrelationships. Third, the taxonomy assumes that each fact has at most one transaction time and one valid time time-stamp (interval or event). (From now on, we use the shorter, but not quite precise, terms 'valid time-stamp' and 'transaction time-stamp'.) However, in application systems with multiple, interconnected temporal relations, multiple time dimensions may be associated with facts as they flow from one temporal relation to another.

In order to address the first and second of the shortcomings, we explore the space of restricted interrelations—in-between the extremes of identity and no interrelation at all—that are possible between the valid and transaction times of facts. While we have focused primarily on comprehensiveness, we have not considered types of restricted interrelations that are of doubtful use. Addressing the third shortcoming, by providing the means for specifying the application system contexts of temporal relations, is the subject of a later paper.

We will not be concerned here with the semantics of time-varying attributes, i.e., how to use time-stamp values and stored attribute values to derive the value of a time-varying attribute. For example, we will not address the issues of how to derive the temperature of a chemical reaction at an arbitrary point in time from time-stamped and stored temperature measurements. We are interested only in the semantics of the time-stamps themselves.

In Section 2, we present a general definition and description of a temporal relation. In the following section, we examine the kinds of restrictions one might impose on temporal relations, considering in turn restrictions on isolated events, on collections of events, on isolated intervals, and on collections of intervals. The final section summarizes our work and points to future research.

2 Temporal Relations

We present a conceptual model of a temporal relation as a prelude to the extensions discussed in the remainder of the paper. A temporal relation has two orthogonal time dimensions, valid time and transaction time. *Valid time* is used for capturing the time-varying nature of the part of reality being modeled by the relation. *Transaction time* models the update activity of the relation. Thus, a temporal relation may be envisioned as a sequence of *historical states* indexed by transaction time.

A temporal relation consists of a set of *temporal elements*, each of which records one or more facts about an object (entity or relationship) from the part of reality being modeled by the temporal relation. Temporal elements have the following attribute values: element surrogate, object surrogate, transaction time-stamp, valid time-stamp (interval or event), time-invariant attribute values, time-varying attribute values, and user-defined times. We examine each briefly in turn.

An *element surrogate* is a system-generated, unique identifier of an element that can be referenced and compared for equality, but not displayed to the user [Dat85, HOT76]. We will discuss element surrogates in more detail shortly.

An *object surrogate* is a unique identifier of the object being modeled by an element. It is used for identifying all the database representations of individual real-world objects. At any point in time, each real-world object may have, in a single relation, a set of associated elements, all with the same object surrogate (c.f., a "life-line" [Sch77] or a "time sequence" [SK86]). Thus, a relation (c.f., a "time sequence collection" [SK86]) can be partitioned into a collection of sets so that elements of distinct sets have distinct object surrogates and elements of any single set have the same object surrogate. This is termed a *per surrogate* partitioning.

Transaction times are generated by the database system itself using monotonically increasing time-stamp generators; thus each historical state has an associated unique transaction time. The granularity of transaction time-stamps is arbitrary, as long as uniqueness is ensured. Transaction time models the update activity of the temporal relation, and as such, its semantics are entirely independent of the application and the enterprise being modeled. The transaction time of an element is the time when the facts recorded by the element were stored in the relation. Therefore, no stored transaction time exceeds the current time. The historical state resulting from a transaction remains unchanged from the time of that transaction to the time of the next transaction. Therefore, the semantics of transaction time have been characterized as stepwise constant. We will associate two transaction times, $tt_e^{\vdash}$ and $tt_e^{\dashv}$, with each element e in a temporal relation. The first, $tt_e^{\vdash}$, is the time when the element e is stored in the relation. The second, $tt_e^{\dashv}$, is the time when the element e is logically removed from the relation. The *existence interval* for e, $[tt_e^{\vdash}, tt_e^{\dashv})$, is thus the time between the transaction time of the historical state in which the element first appeared and the transaction time of the historical state succeeding the one in which the element last appeared.

The element surrogate identifies the element for the purpose of defining the existence interval (in the database) for the element. If a particular event or interval is (logically) deleted, then immediately re-inserted, the two resulting elements will have different element surrogates, allowing the deletion ($tt_e^{\dashv}$) and insertion ($tt_e^{\vdash}$) points to be unambiguously defined. If a modification is made by a transaction executed on the database, the element in the current historical state is (logically) deleted, and a new element, recording the modified information, is stored in the new historical state, indexed by the transaction time of the transaction making the change.

The database system uses the transaction times of elements for implementing the rollback operator [BZ82, Sch77]. In general, any domain of elements with an identity relation and a total ordering is suitable for transaction time. Example domains include the natural numbers and regular date/time values.

Valid times are usually supplied by the user, but they may be system-generated. The valid time-stamp of an element records when the facts represented by the time-varying (and time-invariant) and user-defined time attribute values are true in reality. Valid times are always drawn from the domain of times and dates. The elements of a relation may represent events, in which case the valid time-stamp of an element is a single valid time value. Alternatively, the facts represented by the elements of a relation may be true for a duration of time, in which case the valid time-stamp of an element is an interval consisting of two valid time values. The valid time-stamps are used by the database system for implementing the time-slice operator [BZ82, JMS79].

An element may contain a number of *time-invariant attribute values*, i.e., values that never change. An important example is the *time-invariant key* [NA89] which, although it resembles the object surrogate, is still necessary. Social security, account, and membership numbers are important time-invariant keys in many applications. Non-key time-invariant attribute values also exist, e.g., race.

An element may record several facts about a real-world object, using several *time-varying attribute values*. For example, an element may record both the title and the salary of an employee. Each relation may have an individual valid time-stamp granularity, or the database system may impose a fixed granularity on all relations managed by the database system. While different granularities may be ascribed to individual time-varying attributes within an element, it may still be necessary to fix the (overall) element granularity.

An element may also have several *user-defined times*. Such time-stamps are drawn from a domain of dates and times with an identity relation and a total ordering. User-defined times may be manually supplied or computed by an application program. The system gives no special semantics to user-defined times; they are most appropriately thought of as specialized kinds of time-varying attribute values.

Note that in this conceptual model we do not assume any particular type system on surrogates, historical states, or attributes. In particular, while an element is associated with a valid time-stamp, the model makes no mention of whether tuple time-stamping or attribute-value time-stamping is employed. Neither do we assume a particular data model; elements could be tuples in a relational database [Cod70], records in a network database [Dat90], or events in a time sequence collection [SK86]. Finally, the conceptual model of a sequence of historical states does not imply (nor disallow) a particular physical representation. For example, a temporal relation may be represented as a collection of tuples with an event or interval valid time-stamp and an interval transaction time-stamp [Sno87] or with one or two valid time-stamps and three transaction time-stamps [BZ82], as a backlog relation of insertion, modification, and deletion operations (tuples) with single transaction time-stamps [JMRS90] or with time warp attributes [Tho91], and as tuples containing attributes time-stamped with one or more finite unions of intervals (termed *temporal elements* [Gad88], distinct from the term element used in this paper).

3 Specialized Temporal Relations

In this section, we characterize temporal relations according to the interrelations of their time-stamps. In Sections 3.1 and 3.2, we consider singly stamped elements (event stamped), and in Sections 3.3 and 3.4, we consider doubly stamped elements (interval stamped). In Sections 3.1 and 3.3, we characterize relations considering the time-stamps of individual elements in isolation, and in Sections 3.2 and 3.4, we characterize relations considering the interrelations of time-stamps of distinct elements. We provide examples for most of the specialized temporal relations defined here.

All the definitions of relation types in this section are intensional definitions, i.e., for a relation schema to have a particular type, all its possible (non-empty) extensions must satisfy the definition of the type. The restrictions usually apply only to the historical state in which the element was inserted or the historical state in which the element was logically deleted (i.e., the one following the historical state in which the element last appears). Throughout we assume that the valid and transaction time-stamps are drawn from the same domain, which must be totally ordered. We do not consider transaction time domains such as version numbers that cannot be compared with valid time.

Just as the specializations may be applied to an entire relation, i.e., on a *per relation* basis, they may be applied in turn to each partition of a relation, i.e., on a *per partition* basis. This is true because the partitions are sets of elements. Specifically, a relation satisfies a specialization on a per partition basis if every partition of the particular partitioning in turn satisfies the specialization on a per relation basis. While many partitionings are possible, the most useful partitioning is the per surrogate partitioning mentioned in the previous section. It is solely for simplicity that we state explicitly specializations on mainly a per relation basis. In fact, the application of the specializations on a per partition basis may in many situations prove to be more relevant.

By its very nature, a taxonomy should be comprehensive. While striving towards achieving this, we have at the same time attempted to include only specializations that are of practical interest. We show that with some restrictions, the taxonomy based on isolated events is complete. The inter-event based taxonomy is restricted to cover the concepts of sequentiality and regularity, and the isolated interval based taxonomy covers only regularity. The inter-interval based taxonomy distinguishes between temporal relations where elements successive in transaction time have valid time intervals related in one of the 13 possible ways of ordering two intervals. In this sense, the taxonomy is comprehensive within its scope.

The number of specialized temporal relations in the taxonomy may be too large for some uses. To address this potential problem, we have organized the specializations in generalization/specialization hierarchies. Applications that require a small number of specializations may simply consider only the more general specializations.

3.1 Taxonomy on Isolated Events

In this section we consider only *events* that take place at an instant of time in reality. Let R be a temporal relation, and let e be an element of R. Each element e has a single valid time, vt_e, indicating when the event took place in reality. We consider only a single transaction time, tt_e, which is either the insertion or the deletion time, that is, either $tt_e^{\vdash}$ or $tt_e^{\dashv}$. Each property (e.g., *retroactive*, where an element is valid before it is operated on in the database) is relative to one of these two times. For example, it is possible for a relation to be *deletion retroactive* but not *insertion retroactive*. As discussed earlier, a modification consists of a deletion followed by an insertion. If a relation is, say, deletion retroactive *and* insertion retroactive, it can also be considered modification retroactive. The definitions that follow will mention only a single valid time vt_e and a single transaction time tt_e. In examples where we illustrate the definitions, we will assume that tt_e is $tt_e^{\vdash}$ (i.e., we consider insertion, not deletion or modification).

We formally define a number of specialized temporal relations by restricting the allowed interrelations between valid and transaction time-stamp values of isolated elements.

Eleven of the specialized relations and the general relation are illustrated in Figure 1 where the pairs of time-stamp values of elements are restricted to the shaded regions.

Definition: Temporal relation R is *retroactive* if

$$\forall e \in R\ (vt_e \leq tt_e) \qquad \square$$

Thus, the values of an element are valid before they are entered into the relation, i.e., the event occurred before it was stored. Retroactive relations are common in monitoring situations, such as process control in a chemical production plant, where variables such as temperature and pressure are periodically sampled and stored in a database for subsequent analysis. Further, it is often the case that some (non-negative) minimum delay between the actual time of measurement and the time of storage can be determined. For example, a particular set-up for the sampling of temperatures may result in delays that always exceed 30 seconds. This gives rise to a delayed retroactive relation.

Definition: Temporal relation R is *delayed retroactive with bound* $\Delta t \geq 0$ if

$$\forall e \in R\ (vt_e \leq tt_e - \Delta t) \qquad \square$$

In this and in the other specializations that refer to a time bound Δt, this time bound is a *duration* that may be fixed in length (e.g., 30 seconds, one day) or may be calendric-specific. An example of the latter is one month, where a month in the Gregorian calender contains 28 to 31 days, depending on the date to which the duration is added or subtracted.

Definition: Temporal relation R is *predictive* if

$$\forall e \in R\ (vt_e \geq tt_e) \qquad \square$$

Thus, the values of an element are not valid until some time after they have been entered into the relation. An example is a relation that records direct-deposit payroll checks. Generally a copy of this relation is made on magnetic tape near the end of the month, and sent to the bank so that the payments can be effective on the first day of the next month.

Analogously with the delayed retroactive temporal relation which specializes the retroactive temporal relation, the early predictive temporal relation is the specialization of the predictive temporal relation.

Definition: Temporal relation R is *early predictive with bound* $\Delta t \geq 0$ if

$$\forall e \in R\ (vt_e \geq tt_e + \Delta t) \qquad \square$$

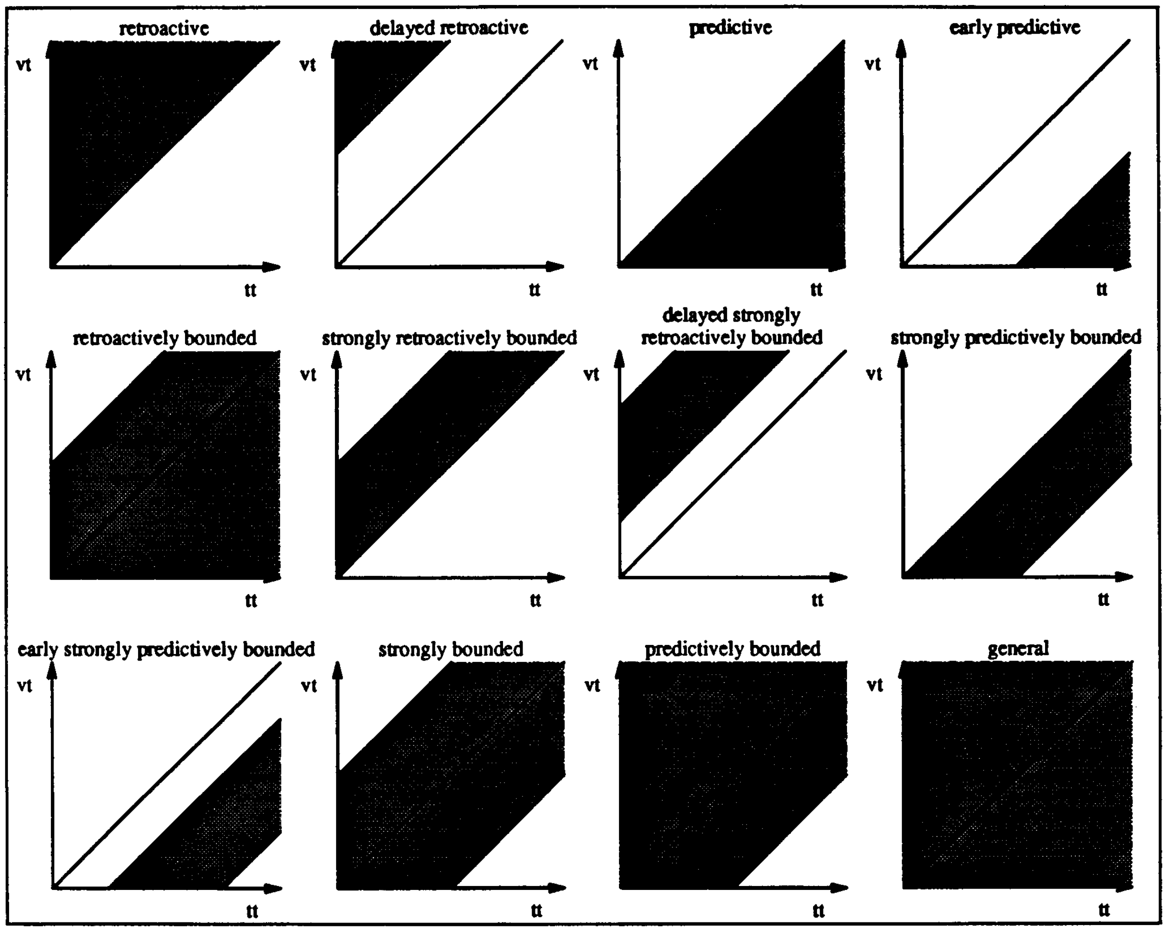
retroactive
delayed retroactive
predictive
early predictive
retroactively bounded
strongly retroactively bounded
delayed strongly retroactively bounded
strongly predictively bounded
early strongly predictively bounded
strongly bounded
predictively bounded
general
vt
tt

Figure 1: Restrictions on Time-stamps in Isolated Event Based Specialized Temporal Relations

The direct-deposit payroll check relation is an example if the tape must be received by the bank at least, say, three days before the day the deposits are to be made effective. Also, this type of relation may be encountered within early warning systems where warnings must be received sometime in advance.

In elements of retroactively bounded temporal relations, the valid time-stamp never is less than the transaction time-stamp by more than a bounded time interval. In all bounded, delayed, and early relations, the bounds are fixed at schema definition time.

Definition: Temporal relation R is *retroactively bounded with bound* $\Delta t \geq 0$ if

$$\forall e \in R\ (vt_e \geq tt_e - \Delta t) \qquad \square$$

Note that in a retroactively bounded relation, the valid time-stamp may exceed the transaction time-stamp. An example is a relation recording the project each employee is assigned to. While assignments may be recorded arbitrarily into the future, an assignment is required to be recorded in the database no later than one month after it is effective.

A strongly retroactively bounded relation is a retroactively bounded temporal relation where the valid time-stamp is less than or equal to the transaction time-stamp.

Definition: Temporal relation R is *strongly retroactively bounded with bound* $\Delta t \geq 0$ if

$$\forall e \in R\ (tt_e - \Delta t \leq vt_e \leq tt_e) \qquad \square$$

The sample relation just discussed is strongly retroactively bounded if future assignments are not stored in the relation.

In a delayed strongly retroactively bounded relation, the valid time-stamp is not only less than the transaction time-stamp within a lower bound—in addition, an upper bound (minimum delay) is also imposed.

Definition: Temporal relation R is *delayed strongly retroactively bounded with bounds* $\Delta t_1 \geq 0$ *and* $\Delta t_2 \geq 0$, where $\Delta t_1 \leq \Delta t_2$, if

$$\forall e \in R\ (tt_e - \Delta t_1 \leq vt_e \leq tt_e - \Delta t_2) \qquad \square$$

The relation that records the assignments of employees is an example of this type of relation if only past assignments are recorded, e.g., if assignments are recorded at most one month after they were effective and if it takes at least two days from the time an assignment is finished until this is known by the data entry clerk.

The strongly predictively bounded and the early strongly predictively bounded relations are symmetrical to the two previous specialized temporal relations. Here the valid time-stamp is in a bounded time interval after the transaction time-stamp, and the early specialization also adds a (positive) lower bound on the valid time-stamp.

Definition: Temporal relation R is *strongly predictively bounded with bound* $\Delta t \geq 0$ if

$$\forall e \in R\,(tt_e \leq vt_e \leq tt_e + \Delta t) \qquad \square$$

Definition: Temporal relation R is *early strongly predictively bounded with bounds* $\Delta t_1 \geq 0$ *and* $\Delta t_2 \geq 0$, where $\Delta t_1 \leq \Delta t_2$, if

$$\forall e \in R\,(tt_e + \Delta t_1 \leq vt_e \leq tt_e + \Delta t_2) \qquad \square$$

Direct deposit pay checks illustrate both types of specialization. The company wants the checks to be valid on the first of the month, but it wants also to make the tape to be sent to the bank as late as possible, generally at most one week before. In addition, the bank needs the tape at least three days in advance.

In a strongly bounded relation, the valid time-stamp may only deviate from the transaction time-stamp within both upper and lower bounds.

Definition: Temporal relation R is *strongly bounded with bounds* $\Delta t_1 \geq 0$ *and* $\Delta t_2 \geq 0$ if

$$\forall e \in R\,(tt_e - \Delta t_1 \leq vt_e \leq tt_e + \Delta t_2) \qquad \square$$

Here, information concerns only the current situation, except that recently valid information and information valid in the near future can be recorded and updated. An example is an accounting relation recording the current month's transactions. Corrections to entries of previous months are stored as compensating transactions in the current month; transactions concerning future months are made to a separate relation.

In elements of predictively bounded temporal relations, the valid time stamp never exceeds the transaction time-stamp by more than a bounded delay. Thus, this kind of relation is symmetric with retroactively bounded relations.

Definition: Temporal relation R is *predictively bounded with bound* $\Delta t \geq 0$ if

$$\forall e \in R\,(vt_e \leq tt_e + \Delta t) \qquad \square$$

Note that in a predictively bounded relation, the valid time-stamp may be less than the transaction time-stamp. In such relations, only information concerning the past and the near-term future may be stored. An example is an order database in which pending orders, constrained by company policy to be no more than 30 days in the future, are stored along with previously filled orders.

A temporal relation is degenerate if the transaction and valid time-stamps of an element are identical (within the selected granularity).

Definition: Temporal relation R is *degenerate* if

$$\forall e \in R\,(vt_e = tt_e) \qquad \square$$

An example is a monitoring situation in which there is no time delay (within the time-stamp granularity) between sampling a value and storing it in the database.

At the implementation level, a degenerate temporal relation can be advantageously treated as a rollback relation due to the fact that relations are append-only and elements are entered in time-stamp order [SR85]. The process of recording degenerate relations is referred to as the *asynchronous method* [Tho91].

A *mapping function* m for a relation R takes as argument an element e of a relation and returns a valid time-stamp, computed using any of the attributes of e, excluding vt_e, but including the surrogate and transaction time-stamp attributes. A temporal relation R is *determined* if it has a mapping function that correctly computes the valid time-stamps of its elements. Sample mapping functions include $m_1(e) = tt_e^{\vdash} + \Delta t$ ("valid after a fixed delay"), $m_2(e) = \lfloor tt_e^{\vdash} - \Delta t \rfloor_{hrs}$ ("valid from the most recent hour"), and $m_3(e) = \lceil tt_e^{\vdash} \rceil_{day} + 8$ hrs ("valid from the next closest 8:00 a.m.").

Definition: Temporal relation R is *determined with mapping function* m if

$$\forall e \in R\,(vt_e = m(e)) \qquad \square$$

Similarly, a relation is *undetermined* if such a function does not exist. For each of the undetermined specialized temporal relations defined already in this section there exists a determined version. To illustrate, consider the determined versions of the retroactive and predictive temporal relations.

Definition: Temporal relation R is *retroactively determined with mapping function* m if

$$\forall e \in R\,(vt_e = m(e) \wedge m(e) \leq tt_e) \qquad \square$$

Thus, a determined relation has a given type if its mapping function obeys the requirement of the type. For example, a relation is retroactively determined if each element is valid from the beginning of the most recent hour during which it was stored.

Definition: Temporal relation R is *predictively determined with mapping function* m if

$$\forall e \in R\,(vt_e = m(e) \wedge m(e) \geq tt_e) \qquad \square$$

For example, a relation is predictively determined if it is valid from the next closest 8:00 a.m. Such a relation might be relevant in banking applications for deposits that are not effective until the start of the next business day.

For further illustration, we present the bounded version of the above two types of relations.

Definition: Temporal relation R is *strongly retroactively bounded determined with mapping function* m *and bound* $\Delta t \geq 0$ if

$$\forall e \in R\,(vt_e = m(e) \wedge tt_e - \Delta t \leq m(e) \leq tt_e) \qquad \square$$

Definition: Temporal relation R is *strongly predictively bounded determined with mapping function* m *and bound* $\Delta t \geq 0$ if

$$\forall e \in R\,(vt_e = m(e) \wedge tt_e \leq m(e) \leq tt_e + \Delta t) \qquad \square$$

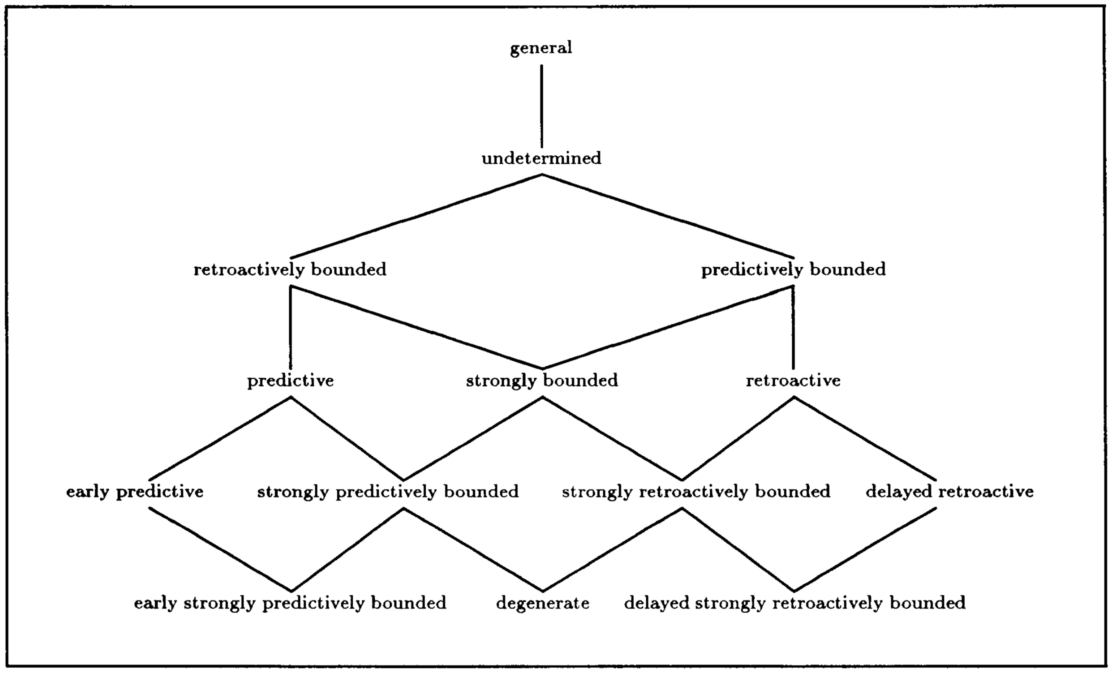

Figure 2: Generalization/Specialization Structure of the Event-based Taxonomy

The examples given previously were in fact bounded.

The generalization/specialization structure of the specialized temporal relations defined above is presented in Figure 2. A relation type can be specialized into any of the successor relation types, and a relation type inherits all the properties of its predecessor relation types (as well as adding additional properties). For clarity, we have included only undetermined relation types; there exist determined counterparts for all the undetermined specialized temporal relations, e.g., strongly bounded determined.

The isolated event based taxonomy is complete with certain assumptions. To state these, note that the specializations in this section correspond to regions of the two-dimensional space spanned by transaction and valid time. There are five assumptions. First, we are interested only in undetermined relationships. Second, we are only interested in regions bounded by lines parallel to the line $tt_e = vt_e$. This means that we do not wish to consider relationships that are dependent on absolute values of the time stamps such as, e.g., the specialization that $vt_e \geq 2 \cdot tt_e$. Third, we consider only relative restrictions on the relationship between valid and transaction times. In combination with the previous assumptions, this implies that only three kinds of lines are of interest when describing restricted regions of the two-dimensional space, namely lines parallel to $tt_e = vt_e$ for which either (1) $vt_e > tt_e$, (2) $vt_e = tt_e$, or (3) $vt_e < tt_e$. Absolute bounds may be added later, by the user of the taxonomy. Fourth, we consider only $\leq$-versions. Pure $<$-versions and mixed versions may be obtained easily. Fifth, only connected regions are considered. Such regions may be used as building blocks to form non-connected regions. As a consequence of the assumptions, at most two lines are required for describing any possible region.

With zero lines we can form no restrictions. Thus, we have a general temporal event relation. With one line, there are two distinct regions for each of the three line-types, resulting in six distinct specialized temporal event relations: early predictive and predictively bounded, predictive and retroactive, and retroactively bounded and delayed retroactive, respectively. With two lines, the are five possibilities corresponding to the combinations (using the numbering of the previous paragraph): (1) and (1) (early strongly predictively bounded), (1) and (2) (strongly predictively bounded), (1) and (3) (strongly bounded), (2) and (3) (stronglv retroactively bounded), and (3) and (3) (delayed strong retroactively bounded). The result is a total of eleven types of specialized temporal relations, each of which is included in the taxonomy.

3.2 Inter-event Based Taxonomy

The previous definitions were based on predicates on individual, event time-stamped elements. A relation schema had a given property if each individual element of any extension meaningful in the modeled reality of the schema satisfied the relevant predicate. We now define restrictions on relation schemas based on the interrelationships of multiple event time-stamped elements in all possible extensions. We examine two aspects: orderings between elements and regularity. In this and later sections, we continue to assume in the examples and explanations that tt_e is $tt_e^{\vdash}$. Recall that while the definitions are made on a per relation ("global") basis, they may also be made on a per partition basis with an arbitrary partitioning, e.g., the per surrogate partitioning.

Definition: Temporal relation R is *globally sequential* if

$$\forall e \in R\, \forall e' \in R\, (tt_e < tt_{e'} \Rightarrow (\max(tt_e, vt_e) \leq \min(tt_{e'}, vt_{e'}))) \quad \square$$

In globally sequential relations, each event must occur *and* be stored before the next event occurs or is (predictively) stored. Therefore, valid time can be approximated with transaction time, yielding an append-only relation that can support historical (as well as transaction time) queries. Such relations may be viewed as approximations to degenerate relations. As an example of the application of this property on a per partition level, R is *per surrogate sequential* if $\forall x \in \pi_{Id}(R)$, $\sigma_{Id=x}(R)$ is globally sequential, where Id is the surrogate attribute.

Now we introduce the notion of a non-decreasing temporal relation. A relation is non-decreasing if elements are entered in valid time-stamp order.

Definition: Temporal relation R is *globally non-decreasing* if

$$\forall e \in R\, \forall e' \in R\, (tt_e < tt_{e'} \Rightarrow vt_e \leq vt_{e'}) \quad \square$$

Sequentiality is generally a stronger property than non-decreasing. However, if the relation is degenerate then the two properties are identical. For completeness, we define also a non-increasing temporal relation where elements are entered in non-increasing valid time-stamp order.

Definition: Temporal relation R is *globally non-increasing* if

$$\forall e \in R\, \forall e' \in R\, (tt_e < tt_{e'} \Rightarrow vt_e \geq vt_{e'}) \quad \square$$

In such relations, as transaction time proceeds, we enter information that is valid further and further into the past. An example is an archeological relation that records information about progressively earlier periods uncovered as excavation proceeds.

Regularity—where transaction time, valid time, or both times occur in regular intervals—is often encountered in temporal relations.

Definition: Temporal relation R is *transaction time event regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R\, \forall e' \in R\, \exists k_e^{e'} (tt_e = tt_{e'} + k_e^{e'} \Delta t) \quad \square$$

Note that the transaction time-stamps of successively stored elements need not be evenly spaced; they are merely restricted to be separated by an integral multiple ($k_e^{e'}$) of a specified duration, Δt. An example is a periodic sampling of some physical variable such as temperature. The process of recording transaction time event regular relations is referred to as the *synchronous method* [Tho91].

Definition: Temporal relation R is *valid time event regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R\, \forall e' \in R\, \exists k_e^{e'}\, (vt_e = vt_{e'} + k_e^{e'} \Delta t) \quad \square$$

The concept of *granularity* of valid time-stamps can be expressed in terms of this property. For example, if the valid time-stamp granularity is one second then, equivalently, the relation is valid time event regular with time unit one second.

Definition: Temporal relation R is *temporal event regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R\, \forall e' \in R\, \exists k_e^{e'}\, (vt_e = vt_{e'} + k_e^{e'} \Delta t \wedge tt_e = tt_{e'} + k_e^{e'} \Delta t) \quad \square$$

A periodic degenerate relation is trivially temporal event regular. Note that the same values of $k_e^{e'}$ must satisfy both transaction and valid time. Therefore, temporal event regular is more restrictive than both valid and transaction time event regular together.

Next, we define strict versions of the three different variants of regular specialized temporal relations.

Definition: Temporal relation R is *strict transaction time event regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R\, (\exists e' \in R(tt_{e'} = tt_e + \Delta t \wedge \neg \exists e'' \in R\, (tt_e < tt_{e''} < tt_{e'})) \vee \neg \exists e' \in R\, (tt_{e'} > tt_e)) \quad \square$$

Thus, either e' is the next element after e, or e is the last element stored.

Definition: Temporal relation R is *strict valid time event regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R\, (\exists e' \in R\, (vt_{e'} = vt_e + \Delta t \wedge \neg \exists e'' \in R - \{e, e'\}\, (vt_e \leq vt_{e''} \leq vt_{e'})) \vee \neg \exists e' \in R\, (vt_{e'} > vt_e)) \quad \square$$

This definition is slightly more complicated than the previous one because we want to disallow elements with identical valid times (which is already impossible with transaction time).

Definition: Temporal relation R is *strict temporal event regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R\, ((\exists e' \in R\, (tt_{e'} = tt_e + \Delta t \wedge vt_{e'} = vt_e + \Delta t \wedge \neg \exists e'' \in R\, (tt_e < tt_{e''} < tt_{e'}) \wedge \neg \exists e'' \in R - \{e, e'\}\, (vt_e \leq vt_{e''} \leq vt_{e'}))) \vee (\neg \exists e' \in R\, (tt_{e'} > tt_e) \wedge \neg \exists e' \in R\, (vt_{e'} > vt_e))) \quad \square$$

While somewhat complex, this definition is just the combination of the two previous definitions, using the same duration for both valid and transaction time.

Note that if relation R' is transaction time event regular with time unit Δt_1 and valid time event regular with time unit Δt_2, then R' is also temporal event regular, the temporal time unit Δt_3 being some common divisor of Δt_1 and Δt_2. Thus, if $\Delta t_1 = 28$ seconds and $\Delta t_2 = 6$ seconds then $\Delta t_3 = 2$ seconds (largest common divisor). For the strict case, however, valid and transaction time event regularity does not imply temporal event regularity.

Analogous with the ordering properties, the above regularity properties can be defined in a global or per partition fashion. However, the non-strict versions have the additional property (not shared with ordering and strictness) that the per partition variant implies the global variant. Note that regularity is a different property than *periodicity*, which encodes facts such as something is true from 2 to 4p.m. during weekdays [LJ88].

All of these characterizations are orthogonal to those given in the previous section for individual events, except that a degenerate event relation is necessarily globally ordered.

The generalization/specialization structures for the temporal relations defined in this section are outlined in Figures 3 and 4. The two structures are orthogonal.

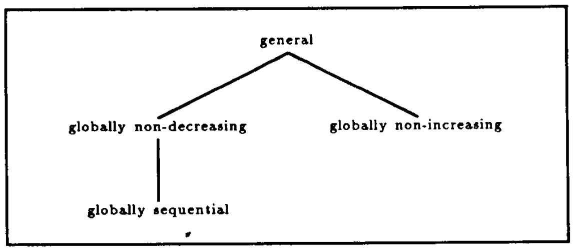

Figure 3: Generalization/Specialization Structure of the Inter-event Based Taxonomy (Part I — orderings)

3.3 Taxonomy on Isolated Intervals

We now turn to interval relations, that is, those relations in which, for each element e of the relation, the valid time is an interval, $[vt_e^{\vdash}, vt_e^{\dashv})$. The transaction times of the element, $tt_e^{\vdash}$ and $tt_e^{\dashv}$, are defined as before. As in Section 3.2, k (possibly indexed) is an integer.

The previous characterizations of events may also be applied to either $vt_e^{\vdash}$ or $vt_e^{\dashv}$. For example, if an interval is stored as soon as it terminates, a designer may state that the interval relation is $vt^{\vdash}$-retroactive and $vt^{\dashv}$-degenerate. If the relation is, say, $vt^{\vdash}$-retroactive and $vt^{\dashv}$-retroactive, it may simply be termed retroactive.

A temporal relation is transaction time regular, valid time regular, or temporally regular if the transaction time intervals, valid time intervals, or both transaction time and valid time intervals are regular, respectively. Note again that these properties concern durations rather than starting events, and that they can be calendric specific, e.g., one month.

Definition: Temporal relation R is *transaction time interval regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R \; \exists k_e \; (tt_e^{\dashv} = tt_e^{\vdash} + k_e \Delta t) \qquad \Box$$

Definition: Temporal relation R is *valid time interval regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R \; \exists k_e \; (vt_e^{\dashv} = vt_e^{\vdash} + k_e \Delta t) \qquad \Box$$

Alternatively, the duration of all intervals in such a relation is an integral multiple of a specified time unit. An example is a relation recording new hires and terminations that observes a company policy that all such hires and terminations be effective on either the first or the fifteenth of each month.

Definition: Temporal relation R is *temporal interval regular with time unit* $\Delta t \geq 0$ if

$$\forall e \in R \; \exists k_e^1 \; \exists k_e^2 \; (tt_e^{\dashv} = tt_e^{\vdash} + k_e^1 \Delta t \wedge vt_e^{\dashv} = vt_e^{\vdash} + k_e^2 \Delta t) \qquad \Box$$

Hence, the time unit must be identical for both transaction and valid time.

The situations where all intervals have the same length are interesting special cases of the above definitions with k_e, k_e^1, and k_e^2 equal to 1. These special cases, we term *strict transaction time interval regular*, *strict valid time interval regular*, and *strict temporal interval regular*.

Recall that the concept of regularity may be applied to relations on a per partition basis as well as globally (as discussed at the beginning of this section).

The specializations in the previous section concern event relations, and the specializations in this section concern interval relations; they are quite different. However, the generalization/specialization structure of the specializations in this section is identical to that of the previous section as illustrated in Figure 4, with the exception that "event" is replaced by "interval."

3.4 Inter-interval Based Taxonomy

As with events, we distinguish restrictions that are applied individually to all intervals and restrictions on the interrelationship between multiple intervals in a relation. The restrictions listed below apply to relations, but they may be applied on a per partition basis as well. Many of these same terms also apply to event relations, and were defined in Section 3.2; context should differentiate these uses.

Definition: Temporal relation R is *globally sequential* if

$$\forall e \in R \; \forall e' \in R \; (tt_e < tt_{e'} \Rightarrow (\max(tt_e, vt_e^{\dashv}) \leq \min(tt_{e'}, vt_{e'}^{\vdash}))) \qquad \Box$$

In such a relation, each interval must occur and be stored before the next interval commences. An example involves the relation previously discussed that records the weekly assignments for employees. If the assignment for the next week is recorded during the weekend then this relation will be per surrogate sequential.

A relation is non-decreasing if elements are entered in valid time-stamp order, and it is non-increasing if elements are entered in reverse valid time-stamp order.

Definition: Temporal relation R is *globally non-decreasing* if

$$\forall e \in R \; \forall e' \in R \; (tt_e < tt_{e'} \Rightarrow vt_e^{\dashv} \leq vt_{e'}^{\vdash}) \qquad \Box$$

Concerning the example just discussed, let us now record each Thursday the next week's assignment. In this case the transaction time (i.e., Thursday) of the next week's assignment (on a per surrogate basis) will occur during the valid time interval of the current week's assignment, and the relation will be per surrogate non-decreasing.

As with events, sequentiality is a stronger property than non-decreasing.

Definition: Temporal relation R is *globally non-increasing* if

$$\forall e \in R \; \forall e' \in R \; (tt_e < tt_{e'} \Rightarrow vt_{e'}^{\dashv} \leq vt_e^{\vdash}) \qquad \Box$$

Definition: Temporal relation R is *globally contiguous* if

$$\forall e \in R \; (\exists e' \in R - \{e\} \; (vt_e^{\dashv} = vt_{e'}^{\vdash} \wedge tt_e < tt_{e'} \wedge \neg \exists e'' \in R - \{e, e'\} (tt_e < tt_{e''} < tt_{e'})) \vee \forall e' \in R - \{e\} \; (vt_e^{\vdash} \geq vt_{e'}^{\dashv})) \qquad \Box$$

This definition states that in a globally contiguous relation, the end of one event coincides with the start of the next event that is stored, unless the event is the last one in the sequence, in which case it occurs after all the other events.

Allen has demonstrated that there exist a total of thirteen possible relationships between two intervals [All83]. These relationships may be denoted *before*, *meets*, *overlaps*, *during*, *starts*, *finishes*, *equal*, and the inverse relationships for all but *equal*, e.g., *inverse before* and *inverse finishes*. For each such relationship, **X**, we can define a property *successive transaction time* **X** that requires that elements, successive in transaction time, are related by **X**. For example, the property *successive transaction time overlaps* requires that intervals that are adjacent in transaction time overlap in valid time, ensuring that the next element began before the previous one completed.

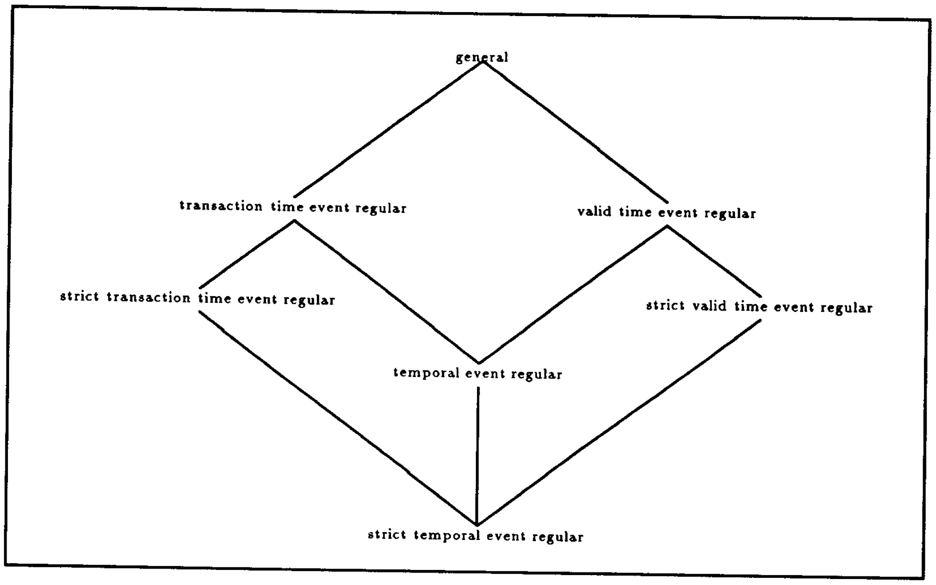
general
transaction time event regular
valid time event regular
strict transaction time event regular
strict valid time event regular
temporal event regular
strict temporal event regular

Figure 4: Generalization/Specialization Structure of the Inter-event Based Taxonomy (Part II — regularity)

Definition: Temporal relation R is *successive transaction time* **X** if

$$\forall e \in R\, (\exists e' \in R - \{e\}\ (vt_e \mathbf{X} vt_{e'} \wedge tt_e < tt_{e'} \wedge \neg\exists e'' \in R - \{e, e'\}(tt_e < tt_{e''} < tt_{e'})) \vee \forall e' \in R - \{e\}\ (tt_e \geq tt_{e'})) \qquad \square$$

Of these, the most interesting is *successive transaction time meets*, which is defined above as *globally contiguous*.

Figure 5 illustrates the specialization/generalization structure for the properties discussed above. In this figure, *successive transaction time* is abbreviated 'st-', and *successive transaction time inverse* is abbreviated 'sti-'.

4 Conclusion and Future Research

A temporal relation has two database system-interpreted time attributes, transaction time and valid time. A transaction time-stamp is a simple value, indicating when a fact is stored in the temporal relation. A valid time-stamp records the validity of a fact, and it may be a simple value (event relation) or an ordered pair of simple values (interval relation). In general, these time-stamps are independent, meaning that facts may be associated with a point or a pair of points in an unrestricted two-dimensional space. In many situations, however, the time points of facts are restricted to limited regions of this space, resulting in specialized temporal relations. Examples include process monitoring, satellite surveillance of crops or weather, accounting applications, and real-time databases. The restricted interrelations of time-stamps constitute important semantics of temporal relation schemas.

In this paper, we considered the specialized semantics of the time attributes in temporal relations. We presented an extensive taxonomy of temporal specializations, some restricting the stamps of individual facts, others restricting the stamps on an inter-fact basis. The practical relevance of the definitions was emphasized by examples. The properties apply to either event or interval temporal relations. A relation may have specialized per element properties (Sections 3.1 and 3.3) as well as specialized inter-element properties (Sections 3.2 and 3.4). The taxonomy provides a better understanding of the nature of individual temporal relations and of how various temporal data models compare.

Future work is indicated in two areas. While specialized temporal relations present an opportunity to optimize temporal queries, more work is needed to exploit such specializations. Our contention is that most previous work in this area is relevant; still, the details need to be worked out.

An overall approach to designing temporal databases is still needed. This paper has considered only half of the problem of designing temporal relations: determining the characteristics of the time-stamp attributes that concern entire elements. Just as important are the characteristics of the individual time-varying attributes. A fully articulated design methodology for temporal relations must address both time-stamp attributes and time-varying attributes.

Acknowledgements

The topic of temporal specialization was introduced during discussions between the second author and Arie Segev, whose comments encouraged further exploration of what was then termed "coupled relations." Mike Stonebraker's use of valid time examples to illustrate rollback relations in Postgres (which only record transaction time) implied that Postgres supports temporal relations; the research reported here into this implication shows that it is true. Comments on earlier drafts of this paper by Lars Bækgaard, Curtis Dyreson, Suchen Hsu, Leo Mark and Michael Soo helped improve the presentation.

The work was conducted while the first author visited the University of Arizona, and was supported in part by NSF grant ISI-8902707. In addition, the first author was supported by Danish NSF (Statens Naturvidenskabelige Forskningsråd) grant 11-8696-1 SE.

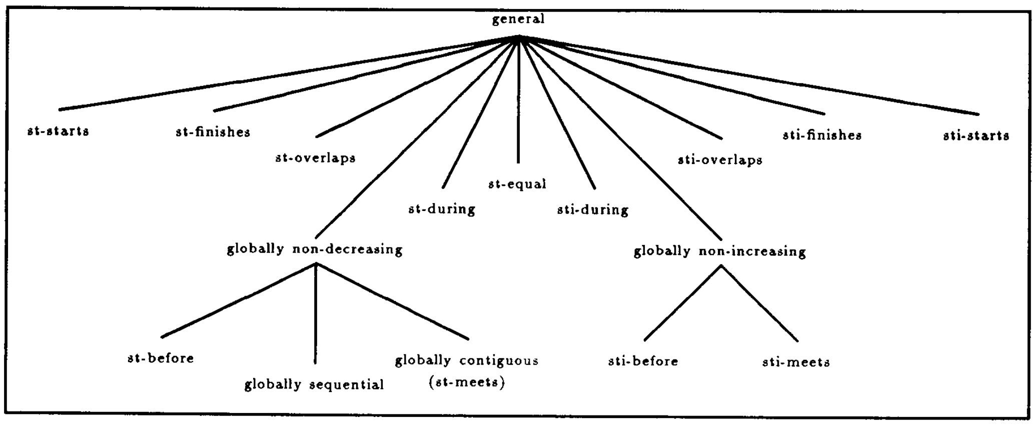

Figure 5: Generalization/Specialization Structure of the Inter-interval Based Taxonomy

References

[All83] J.F. Allen. Maintaining Knowledge about Temporal Intervals. *CACM*, 26(11):832–843, November 1983.

[BZ82] J. Ben-Zvi. *The Time Relational Model.* PhD dissertation, Computer Science Department, UCLA, 1982.

[Cod70] E. F. Codd. A Relational Model of Data for Large Shared Data Banks. *CACM*, 13(6):377–387, June 1970.

[Dat85] C. J. Date. *An Introduction to Database Systems—Volume II.* The Systems Programming Series. Addison Wesley Publishing Company, First edition, July 1985.

[Dat90] C.J. Date. *An Introduction to Database Systems — Volume I.* Systems Programming Series. Addison-Wesley, Reading, MA, Fifth edition, 1990.

[Gad88] S.K. Gadia. A Homogeneous Relational Model and Query Languages for Temporal Databases. *ACM TODS*, 13(4):418–448, December 1988.

[HOT76] P. Hall, J. Owlett, and S.J.P. Todd. Relations and Entities. In G. M. Nijssen, editor, *Modelling in Data Base Management Systems*, pages 201–220. North-Holland, 1976.

[JMR91] C.S. Jensen, L. Mark, and N. Roussopoulos. Incremental Implementation Model for Relational Databases with Transaction Time. *IEEE TKDE*, 3(4):to appear, December 1991.

[JMRS90] C.S. Jensen, L. Mark, N. Roussopoulos, and T. Sellis. Using Caching, Cache Indexing, and Differential Techniques to Efficiently Support Transaction Time. University of Maryland, CS-TR-2413, UMIACS-TR-90-25, February 1990.

[JMS79] S. Jones, P. Mason, and R. Stamper. LEGOL 2.0: A Relational Specification Language for Complex Rules. *Info. Sys.*, 4(4):293–305, November 1979.

[LJ88] N. Lorentzos and R. Johnson. Extending Relational Algebra to Manipulate Temporal Data. *Info. Sys.*, 13(3): 288–296, 1988.

[NA89] S. B. Navathe and R. Ahmed. A Temporal Relational Model and a Query Language. *Info. Sci.*, 49(1):147–175, October 1989.

[SA85] R.T. Snodgrass and I. Ahn. A Taxonomy of Time in Databases. In *Proceedings of the ACM SIGMOD '85*, pages 236–246, 1985.

[SA86] R.T. Snodgrass and I. Ahn. Temporal Databases. *IEEE Computer*, 19(9):35–42, September 1986.

[Sch77] B. Schueler. Update Reconsidered. In *Proceedings of the IFIP Working Conference on Modelling in Data Base Management Systems*, pages 149–164, 1977.

[SK86] A. Shoshani and K. Kawagoe. Temporal Data Management. In *Proceedings of the Twelfth VLDB Conference*, pages 79–88, August 1986.

[Sno87] R.T. Snodgrass. The Temporal Query Language TQuel. *ACM TODS*, 12(2):247–298, June 1987.

[SR85] M.R. Stonebraker and L.A. Rowe. The Design of Postgres. University of California at Berkeley, UCB/ERL 85/95, November 1985.

[SS87] A. Segev and A. Shoshani. Logical Modeling of Temporal Data. In *Proceedings of the ACM SIGMOD '87*, pages 454–466, May 1987.

[Tho91] P.M. Thompson. *A Temporal Data Model Based on Accounting Principles.* PhD dissertation, University of Calgary, March 1991.