Valid-Time Indeterminacy in Temporal Relational Databases: A Family of Data Models

Luca Anselma
Dipartimento di Informatica
Università di Torino
Torino, Italy
e-mail: anselma@di.unito.it

Paolo Terenziani
Dipartimento di Informatica
Università del Piemonte Orientale
Alessandria, Italy
e-mail: terenz@mfn.unipmn.it

Richard T. Snodgrass
Department of Computer Science
University of Arizona
Tucson, AZ, USA
e-mail: rts@cs.arizona.edu

Abstract—**Valid-time indeterminacy concerns not knowing exactly when a fact holds in the modeled reality. In this paper, we first propose a reference approach (data model and algebra) in which all possible temporal scenarios induced by valid-time indeterminacy can be extensionally modeled. We then specify a family of sixteen more compact representational data models. We demonstrate their correctness with respect to the reference approach and analyze several properties, including their data expressiveness and correctness with respect to the reference approach. Finally, we compare these compact models along several relevant dimensions.**

Keywords-temporal relational database; valid-time indeterminacy; data models and algebras

I. Introduction

Time is pervasive and in many situations the dynamics of domains is one of the most relevant aspects to be captured by a data model. Many approaches to temporal databases (TDBs) have been developed over the last two decades. In particular, TSQL2 [9] represents a consensus of part of the community and BCDM [6; 9] has provided a unifying underlying semantics of several relational TDB approaches including TSQL2.

Valid-time indeterminacy ("don't know when" information [4]) comes into play whenever the valid time associated with some piece of information in the database is (partially or totally) unknown. For instance, consider Ex.1 (at the granularity of hours).

Ex.1. On Jan 1st 2010 between 1am (included) and 4am (excluded) John had breathing problems. ◊

The fact "John had breathing problems" holds at an unknown number of time units (hours), ranging from hours 1 to 3 inclusive (e.g., it may hold on 1, 2, and 3, or on 1 and 3, or on 2 only; for the sake of brevity, in this paper we denote by "n" the hour from "n" to "n+1", and we assume to start the numbering of hours on Jan 1st 2010). Notice that, as a border case, also the fact that a given event might have occurred or not (i.e., indeterminacy about the *existence* of the fact) may be interpreted as a form of valid-time indeterminacy; consider:

Ex.2. On Jan 1st 2010 between 1am (included) and 4am (excluded) Mary might have had an ischemic stroke. ◊

Coping with valid-time indeterminacy is important in many database applications, as the time when facts happen is often partially unknown. However, the treatment of valid-time indeterminacy has not received much attention in the TDB literature (for example, only determinate time is treated by BCDM; see, however, the survey in [7]).

In our approach, we cope with valid-time indeterminacy in relational databases by exploiting a methodology which has often been used in Computer Science:

(1) first, we propose a reference approach (data model and algebra) coping with the full complexity of the phenomenon; and only then

(2) we devise approaches adopting more user-friendly, compact and/or efficient representations.[1]

Our reference approach allows one to *extensionally model* (bringing to mind *data expressiveness*) and *query* (*query expressiveness*) all possible temporal scenarios induced by valid-time indeterminacy. Our approach provides a *consistent extension* of BCDM, in the sense that determinate valid time can be easily coped with as a special case (thus granting for the compatibility and interoperability with existent approaches). However, data/query expressiveness is not the only criterion to evaluate approaches. It is also important to provide users with formalisms that model phenomena in a "natural" and "compact" way, and computational efficiency is important in many applications (see again, e.g., footnote 1).

Therefore, we first identify four parameters (for example, one being the ease and compactness in coping with constraints about valid-time minimal duration). Each parameter is independently satisfied (or not). On the basis of these parameters, we propose a family of sixteen different approaches allowing one to cope with any combination of such parameters in a more compact and user-friendly way (with respect to the reference approach). Each approach is characterized

(i) by a different *formalism* to represent valid time,

(ii) by the definition of *set operations* (union, intersection and difference) on the given representation of valid time, and

(iii) by the *relational algebra operations* based on such set operations.[2]

[1] For example, Allen [1] devised a reference approach (the Interval Algebra) for representing and reasoning with qualitative relations about periods. Then, e.g., for the sake of user-friendliness [5] or of computational efficiency [8], researchers devised more restricted representations, often at the price of losing some expressiveness of the reference approach.

[2] E.g., as in BCDM, TSQL2 and many TDB approaches, in our approach temporal Cartesian product involves the intersection of the valid times of the tuples being "combined".

For each data representation formalism, we study its *semantics* and *data expressiveness* with respect to the reference approach, and also analyze whether it is a *consistent extension* of BCDM. We have defined the set operators within the different languages in such a way that they are proven to be *correct* with respect to the reference approach (roughly speaking, this means that – although they operate on a more compact representation – they provide the same results as the reference approach). However, we proved that not all the sixteen representations could support a *closed* definition of set operators: in some approach, the correct result of set operations cannot be expressed in the representation language (of course, only approaches which support a closed definition of set operators –"closed" approaches for short– are suitable for DB applications). For each "closed" approach, we define the relational algebraic operators as a polymorphic adaptation of the operators of the reference approach and determine whether each is a consistent extension of the BCDM operators.

Finally, we summarize the different results with a comparison table. This paper thus provides users with a family of approaches among which they can choose, in order to find the best-suited approach to model their own application domain, as well as a formal framework which can be used in order to analyze and classify past (and, possibly, future) TDB approaches to valid-time indeterminacy.

For the sake of brevity, in this paper we only describe our reference approach (Section II) and one of the compact approaches (Section III), just summarizing in Section IV results concerning the other approaches in the family. Finally, in Section V, we propose comparisons and conclusions. It is worth noting that the treatment of probabilities, of multiple granularities, and the extension of SQL language are outside the scope of this paper.

II. Extensional Approach

In this section, we introduce the reference approach we propose to cope with temporal indeterminacy.

A. Disjunctive temporal elements

As in BCDM and in many approaches in the literature (see, e.g., [9]), as well as in our model, time is discrete, linearly ordered, and isomorphic to the integers. For the sake of simplicity, a single granularity (e.g., hour) is assumed.

Definition: **chronon**. The chronon is the basic time unit. The chronon domain TC, also called timeline, is the set of chronons $\{..., c_i, ..., c_j, ...\}$, with $c_i < c_j$. ■

As in BCDM, *temporal elements* are used in order to associate with each tuple its valid time.

Definition: **Temporal element**. A temporal element is a set of chronons, defined on the domain 2^{TC}. ■

For the sake of efficiency, many TDB approaches represent convex sets of chronons using time periods (consider, e.g., TSQL2 [9]). However, in most cases, time periods are interpreted as a compact representation of a set of chronons (see the discussion in Chapter 10 of [9]). In this paper we do not consider time periods, leaving that as an optimization which can be easily added on top of our approach later on.

Disjunctions of temporal elements are a natural way of coping with valid-time indeterminacy, in which each temporal element models one of the alternative possible temporal scenarios (any one of which could be valid).

Definition: **Disjunctive temporal element** (termed *DTE* henceforth). A disjunctive temporal element is a disjunctive set of temporal elements. Given a temporal domain TC, a DTE is an element in $2^{2^{TC}}$. ■

For example, the following DTE models the valid time of the fact in Ex.1 above: $\{\{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$. ◊

Notice that indeterminacy about existence is simply modeled by including the empty temporal element within a DTE. Determinate times can be simply modeled through a DTE containing just one temporal element (called *singleton DTE* henceforth).

Property: Consistent extension of BCDM. Any (determinate) temporal element can be modeled by a singleton DTE. ■

B. Temporal tuples and relations

To represent facts that are temporally indeterminate, DTEs are used as timestamps. Intuitively, DTEs cope with valid-time indeterminacy by explicitly modeling all the alternative temporal scenarios.

Definition: (valid-time) indeterminate tuple and relation. Given a schema $A_1, ..., A_n$ (where each A_i represents a non-temporal attribute on the domain D_i), a (valid-time) indeterminate relation r is an instance of the schema $A_1, ..., A_n \mid VT$ defined over the domain $D_1 \times ... \times D_n \times 2^{2^{TC}}$ in which value-equivalent tuples are not admitted (as in BCDM). Each tuple $t = (v_1, ..., v_n \mid d) \in r$, where d is a DTE, is termed a (valid-time) indeterminate tuple. The DTE $d = \{\{c_i, ..., c_j\}, ..., \{c_h, ..., c_k\}\}$ within tuple t denotes that the tuple t holds either at each chronon in $\{c_i, ..., c_j\}$ or … or at each chronon in $\{c_h, ..., c_k\}$. ■

For example, consider a temporal indeterminate relation called CLINICAL_RECORD. The first tuple models Ex.1. The second tuple models Ex.2, considering the additional knowledge (constraint) that the ischemic stroke, if any, has been unique, and has occurred in (at most) one hour. Finally, the third tuple models the following.

Ex.3. On Jan 1st 2010 Sue might have had an ischemic stroke either at 1am or at 2am.

CLINICAL_RECORD
{ (John, breath | {{1},{2},{3},{1,2},{1,3},{2,3},{1,2,3}}),
  (Mary, stroke | {{},{1},{2},{3}}),
  (Sue, stroke | {{},{1},{2}}) } ◊

C. Lattice of scenarios

The elements of 2^{TC} with the standard set inclusion form a lattice which represents the space of all possible alternative scenarios over the temporal domain TC. We term this a *lattice of scenarios* (over TC).

Property: Expressiveness. By definition, the formalism in this section allows one to express (i.e., to associate with each tuple) any combination of possible scenarios (i.e., any subset of the lattice of scenarios). ■

In Sections III and IV we describe also less expressive (but more compact) formalisms, which cannot represent all possible combinations of scenarios (i.e., not all subsets of the lattice of scenarios).

D. Algebraic operations

Codd designated as complete any query language that was as expressive as his set of five relational algebraic operators: relational union ($\cup$), relational difference ($-$), selection (σ_P), projection (π_X), and Cartesian product ($\times$) [2]. Here we generalize these operators to cover (valid-time) indeterminate relations. As in several TDB approaches, our temporal operators behave as standard non-temporal operators on the non-temporal attributes, and apply set operators on the temporal component of tuples (consider, e.g., [9]). Specifically, as in many TDB approaches, including TSQL2 and BCDM, in our proposal Cartesian product involves the intersection of the temporal components, projection and union involve their union, and difference the difference of temporal components (this definition can be motivated by a *sequenced* semantics [3]: results should be valid independently at each point of time). For the sake of brevity, only temporal Cartesian product ($\times^{ti}$) is defined here.

Definition: ***temporal Cartesian product.*** Let *r* and *s* denote two (temporal) indeterminate relations. The temporal Cartesian product of *r* and *s* is defined as

$$r \times^{ti} s \;=\; \{\, (\, v_r \cdot v_s \mid t\,) \mid \exists t_r\ \exists t_s\ (\ (\, v_r|t_r\,) \in r \wedge (\, v_s \mid t_s\,) \in s \wedge t = t_r \cap^{ti} t_s \wedge t \neq \varnothing\ )\ \},$$

where $\cap^{ti}$ denotes the intersection between DTEs. ■

Indeed, a further step is needed to cope with valid-time indeterminacy: the set operators of intersection, union and difference must be generalized to apply to DTEs.

Definition: $\cup^{ti}$, $\cap^{ti}$, and $-^{ti}$. Given two DTEs *DA* and *DB*, and denoting temporal elements by *A* and *B,* the operations of union ($\cup^{ti}$), intersection ($\cap^{ti}$), and difference ($-^{ti}$) between *A* and *B* are defined as the DTE obtained through the pairwise application of standard set operations on temporal elements:

$$DA \cup^{ti} DB = \{A \cup B \mid A \in DA \wedge B \in DB\ \}$$
$$DA \cap^{ti} DB = \{A \cap B \mid A \in DA \wedge B \in DB\ \}$$
$$DA -^{ti} DB = \{A - B \mid\ A \in DA \wedge B \in DB\ \}.$$ ■

Intuitively, DTEs represent valid-time indeterminacy by eliciting all possible alternative determinate scenarios. The rationale behind our definition is simply that the pairwise combination of each alternative scenario must be taken into account. For instance, considering the CLINICAL_RECORD relation, $\{\{\}, \{1\}, \{2\}, \{3\}\} \cap^{ti} \{\{\}, \{1\}, \{2\}\}$ identifies all times when both Mary and Sue had a stroke, and the final result is the set of scenarios obtained by combining each scenario for Mary and Sue through pairwise intersection, i.e., $\{\ (\{\}\cap\{\}), (\{\}\cap\{1\}), (\{\}\cap\{2\}), (\{1\}\cap\{\}), (\{1\}\cap\{1\}), (\{1\}\cap\{2\}), (\{2\}\cap\{\}), (\{2\}\cap\{1\}), (\{2\}\cap\{2\}), (\{3\}\cap\{\}), (\{3\}\cap\{1\}), (\{3\}\cap\{2\})\ \} = \{\ \{\}, \{1\}, \{2\}\ \}$.

Property: Consistent extension (set operators). In cases where only singleton DTEs are used, the set operators $\cup^{ti}$, $\cap^{ti}$, and $-^{ti}$ are equivalent (by definition) to the standard set operators $\cup$, $\cap$ and $-$. ■

As a consequence of the above property (and of treating non-temporal attributes in the same way), our temporal algebra is a consistent extension of BCDM's one (considering valid time only).

Property: Consistent extension (relational algebraic operators). If singleton DTEs are used as valid times associated with tuples, the relational operators $\cup^{ti}$, $-^{ti}$, $\sigma_P{}^{ti}$, $\pi_X{}^{ti}$ and $\times^{ti}$ are equivalent to the standard BCDM valid-time relational operators $\cup^{t}$, $-^{t}$, $\sigma_P{}^{t}$, $\pi_X{}^{t}$ and $\times^{t}$. ■

III. COMPACT REPRESENTATIONS: DETERMINATE AND INDETERMINATE CHRONONS

The above treatment of valid-time indeterminacy is expressive but has several limitations. It is not compact and thus possibly not user-friendly, since all possible scenarios need to be elicited. Computational efficiency is a related crucial issue, since temporal and algebraic operators must consider all the combinations of possible scenarios.

More compact (and efficient) representations of temporal indeterminacy can be devised, sometime at the price of loosing part of the data expressiveness of the reference extensional approach. Indeed, the limited expressiveness may be acceptable in several real-world domains. For instance, in this section we present a compact approach useful in domains where

(i) one can identify a (possibly empty) set of chronons in which the fact certainly holds (*determinate chronons* henceforth), and a set of chronons at which it may hold (*indeterminate chronons* henceforth), and

(ii) indeterminate chronons are independent of each other, in the sense that all combinations of indeterminate chronons are possible alternative scenarios.

For instance, consider the following.

Ex.4. On Jan 1st 2010 Tim had breathing problem certainly at 1am and possibly at 2am or 3am. ◊

Here the fact holds in hour 1 and possibly holds in hours 2 and 3 (meaning that it may hold at *{1}, {1,2}, {1,3}, {1,2,3}*). In this section, we show that valid times of this type can be modeled by a representation formalism that is (strictly) less *data expressive* than the formalism of DTEs, yet supports a more compact and user-friendly representation.

In the following, we first cope with temporal elements, and then extend our approach to cope with relations and algebraic operators.

A. Determinate+Indeterminate Temporal Elements

Definition: Determinate+Indeterminate temporal elements (DITEs henceforth). A DITE is a pair *<d,i>*, where *d* and *i* are temporal elements. ■

Intuitively, the first element of the pair identifies the determinate chronons and the second element the indeterminate ones. The extensional semantics of such a representation can be formalized taking advantage of the general approach in Section II.

Extensional semantics of DITEs. The semantics of a DITE $<d,i>$ is the DTE consisting of all and only the sets $\{ e1 \cup e2 \mid e1 \in d, e2 \in 2^i \}$. We term *Ext* the function that associates its extensional semantics with each DITE. ■

For example, Ex.4 can be represented by the determinate+indeterminate temporal element $<\{1\},\{2,3\}>$, and its underlying semantics is the DTE $Ext(<\{1\},\{2,3\}>) = \{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}\}$. ◊

DITEs are less expressive than DTEs, since not all combinations of temporal scenarios can be expressed.

Property: Expressiveness of DITE. Given a temporal domain *TC*, DITEs allow one to express all and only the subsets of $2^{2^{TC}}$ of the form $DET \cup 2^{INDET}$, where $DET \subseteq T^C$ and $INDET \subseteq T^C$. ■

Intuitively, the formalism only allows one to cope with those subsets of TC in which (besides determinate chronons) all the combinations of indeterminate chronons are present. For instance, Ex.3 is not expressible, since the indeterminate chronons 1 and 2 are not independent of each other since they are mutually exclusive. Determinate valid time can be easily captured by means of DITEs, as stated by the following property.

Property: Consistent extension (DITE). Temporal elements can be modeled by DITEs of the form $<DET,\{\}>$ (i.e., elements having an empty set of indeterminate chronons). ■

B. *Determinate+Indeterminate Tuples and Relations*

We deliberately maintain the same definition for both the association of time with tuples and for the definition of the algebraic operators. Informally, we could say that the definitions in Section II can be abstracted and interpreted in a polymorphic way, by substituting the proper definition of temporal element and of set operators on temporal elements.

Definition: DI tuple and DI relation. Given a schema $A_1, ..., A_n$ where each A_i represents a non-temporal attribute on the domain D_i, a temporal DI relation r is an instance of the schema $A_1, ..., A_n \mid DET,INDET$ defined over the domain $D_1 \times ... \times D_n \times 2^{TC} \times 2^{TC}$ in which value-equivalent tuples are not admitted. Each tuple $t = (v_1, ..., v_n \mid <d,i>) \in r$, where d and i are temporal elements, is termed a *DI tuple*. The DITE $<d,i> =<\{c_i, ..., c_j\},\{c_h, ..., c_k\}>$ within tuple t denotes that the tuple t holds in each chronon in $Ext(<d,i>)$, i.e., it holds in $\{c_i, ..., c_j\}$ and may hold in any chronon in $\{c_h, ..., c_k\}$. ■

Determinate time can be easily modeled by DITEs with empty indeterminate component.

Property: Consistent extension (DI relations). Determinate temporal relations can be modeled by DI relations in which each tuple is associated with a DITE of the form $<d,\{\}>$. ■

C. *Determinate+Indeterminate Algebraic Operations*

For the sake of brevity, only Determinate+Indeterminate Cartesian product ($\times^{DI}$) is defined here. Cartesian product is a polymorphic extension of the definition in Section II.

Definition: DI temporal Cartesian product. Let r and s denote two DI relations, then the Cartesian product of r and s is defined as:

$$r \times^{DI} s = \{ < v_r \cdot v_s \mid t > \mid \exists t_r\ \exists t_s\ (< v_r \mid t_r > \in r \wedge < v_s \mid t_s > \in s \wedge t = t_r \cap^{DI} t_s \wedge t \neq \varnothing) \}$$

where $\cap^{DI}$ denotes the intersection between DITEs. ■

At this point, the set operations of union ($\cup^{DI}$), intersection ($\cap^{DI}$) and difference ($-^{DI}$) between DITEs can be defined.

Definition: $\cup^{DI}$, $\cap^{DI}$, and $-^{DI}$. Given two DITEs $<d_1,i_1>$ and $<d_2,i_2>$,

$$<d_1,i_1> \cup^{DI} <d_2,i_2> = < d_1 \cup d_2,\ i_1 \cup i_2>$$
$$<d_1,i_1> \cap^{DI} <d_2,i_2> = < d_1 \cap d_2,\ (d_1 \cup i_1) \cap (d_2 \cup i_2)>$$
$$<d_1,i_1> -^{DI} <d_2,i_2> = < d_1 - (d_2 \cup i_2),\ (d_1 \cup i_1) - d_2>. ■$$

The following property is essential to ensure that the above representation formalism is suited to be used within a temporal algebra.

Property: Closure of DI set operators. The representation language of DITE is closed with respect to the operations of $\cup^{DI}$, $\cap^{DI}$ and $-^{DI}$. ■

Correctness with respect to the extensional semantics is also an essential feature.

Property: Correctness of set operators. The set operators $\cup^{DI}$, $\cap^{DI}$, and $-^{DI}$ are correct with respect to their underlying semantics: given two DITEs $<d_1,i_1>$ and $<d_2,i_2>$ and indicating by *Ext()* their extensional semantics, we have that:

$$Ext(<d_1,i_1> \cup^{DI} <d_2,i_2>) = Ext(<d_1,i_1>) \cup^{ti} Ext(<d_2,i_2>)$$
$$Ext(<d_1,i_1> \cap^{DI} <d_2,i_2>) = Ext(<d_1,i_1>) \cap^{ti} Ext(<d_2,i_2>)$$
$$Ext(<d_1,i_1> -^{DI} <d_2,i_2>) = Ext(<d_1,i_1>) -^{ti} Ext(<d_2,i_2>). ■$$

Since DITEs with empty indeterminate component reduce to a temporal element, the following property trivially follows.

Property: Consistent extension (set operators on DITEs and relational algebra). If only DITEs of the form $<d,\{\}>$ are used, the operators $\cup^{DI}$, $\cap^{DI}$, and $-^{DI}$ are equivalent (by definition) to $\cup^{ti}$, $\cap^{ti}$, and $-^{ti}$, and the relational algebraic operators $\cup^{DI}$, $-^{DI}$, $\sigma_P{}^{DI}$, $\pi_X{}^{DI}$ and $\times^{DI}$ are equivalent (by definition) to $\cup^t$, $-^t$, $\sigma_P{}^t$, $\pi_X{}^t$ and $\times^t$. ■

IV. COMPACT REPRESENTATIONS

In this section, we sketch a set of alternative approaches that we have defined and analyzed.

A. *Methodology*

We have adopted the same methodology used in Section III: for each representation formalism,

i) we have specified its extensional semantics;
ii) we have analyzed its data expressiveness;
iii) we have defined the set operators, proving their correctness; and
iv) we have checked the closure of such operators.

Note that the methodology we propose to embed such formalisms into a temporal relational algebra is always the same (specified in Section III): we associate the new temporal representation with tuples and relations and we adopt the temporal relational operators defined in Section II in a polymorphic way (i.e., by adopting the proper set operators on the temporal component).

B. Parameters

Several parameters can be considered to propose alternative representations for indeterminate temporal elements. Our choice has been driven by considerations on expressiveness and usefulness.

1. Possibility of expressing, besides indeterminate chronons, also a determinate component in a compact way;
2. Possibility of coping with non-independent indeterminate chronons (i.e., possibility of excluding some of the possible combinations);
3. Possibility of expressing a minimum constraint on the number of chronons;
4. Possibility of expressing a maximum constraint on the number of chronons.

In no way we claim that the parameters we have identified are the only ones worth investigating, and we wish to later consider other parameters.

The first parameter has been already discussed in Section III, considering the <Determinate, Indeterminate> case. The other option we have taken into account is that only indeterminate chronons can be specified, with the same interpretation discussed in Section III.

Also the issue of non-independent indeterminate chronons has been already discussed in Section III. Coping with non-independent indeterminate chronons involves the necessity of representing the possible combinations only, i.e., of listing the desired alternatives. For instance, the mutual exclusion in Ex.3 can be expressed by the set {{}, {1}, {2}} containing the allowed alternative scenarios.

The minimum/maximum constraint is useful in order to explicitly model constraints about temporal duration. For instance, the constraint that ischemic stroke happened in at most one hour (see Ex.2) can be stated by setting the maximum cardinality constraint to 1. It is worth stressing that we intend that the minimum or maximum constraints can be set independently for each tuple, and, in case multiple alternatives are considered, for each alternative.

Since the parameters are orthogonal, meaning that all possible combinations are feasible, we identify sixteen languages to express valid-time indeterminacy, plus the extensional one discussed in Section II.

Notation. In the following, we use short tags to denote the parameters and, thus, the seventeen formalisms. "EA" stands for the extensional approach. "I" stands for the treatment of indeterminate chronons, and "D+I" for the treatment of both determinate and indeterminate chronons. Superscript "*" stands for the possibility of specifying multiple alternatives concerning the indeterminate temporal element (i.e., of coping with non-independent indeterminate chronons). Finally, superscripts "n" and "N" stand for the possibility of expressing minimality and maximality constraints respectively. Combinations of tags represent combinations of parameters.

The seventeen languages thus are EA, I, D+I, I^*, $D+I^*$, I^n, $D+I^n$, $I^{n,*}$, $D+I^{n,*}$, I^N, $D+I^N$, $I^{N,*}$, $D+I^{N,*}$, $I^{n,N}$, $D+I^{n,N}$, $I^{n,N,*}$, and $D+I^{n,N,*}$.

Specifically, D+I represents the approach discussed in Section III, and $D+I^*$ stands for the approach in which multiple alternative indeterminate components can be specified[3].

In the following, we briefly summarize the main results we have obtained.

C. Analysis of the approaches

The first, fundamental, property we consider is closure. In fact, if the temporal representation is not closed with respect to the set operators[4], the relational algebra itself is not closed (indeed, is not an algebra!). Hence, approaches for which closure does not hold are not suitable in the DB context. We have proven that ten of the seventeen approaches are closed.

Property: closure. The formalisms EA, I, D+I, I^*, $I^{n,*}$, $D+I^{n,*}$, $I^{N,*}$, $D+I^{N,*}$, $I^{n,N,*}$, $D+I^{n,N,*}$ are closed with respect to set operators, while the formalisms $D+I^*$, I^n, $D+I^n$, I^N, $D+I^N$, $I^{n,N}$, $D+I^{n,N}$ are not. ■

It is interesting to notice that, even if the language D+I, described earlier, is closed, adding the possibility of listing alternatives concerning indeterminate chronons (i.e., the language $D+I^*$) results in a language that is not closed. Consider the set operator of difference and the following operation in $D+I^*$:

<{1,2}, {}> – <{}, {{1},{2}}>.

The extensional semantics of the result is the DTE *{{1}, {2}, {1,2}},* which is not expressible in $D+I^*$ since it has an empty determinate component (because the temporal elements have no common chronon), but the empty temporal element is not present.

Considering the closed formalisms, we compare their expressiveness in Figure 1. We have proven that four of the nine representational formalisms are as expressive as the extensional approach EA.

Property. The approaches $D+I^{n,*}$, $D+I^{n,N,*}$, $I^{n,*}$, $I^{n,N,*}$ are as expressive as EA. I, D+I, I^*, $I^{N,*}$, $D+I^{N,*}$ are less expressive than EA. ■

This is an important result, meaning that $D+I^{n,*}$, $D+I^{n,N,*}$, $I^{n,*}$ and $I^{n,N,*}$ can express (possibly in a more compact way) all the possible combinations of alternative scenarios. Nevertheless, it is worth recalling that expressiveness is not the only criterion worth to be considered (otherwise EA could suffice). Compactness is also important, as is the degree of effort that users have to make to model their application domains.

For instance, consider Ex.4: it can be expressed in a more compact way in D+I than in $I^{n,N,*}$, even though D+I is less expressive than EA. In fact, on the one hand in D+I it can be expressed —as described in Section 3.1— as *<{1},{2,3}>*. On the other hand, in $I^{n,N,*}$ it can be expressed as the set of alternatives

[3] For instance, D+I* can model Ex.3 as <{}, {{}, {1}, {2}}> where D={} and I*={{}, {1}, {2}}.

[4] Of course, we intend that set operators must be defined in such a way that they are correct with regard to the underlying extensional semantics.

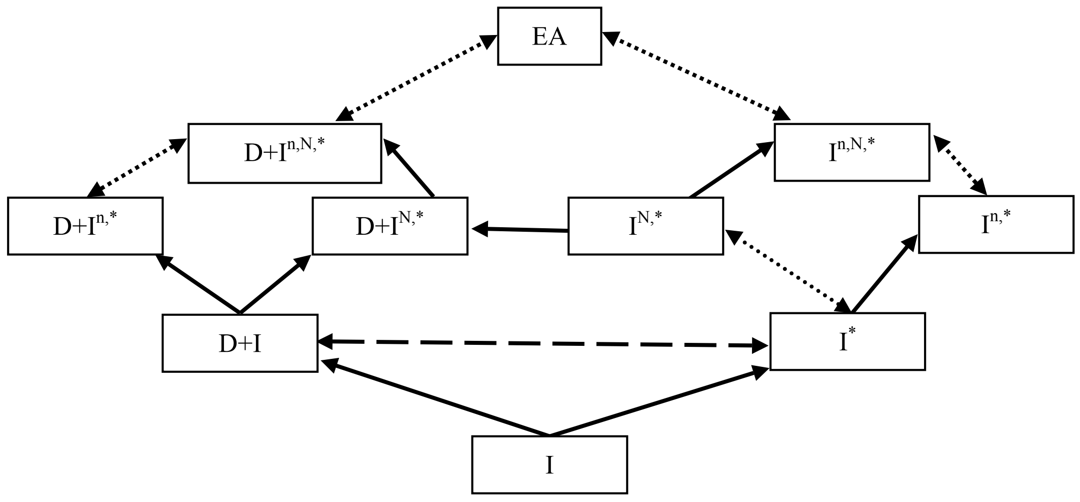

Figure 1. Graphical representation of the data expressiveness of the nine closed languages for representing valid-time indeterminacy studied in our work (plus the extensional approach, EA). Solid lines connect a less expressive to a more expressive language. Dotted lines connect languages with equal expressiveness. The dashed line connects two incomparable languages. Only the main relations are represented.

$$\{\, \{1\}^{\geq 1}, \{1,2\}^{\geq 2}, \{1,3\}^{\geq 2}, \{1,2,3\}^{\geq 3} \,\},$$
containing four alternatives.

It is interesting to notice how the expressiveness changes as we add parameters to a language.

For example, starting from the D+I language described in Section III, if we add the possibility of expressing alternatives concerning the indeterminate component, we derive the language D+I*, which is not closed, as commented above. However, if we add to D+I the possibility of expressing both alternatives concerning the indeterminate component and minimality constraints (parameters (2) and (3) in Section IV.B), we obtain a closed language, D+I^{n,*}, which is strictly more expressive than D+I, and that is as expressive as EA. The alternatives can express disjuncts in a DTE and each temporal element can be represented by imposing the minimum constraint equal to the number of chronons in the temporal element. Obviously all languages that, as EA, can represent any subset of the lattice of alternative scenarios are closed.

If we add to D+I* the possibility to express maximality constraints (obtaining D+I^{N,*}), we obtain a closed language, with different expressive power. As a matter of fact, D+I^{N,*} cannot express arbitrary DTEs since all extensions have to include either the empty temporal element (since the determinate component is empty) or a same temporal element (since the determinate component is not empty).

A further asymmetry can be observed in that the expressiveness of D+I cannot be compared with I*. For example, on the one hand the DTE *{{}, {2}, {3}}* can be expressed by I* as the set *{{2}, {3}}* containing two alternatives, but cannot be expressed by D+I, because –since the empty temporal element is present– the determinate component must be empty, but including the chronons 2 and 3 in the indeterminate component would necessarily include also the temporal element *{2,3}*. On the other hand, we cannot conclude that I* is more expressive than D+I, because, for example, the DTE *{{2}, {2,3}}* is expressible by D+I as *<{2}, {3}>,* but it cannot be expressed by I* because it cannot contain the empty temporal element, which is necessarily contained in all DTEs generated by I*.

In Table 1, the seventeen approaches are summarized and compared considering different parameters. Gray rows are used to mark the languages that are not closed with regard to set operators. The first four columns indicate whether a feature is present or not in the approach. The last three columns state whether it is possible to express a phenomenon in a compact/user-friendly way (e.g., EA allows one to express the minimal duration constraint, but only eliciting all possible cases; thus EA does *not* exhibit the Min property).

TABLE I. COMPARISON OF THE SEVENTEEN APPROACHES.

	Full Expr	**Consist Ext**	**Depen d**	**NO exi st**	**D et**	**M in**	**M ax**
I							
I^n	▨	▨	▨	▨	▨	▨	▨
I^N	▨	▨	▨	▨	▨	▨	▨
I^{n,N}	▨	▨	▨	▨	▨	▨	▨
I*			X				
I^{n,*}	X	X	X	X		X	
I^{N,*}			X				X
I^{n,N,*}	X	X	X	X		X	X
D+I		X		X	X		
D+I^n	▨	▨	▨	▨	▨	▨	▨
D+I^N	▨	▨	▨	▨	▨	▨	▨
D+I^{n,N}	▨	▨	▨	▨	▨	▨	▨
D+I*	▨	▨	▨	▨	▨	▨	▨
D+I^{n,*}	X	X	X	X	X	X	
D+I^{N,*}		X	X	X	X		X
D+I^{n,N,*}	X	X	X	X	X	X	X

More in detail, “Full Expr” indicates whether the approach has the same expressiveness as the extensional approach EA, “Consist Ext” indicates whether the approach is a consistent extension with regard to a determinate valid-time approach, “Depend” indicates whether it is possible to express non-independency between indeterminate chronons (e.g., mutual exclusion), “No Exist” indicates whether it is possible to state that there is *not* existential indeterminacy. “Det” regards the possibility of compactly express determinate chronons, “Min” indicates the possibility of compactly express minimal cardinality constraints (i.e., minimal duration), and “Max” indicates the possibility of expressing maximal cardinality constraint (i.e., maximal duration).

V. CONCLUSIONS

In this paper, we propose a spectrum of approaches to cope with valid-time indeterminacy, analyzing their properties and their suitability to model phenomena in a compact way. From Table 1, it emerges that there are few approaches not suitable to TDB (i.e., $D+I^*$, I^n, $D+I^n$, I^N, $D+I^N$, $I^{n,N}$, $D+I^{n,N}$). Among suitable approaches, $D+I^{n,N,*}$ is the best choice if one wants to reconcile the full data expressiveness of EA with the possibility of coping with both a determinate component and cardinality constraints in a compact way.

However, it is worth noticing that the incorporation of each parameter has its own price. For instance, moving from I to D+I adds the need to cope with two components both in the data representation and in the operations; moving further to $D+I^*$ involves the capability of managing sets of alternatives (data representation) and of pairwise combining them in set operators (which makes the evaluation of relational operators combinatorial).

Therefore, we think that there is not a “best approach” per se: the main contribution of Table 1 is to indicate users and developers the approach “*best suited*” to model the specific application they work with. For example, in case a user needs to compactly express determinate chronons, but s/he does not need to cope with either non-independent indeterminate chronons, or cardinality constraints, Table 1 shows that the D+I approach is the “best suited” one. Actually, the choice of the “best suited” approach might be done by the DB administrator, on the basis of the specific domain/application. Specifically, we envision the development of a user-friendly interface to help administrators in this choice (e.g., by exemplifying the choice criteria summarized in Table 1).

Moreover, the lattice in Figure 1 can be used as a framework to analyze the expressiveness of nascent and (possibly) future approaches in the literature. Few relational TDB approaches have coped with temporal indeterminacy (mostly in the context of incomplete data or probabilistic databases – see the survey in [7]). For the sake of brevity, here we focus on the work of Dyreson and Snodgrass [4]. This approach requires probabilities to be specified. Thus, our family of approaches can apply also to domains where probabilities about time are unknown (this is usually the case, e.g., in the medical context). In [4], valid-time indeterminacy is coped with by associating a period of indeterminacy with a tuple. A period of indeterminacy is a period between two indeterminate instants, each one consisting of a range of chronons and of a probability distribution over it. Since the ranges of chronons defining the starting and ending points of a period cannot overlap, periods of indeterminacy must have at least one “determinate” chronon. Thus, indeterminacy about existence cannot be expressed, and, disregarding probabilities, Snodgrass and Dyreson’s approach is strictly enclosed in our D+I approach as regards expressiveness.

As future work, we aim at extending our approach by extensively applying the methodology used in this paper to consider other parameters, besides the ones described in Section IV.

Additionally, we plan to provide a prototype implementing all of these approaches. Our goal is to develop a highly modular and layered architecture, so that an approach coping with an additional feature (e.g., $D+I^*$) is implemented on top of the simpler one (e.g., D+I).

Finally, our long-term goal is the development of suitable extensions to TSQL2 in order to deal with the different forms of indeterminacy considered in this paper.

REFERENCES

[1] J. F. Allen, “Maintaining knowledge about temporal intervals”, *Communications of the ACM*, v.26 n.11, p.832-843, Nov. 1983.

[2] E. F. Codd, Relational Completeness of Data Base Sublanguages. In: R. Rustin (ed.), *Database Systems* 65-98, Prentice Hall and IBM Research Report RJ 987, San Jose, California, 1972.

[3] J. Dunn, S. Davey, A. Descour, and R.T. Snodgrass, Sequenced Subset Operators: Definition and Implementation, proc. ICDE'02, 2002.

[4] C. E. Dyreson and R. T. Snodgrass, “Supporting Valid-time Indeterminacy”, *ACM Transactions on Database Systems*, 23(1), March 1998, pp. 1-57.

[5] C. Freksa, “Temporal reasoning based on semi-intervals”, *Artificial Intelligence*, 54 (1), pp. 199–227, 1992.

[6] C. S. Jensen and R. T. Snodgrass, “Semantics of Time-Varying Information”, *Information Systems*, 21(4), 311–352, 1996.

[7] C. S. Jensen and R. T. Snodgrass (eds), "Temporal Database Entries for the Springer Encyclopedia of Database Systems," *TimeCenter TR-90*, May 2008, 337+v pages (entries from the *Encyclopedia of Database Systems*, Editors-in-chief: Liu, Ling; Özsu, M. Tamer, Springer, 2009).

[8] A. Krokhin, P. Jeavons, and P. Jonsson, “Reasoning about temporal relations: The tractable subalgebras of Allen's interval algebra”, *Journal of the ACM*, 50(5), p.591-640, 2003.

[9] R.T. Snodgrass (ed), *The TSQL2 Temporal Query Language*, Kluwer, 1995.