An AI Approach to Temporal Indeterminacy in Relational Databases

Luca Anselma[1], Luca Piovesan[2] and Paolo Terenziani[2]

[1] Dipartimento di Informatica, Università di Torino, Corso Svizzera 185, 10139 Torino, Italy
[2] DISIT, Università del Piemonte Orientale "A. Avogadro", Alessandria, Italy
anselma@di.unito.it, luca.piovesan@uniupo.it
paolo.terenziani@uniupo.it

Abstract. Time is pervasive of the human way of approaching reality, so that it has been widely studied in many research areas, including Artificial Intelligence (AI) and *relational* Temporal Databases (TDB). Indeed, while thousands of TDB papers have been devoted to the treatment of determinate time, only few approaches have faced temporal indeterminacy (i.e., "don't know exactly when" indeterminacy). In this paper, we propose a new AI-based methodology to approach temporal indeterminacy in relational DBs. We show that typical AI techniques, such as studying the *semantics* of the *representation formalism*, and adopting *symbolic manipulation* techniques based on such a semantics, are very important in the treatment of indeterminate time in relational databases.

Keywords: Temporal data, data representation and semantics, query semantics, symbolic manipulation

1 Introduction

Time is pervasive of our way of dealing with reality. As a consequence, time has been widely studied in many areas, including AI and DBs. In particular, the scientific DB community agrees that time has a special status with respect to the other data, so that its treatment within a *relational* database context requires dedicated techniques [1, 2]. A plethora of dedicated approaches has been developed in the area of temporal relational databases (TDB in the following; see, e.g., [3, 4]). Different data models, and algebraic operations to query them, have been introduced in the literature. However, to the best of our knowledge, no TDB approach has explicitly identified the fact that, while adding time to a relational DB, one adds ***implicit knowledge*** (i.e., the semantics of time) in it. This is particularly true in case temporal indeterminacy is considered (i.e., "*don't know exactly when*" indeterminacy [5]), since no TDB approach makes all the alternative cases explicit. In this paper we argue that, since a high degree of implicit information is present in temporally indeterminate DB data, a temporal indeterminate DB is indeed close to a (simplified) knowledge base, so that

AI techniques are important to properly cope with it. In this paper, we propose an AI-based methodology to deal with temporal indeterminacy:

(i) We formally define and extend the *snapshot semantics* [2] to cope also with temporal indeterminacy,
(ii) We propose a 1NF ***representation model*** for "*interval-based*" temporal indeterminacy
(iii) We analyse the ***semantics*** of the representation model, showing that (at least) two alternatives are possible
(iv) We define the relational algebraic operators (which perform ***symbolic manipulation*** on the model) to query the representational model, for both the alternative semantics, showing that only with one of them it is possible to devise a relational algebra which is both *closed* with respect to the model and *correct* with respect to the semantics.

Result (iv) enforces the core message of our approach: in TDBs, the representational model contains *implicit* (temporal) information. Thus, AI techniques could\should be used to analyse its *semantics*, and devise *algebraic operators* that perform *symbolic manipulation* on the representational model, *consistently* with the devised semantics.

2 Background

Most TDB approaches focus on individual occurrences of facts, whose time of occurrence (***valid time*** [2]) is exactly known. However, in many real-world cases, the exact time of occurrence of facts is not known, and can only be approximated, so that *temporal indeterminacy* (i.e., in the TDB context, "*don't know exactly when*" indeterminacy [5]) has to be faced. Temporal indeterminacy is so important that "support for temporal indeterminacy" was already one of the eight explicit goals of the data types in TSQL2 consensus approach [2]. Despite its importance, and differently from the area of AI, in the area of TDBs only few approaches coping with temporal indeterminacy have been devised (see the surveys in [5, 6]).

Dyreson and Snodgrass [7] cope with valid-time indeterminacy 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 granules and of a probability distribution over it. However, in [7], no relational algebra is proposed to query temporally indeterminate data. Dekhtyar et al. [8] introduce temporal probabilistic tuples to cope with a quite specific form of temporal indeterminacy, concerning instantaneous events only, and provide algebraic relational operators. Anselma et al. [9, 10] identify different forms of temporal indeterminacy, and propose a family of achievable representational models and algebrae. However, such an approach is semantic-oriented, abstract and not in 1NF (thus not suitable for a direct implementation). A 1NF approach for a form of temporal indeterminacy has been proposed in [11], but no semantics for the model has been presented.

3 Snapshot semantics for temporal relational databases

A premise is very important, when starting a discussion about the *semantics* of temporal DBs. Indeed, seen from an AI perspective, a "traditional" non-temporal database is just an elicitation of all and only the facts that are true in the modeled mini-world. In such a sense, the semantics of a non-temporal DB is "trivial", since the DB does not contain any implicit data\information. Since the data is explicit, no "AI-style" reasoning mechanism is required, and query operators are used just to *extract* the relevant data from a DB. However, such an "easy" scenario changes when *time* is introduced into DBs, to associate each fact with the time when it holds (usually called *valid time* [2]). Roughly speaking, in such a case, eliciting explicitly all true facts would correspond to elicit, for each possible unit of in time, all the facts that hold at that unit. Despite the extreme variety of TDB approaches in the literature, almost the totality of them is based, explicitly or (in many cases) implicitly, on this idea, commonly termed "*snapshot semantics*": a TDB is a set of "standard" (non-temporal) DBs, each one considering a snapshot of time, and eliciting all facts (tuples) that hold at that time (see, e.g., the "consensus" BCDM semantics, which is the semantics for TSQL2 and for many other TDB approaches [2]). Of course, for space and time efficiency reasons, no approach in the literature directly implements TDBs making all such data explicit: representational models are used to encode facts in a more compact and efficient form. Notably, this is a dramatic departure from "traditional" DB concepts: a *temporal* DB is no more an elicitation of all facts that hold in the modelled mini-world, but a compact ***implicit*** representation of them. Therefore, in this paper, we propose that the following "AI-style" methodological requirements must be taken into account. First,

(M1) *a semantics for making explicit the intended meaning of the representational models must be devised*.

In such a context, the algebraic query operators cannot simply select and extract data (since some data are implicit). Making all data explicit before\while answering queries is certainly not a good option (for the sake of space and time efficiency). Thus

(M2) *algebraic operators must operate on the (implicit) representation*

(M3) *algebraic operators must provide an output expressed in the given representation (i.e., the representation formalism must be closed with respect to the algebraic operators)*

(M4) *algebraic operators must be correct with respect to the semantics of the representation*

In the rest of this section, we provide a new "*functional*" way to describe the snapshot semantics for *determinate* time TDBs, that we later extend to *indeterminate* time in Section 3, as a starting point to realize the above AI-style methodology.

2.1 Data Semantics of Determinate Time DBs: a "Functional" Perspective

We first introduce the notion of tuple, relation, and database. We then move to the definition of time, and define the notion of (semantics of) a temporal database.

Definition 1. (non-temporal) Database, Relation, Tuple. A (non-temporal) relational database *DB* is a set of relations over the relational schema $\sigma = (R_1{:}s_i , ..., R_k{:}s_j)$ where $s_i, ..., s_j \in S$ are the sorts of $R_1, ..., R_k$, respectively. A relation $R(x_1, ..., x_k){:}s$ of sort $s \in S$ is a sequence of attributes $x_1, ..., x_k$ each with values in a proper domain $D_1,\ldots D_k$. An instance $r(R{:}s)$ of a relation $R(x_1, ..., x_k)$ of sort $s \in S$ is a set $\{a_1, \ldots, a_n\}$ tuples, where each tuple a_i is a set $<v_1,\ldots,v_k>$ of values in $D_1\times\ldots\times D_k$. ■

Notation. In the following, we denote by DB_σ the domain of all possible database instances over a schema σ. ■

In AI, the ontology of time has attracted a lot of attention, and many different possibilities have been investigated. Some approaches, for instance, consider both points and intervals as basic time units (see, e.g., [12]), while in other approaches time points exist only as interval boundaries (see, e.g., [13]). Another important distinction regards time density: time can be represented as discrete, dense or continuous. Finally, time can be linear or branching. The review in [14] discusses in detail such aspects and compares the approaches coping with time in ontologies.

On the other hand, most TDB approaches, including TSQL2 [2], and the BCDM "consensus" semantics [2], simply assume that time is linear, discrete and bounded, and term *chronon* the basic time unit.

Definition 2. Temporal domain D_T. We assume a limited precision for time, and call *chronon* the basic time unit. The domain of chronons is finite, and totally ordered. The domain of valid times D_T is given as a set $D_T=\{c_1,\ldots,c_k\}$ of chronons. ■

In the *snapshot semantics* [2], a TDB is a set of conventional (non-temporal) databases, one for each chronon of time. We formalize such a semantics through the introduction of a function, relating chronons with (non-temporal) databases.

Definition 3. Temporal database (semantic notion). Given a relational schema $\sigma = (R_1{:}s_i , ..., R_k{:}s_j)$ a temporal database DB^T is a function f_{σ,D_T}: $D_T \rightarrow DB_\sigma$ ■

Analogously, a temporal relation r^T is a function from D_T to the set of tuples of r^T that hold at each chronon in D_T.

Definition 4. Time slice. Given a temporal database DB^T and a temporal relation $r^T \in$ in DB^T, and given a chronon $c \in \boldsymbol{D_T}$, we define the time slice of DB^T (denoted by $DB^T(c)$) and of r^T (denoted by $r^T(c)$) the result of the application of the functions DB^T and r^T to the chronon c. ■

Example. 1. Let us consider a simple database DB^T_1 modeling patient symptoms. DB^T_1 contains a unique relation SYM of schema <Patient,Symptom,Value> and contains two facts:

(f1) John had high fever from 10 to 12

(f2) Mary had moderate fever from 11 to 13

(in the example, we assume that chronons are at the granularity of hours, and hour 1 represents the first hour of 1/1/2018). The TDB (semantic notion) modeling such a state of affairs is the following (for clarity and simplicity, we omit the chronons in D^T for which no tuple holds, and we omit the name of the relation(s)).

10 → {<John, fever, high>}
11 → {<John, fever, high>,<Mary, fever, moderate>}
12 → {<John, fever, high>,<Mary, fever, moderate>}
13 → {<Mary, fever, moderate>}

In this example $DB^T_1(10) = SYM^T(10) = \{<John, fever, high>\}$ ■

Notably, Definition 3 above is a purely "semantic" definition. Other definitions of the snapshot semantics for TDBs, such as the one in the "consensus" BCDM [2] model, are more "operational" and are closer to actual representations[1].

2.2 Query semantics

In TDBs, the *semantic of queries* is commonly expressed by specifying in terms of ***relational algebraic operators***. 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 ($\times$). Though different approaches have generalized such operators to cope also with TDBs, there is a common agreement that such operators should be a *consistent extension* of standard Codd's operators, and that they should be *reducible* to them in case time is removed (see, e.g., [2, 15]). In other words, temporal algebraic operators should behave exactly as Codd's non-temporal ones, at each point (chronon) of time. Given our definitions above, such a requirement can be formally stated as below.

Definition 5. **Relational algebraic operators on determinate time databases ("semantic" notion).** Denoting by Op^C a Codd's operator, and by Op^T its corresponding temporal operator, Op^T must be defined in such a way that the following holds: $\forall c \in D_T \ (Op^T(r^T,s^T)\ (c) = Op^C(r^T(c),s^T(c)))$ ■

(In Definition 5 above, we assume that r^T and s^T are temporal relations in a temporal database DB^T, and that Op is a binary operator. $r^T(c)$ represents the *time slice* of r^T at the chronon *c*. The definition of unary operators is analogous).

Of course, the "purely semantic" definition above is highly inefficient, as snapshot(s) of the underlying relation(s) at each single chronon are computed. Thus, more "operational" definitions of algebraic operators have been proposed in the literature. Notably, however, the "commonly agreed" BCDM definition of the semantics of algebraic operators is consistent with Definition 5 above.

2.3 Implementations of (determinate time) temporal databases

Different realizations of determinate time TDBs have been proposed in the literature. All of them (except few "pioneering" approaches) respect the above data and query semantics, and provide an efficient implementation for it. The large majority of such approaches enforce at least two key requirements to achieve efficiency: (i) *1NF is*

[1] Indeed, the most common way of presenting the semantics of a temporal database is the one in BCDM, in which each tuple is paired with all the chronons when it holds. In BCDM, temporal databases directly associate times with tuples, so that the semantics of Example 1 above would be modeled as follows: {<John, fever, high, {10,11,12}>, <Mary, fever, moderate, {11,12,13}>.

used to represent data, (ii) temporal algebraic operators directly manipulate the representation.
In Section 3 we extend the semantic framework introduced so far to provide the general semantics of temporal indeterminacy in TDBs. Then, in Section 4, we move to a representational model, considering the requirements (i) and (ii) above, and following the methodological requirements (M1-M4) identified in Section 2.

3 Snapshot semantics of temporal indeterminacy in TDB

In TDBs, the notion of temporal indeterminacy is usually paraphrased as "*don't know exactly when*" indeterminacy (consider, e.g., the Encyclopedia survey in [5]): facts hold at times that are *not exactly* known. An example is reported in the following:

Example 2. As a running example, let us consider a simple database DB^{IT}_1 modeling patient symptoms. The database contains a unique relation SYM^{IT} of schema <Patient,Symptom,Value> and models two facts:

(f1) John had high fever at 10 and 11, and possibly at 12, or 13, or both.

(f2) Mary had moderate fever at 12 and 13, and possibly at 11.

(In the example, we assume that chronons are at the granularity of hours, and hour 1 represents the first hour of 1/1/2018).

3.1 Data Semantics of Indeterminate Time DBs

Of course, we can still retain the definition of the temporal domain D_T provided in Section 2. However, the definition of an indeterminate temporal database is different: informally speaking, an indeterminate TDB is simply a set of alternative determinate-time TDBs, each one encoding one of the different possibilities. Technically speaking, such a definition requires the introduction of a set of functions.

Definition 6. Indeterminate temporal database (semantic notion). Given a relational schema $\sigma = (R_1{:}s_i , ..., R_k{:}s_j)$, an indeterminate temporal database DB^T is a set $S(DB^{IT})=\{f^1, \ldots, f^k\}$ of functions f^i_{σ,D_T}: $D_T \rightarrow DB_\sigma$ ■

Analogously, a temporally indeterminate relation r^{IT} is a set $S(r^{IT})$ of functions from D_T to the set of tuples of r^T that hold at each chronon in D_T.

As an example, eight functions are necessary to cover all the alternative possibilities (henceforth called ***scenarios***) for Example 2.

Example. 2 (cont).
The indeterminate temporal database DB^{IT} (semantic notion) modeling Example 2 consists of a unique relation SYM^{IT} and is shown in the following (for the sake of brevity, we denote with "J" the tuple <John, fever, high> and with "M" the tuple <Mary, fever, moderate>).

f^1	f^2	f^3	f^4
10 → {J}	10 → {J}	10 → {J}	10 → {J}
11 → {J}	11 → {J}	11 → {J}	11 → {J}
12 → {M}	12 → {J,M}	12 → {M}	12 → {J,M}
13 → {M}	13 → {M}	13 → {J,M}	13 → {J,M}

f^5	f^6	f^7	f^8
10 → {J}	10 → {J}	10 → {J}	10 → {J}
11 → {J,M}	11 → {J,M}	11 → {J,M}	11 → {J,M}
12 → {M}	12 → {J,M}	12 → {M}	12 → {J,M}
13 → {M}	13 → {M}	13 → {J,M}	13 → {J,M} ■

For the technical treatment that follows, it is useful to introduce the notion of *scenario* slice, which "selects" a specific scenario.

Definition. Scenario slice. Given an indeterminate temporal database $DB^{IT} = \{f^1,...,f^k\}$ and a temporal relation $r^{IT} \in DB^{IT}$, and given any $f \in \{f^1,...,f^k\}$, we define the *scenario* slice f of DB^{IT}(denoted by DB_f^{IT}) and of r^{IT} (denoted by r_f^{IT}) the determinate temporal database and the determinate temporal relation obtained by considering only the alternative f for DB^{IT} ■

Example 3. For example, considering Example 2 above, and the scenario f^1, $DB_{f^1}{}^{IT}=SYM_{f^1}{}^{IT}$= {10→{J}, 11→{J}, 12→{M}, 13→{M}}. ■

3.2 Query semantics

Of course, for the algebraic query operators, we can still retain all the general requirements discussed so far for determinate time. However, we have to generalize the above approach, to consider the fact that a set of *alternative* (determinate) temporal databases (*scenarios*) are involved. Therefore, given two temporally indeterminate relations r^{IT} and s^{IT}, binary temporal algebraic operators must consider, at each chronon, all the possible combinations of the scenarios $f_r \in S(r^{IT})$ of r^{IT} and $f_s \in S(s^{IT})$ of s^{IT}.

Definition 8. **Relational algebraic operators on indeterminate temporal databases ("semantic" notion).** Denoting by Op^C a Codd's operator, and by Op^{IT} its corresponding temporal operator for indeterminate time Op^{IT} must be defined in such a way that the following holds

$$\forall c \in D_T \; (Op^T(r^T,s^T)\,(c) = \cup_{f_r \in S(r^{IT}) \wedge f_s \in S(s^{IT})} Op^C(f_r(c), f_s(c))) \quad ■$$

(In Definition 5, r^{IT} and s^{IT} are temporal relations in a temporally indeterminate database DB^{IT}, and Op is a binary operator. $f_r(c)$ represents the *time slice* at the chronon c of the scenario f_r of r^{IT}. The definition of unary operators is simpler).

We regard Definition 8 as one of the major results of this paper: until now, no approach in the TDB community has been able to clarify the semantics of temporal algebraic operators on indeterminate time in terms of their Codd's counterparts. But, obviously, this is just *data* and *query semantics*: a direct implementation of the data model and algebraic operators defined so far would be highly inefficient, as regard

both space and time. As a consequence, "compact" representational models and operators on them should be identified. We address this issue in the next section.

4 Possible "compact" approaches to temporal indeterminacy

The most frequently adopted representational model to cope with (valid) time in a compact and 1NF way is the interval-based representation (consider, e.g., the TSQL2 "consensus" representational model [2]). A time interval (compactly modelled by a starting and an ending time) is associated with each temporal tuple, to denote that the (fact represented by the) tuple holds in each chronon in the interval. In the indeterminate time context, such an interval-based representation has also been used, e.g., in [7, 11, 16–18]. As in such approaches, we associate four temporal attributes (say T1, T2, T3, and T4) with each temporal tuple, to compactly represent the intervals when it certainly and possibly holds.

Definition 8. Temporally indeterminate Database, Relation, Tuple (representational model). A temporally indeterminate relational database DB^{IT} is a set of (temporally indeterminate) relations over the relational schema $\sigma = (R_1{:}s_i, ..., R_k{:}s_j)$ where $s_i, ..., s_j \in S$ are the sorts of $R_1, ..., R_k$, respectively. A relation $R(x_1, ..., x_k|T1,T2,T3,T4){:}s$ of sort $s \in S$ is a sequence of non-temporal attributes $x_1, ..., x_k$ each with values in a proper domain $D_1,\ldots D_k$, and temporal attributes *T1, T2, T3, T4 with domain* D_T. An instance $r(R{:}s)$ of a relation $R(x_1, ..., x_k|T1,T2,T3,T4){:}s$ is a set $\{t_1, \ldots, t_n\}$ tuples, where each tuple t_i is a set $<v_1,\ldots,v_k|t_1,t_2,t_3,t_4>$ of values in $D_1\times\ldots\times D_k\times D_T\times D_T\times D_T\times D_T$. ■

Example 4. In the temporally indeterminate context, the relation SYM (called SYM^{IT}) may be represented with the schema <Patient,Symptom,Value| T_1,T_2,T_3,T_4> . Tuples of SYS are shown in Examples 5 and 7 below ■

Intuitively and ***roughly*** speaking, the semantics of such a compact 1NF "interval-based" representation of temporal indeterminacy is the following:

(sem1) the fact represented by the tuple **$<v_1,\ldots,v_k|t_1,t_2,t_3,t_4>$** occurs possibly in the (chronons in the) time intervals $[t_1,t_2)$ and $[t_3, t_4)$, and certainly in the time interval $[t_2,t_3)$.

We now show that an "informal" semantics like (sem1) above is not enough: it must be fully formalized as a starting point for devising a "proper" representational model and algebra, following the methodological requirements M1-M4 above.

4.1 "Single occurrence" semantics

A first way of interpreting the "ambiguous" semantics (sem1) above is formally described in Definition 9 below. For the sake of space constraints, in Definition 9 we adopt a compact notation to represent scenarios: given a temporally indeterminate tuple with non-temporal part v, we denote by $v([c_1,c_2])$ the scenario $\{c_1\rightarrow\{v\}, c_1+1\rightarrow\{v\}, \ldots, c_2\rightarrow\{v\}\}$.

Definition 9. Representation semantics (sem1'). The semantics of an indeterminate time tuple $\langle v|t_1,t_2,t_3,t_4\rangle$ in the representational model in Definition 8 is the set of scenarios

$\{v([t_2,t_3-1]), v([t_2,t_3]),v([t_2,t_3+1]),v([t_2,t_3+2]), \ldots, v([t_2,t_4-1]),$
$v([t_2-1,t_3-1]), v([t_2-1,t_3]),v([t_2-1,t_3+1]),v([t_2-1,t_3+2]), \ldots, v([t_2-1,t_4-1]),$
$v([t_2-2,t_3-1]), v([t_2-2,t_3]),v([t_2-2,t_3+1]),v([t_2-2,t_3+2]), \ldots, v([t_2-2,t_4-1]), \ldots,$
$v([t_1,t_3-1]), v([t_1,t_3]),v([t_1,t_3+1]),v([t_1,t_3+2]), \ldots, v([t_1,t_4-1])\}$ ■

In Definition 9, we formalize that the fact *v* occurred in a ***convex*** (i.e., with no gap) time interval, which includes all the chronons in $[t_2,t_3)$, and may extend forward until chronon t_4 (excluded) and backward until chronon t_1. This is, probably, the most intuitive notion of temporal indeterminacy in TDBs: each tuple represents ***a single occurrence*** of a fact, and temporal indeterminacy concerns the starting and ending chronons of it. In such a context, it looks natural to impose $t_1 \leq t_2 < t_3 \leq t_4$, thus granting that there is at least one chronon in which the fact certainly occurs (see, e.g., [7]).

Example 5. Given the temporally indeterminate relation SYM^{IT}, with the semantics (sem1') above, the fact

(f2) Mary had moderate fever at 12 and 13, and possibly at 11

can be represented by the tuple <Mary,fever,moderate|11,12,14,14>.
The semantics of such a tuple consists of two possible scenarios:

	11 → {M}
12 → {M}	12 → {M}
13 → {M}	13 → {M} ■

Notably, if we assume the semantics (sem1'), the fact (f1)

(f1) John had high fever at 10 and 11, and possibly at 12, or 13, or both

cannot be represented in the representational model: as a matter of fact, the tuple

<John,fever,high|10,10,12,14>

would be interpreted as the compact representation of the semantics below:

10 → {J}	10 → {J}	10 → {J}
11 → {J}	11 → {J}	11 → {J}
	12 → {J}	12 → {J}
		13 → {J}

while the scenario <10 → {J}, 11 → {J}, 13 → {J}> would not be part of the semantics of the representation. Indeed, if we assume (sem1'), each tuple represents a single occurrence of a fact, while the latter scenario above represents two separate occurrences, one at [10,12), and one at [13,14).

Of course, the specification of the semantics is fundamental also for the definition of the algebraic operators. In particular, we must grant that such operators (i) are correct wrt the semantics, and (ii) are closed wrt the representational model.
Notably, if we assume the semantics (sem1') for the representational model in Definition 8, there is no way to satisfy both requirements (i) and (ii)[2]. A trivial counterexample is discussed in the following, considering algebraic difference.

[2] Notably, it is possible to show that it is not possible to define correct algebraic operators closed with respect to the representational model also in case one admits the possibility that facts in the TDBs do not necessarily occur, i.e., imposing $t_1 \leq t_2 \leq t_3 \leq t_4$ in the representational model. We cannot show such a generalization here, for the sake of space constraints.

Example 6. Consider the difference between two relations $r1^{IT}$ and $r2^{IT}$ having the same schema $(A_1,\ldots,A_k|T_1,T_2,T_3,T_4)$. Let $r1^{IT}=\{<a_1,\ldots,a_k|1,3,5,7>\}$ and $r2^{IT}=\{<a_1,\ldots,a_k|3,3,8,8>\}$ (i.e., the two tuples are value-equivalent, and the tuple in $r2^{IT}$ is determinate, starts at 3 and ends at 7). In such a case the result of the difference $r1^{IT}-^{IT}\ r2^{IT}$ should be a fact $a_1,\ldots,a_k$ which may not occur, or occurs in {2}, or in {1,2}. A tuple with such a semantics cannot be represented in the given representation. Thus, this example suffices to show that (the semantically correct) difference is not closed with respect with the given formalism (with the semantics (sem1') above). ■

4.2 "Independent chronons" semantics

A different way of interpreting the "rough" semantics (sem1) above is provided in Definition 10 where, for the sake of space constraints, we adopt the following compact notation to represent scenarios: given a temporally indeterminate tuple with non-temporal part v, we denote by $v(\{c_1,c_2,\ldots,c_k\})$ the scenario $\{c_1\rightarrow\{v\}, c_2\rightarrow\{v\}, \ldots, c_k\rightarrow\{v\}\}$; furthermore, we denote by $P^S(A)$ the power set of a set A.

Definition 10. Representation semantics (sem1'). The semantics of an indeterminate time tuple $<v|t_1,t_2,t_3,t_4>$ in the representational model in Definition 8 is the set of scenarios $v(\{t_2,t_2+1,t_2+2,\ldots,t_3-1\}\cup T \setminus T\in P^S(\{c \setminus c\in([t_1,t_2)\cup[t_3,t_4))\})$ ■

In such a semantics, there is no notion of single occurrence at all. *v* certainly holds in each chronon in $[t_2,t_3)$ (if any), and may hold in each one of the chronons c in $[t_1,t_2)$ and in $[t_3,t_4)$, independently of each other. In such a context, it is natural to impose $t_1\le t_2\le t_3\le t_4$, so that the fact may also not be certain in a chronon, in case $t_2=t_3$.

Example.7 Given the temporally indeterminate relation SYM^{IT}, with the semantics (sem1') above, the fact (f1)
(f1) John had high fever at 10 and 11, and possibly at 12, or 13, or both is represented in the representational model by the tuple
<John,fever,high|10,10,11,13>
which has the semantics discussed above (in short, the fact may hold at {10,11}, or at {10,11,12}, or at {10,11,13}, or at {10,11,12,13}). ■

With such a semantics for the representational model, it is possible to define correct and closed algebraic operators as follows:

Definition 11. **Algebraic operators for indeterminate time (independent chronons semantics)**. Let r and s denote relations of the same sort and $<v|t_1,t_2,t_3,t_4>$ a tuple with non-temporal part v and temporal part t_1,t_2,t_3,t_4.

$r \cup^{IT} s = \{ < v| t_1,t_2,t_3,t_4> | < v| t_1,t_2,t_3,t_4 > \in r \vee < v| t_1,t_2,t_3,t_4 > \in s\}$

$r \times^{IT} s = \{ < v_r \cdot v_s| t_1,t_2,t_3,t_4 > | \exists\, t'_1,t'_2,t'_3,t'_4\ \exists\, t''_1,t''_2,t''_3,t''_4\ (< v_r| t'_1,t'_2,t'_3,t'_4> \in r \wedge < v_s| t''_1,t''_2,t''_3,t''_4 > \in s \wedge t_1=\max(t'_1, t''_1) \wedge t_4=\min(t'_4, t''_4) \wedge t_1\le t_4 \wedge$
let $t_s=\max(t'_2, t''_2) \wedge t_e=\min(t'_3, t''_3)$
if $t_s\le t_e$ then $t_2=t_s \wedge t_3=t_e$ else $t_2=t_3=t$ where t is any value in $[t_1,t_4)\}$

$\pi^{IT}{}_X(r) = \{ < v| t_1,t_2,t_3,t_4 > | \exists v_r,t_1,t_2,t_3,t_4\ (< v_r| t_1,t_2,t_3,t_4 > \in r \wedge v = \pi_X(v_r) \}$

$\sigma^{IT}{}_P(r) = \{ < v|t > | < v|t > \in r \wedge P(v) \}$

$r -^{IT} s = \{ < v| t_1,t_2,t_3,t_4> | (\exists v, t_1,t_2,t_3,t_4\ (< v| t_1,t_2,t_3,t_4 > \in r \wedge$

$$\neg\exists\, t'_1,t'_2,t'_3,t'_4\ (\ <v|\ t'_1,t'_2,t'_3,t'_4> \in s\ ))) \vee$$
$$(\exists\, t'_1,t'_2,t'_3,t'_4\ \exists\, t''_1,t''_2,t''_3,t''_4\ (<v|\ t'_1,t'_2,t'_3,t'_4> \in r\ \wedge <v|\ t''_1,t''_2,t''_3,t''_4> \in s$$
$$\wedge\ t_1,t_2,t_3,t_4 = \text{difference}([t'_1,t'_4),\ [t'_2,t'_3),\ [t''_1,t''_4),[t''_2,t''_3))\}$$

where difference can be defined by the following function (where *s* is a function that returns the starting point of an interval and *e* returns the ending point, and the function *Nor* is used to reformat the output in case $t_2>t_3$, i.e., $Nor(<t_1,t_2,t_3,t_4>)=<t_1,t_2,t_3,t_4>$ if $t_1\le t_2\le t_3\le t_4$,
$Nor(<t_1,t_2,t_3,t_4>)=<t_1,t,t,t_4>$ where $t_1\le t\le t_4$ if $t_2>t_3$)

difference(p1,n1,p2,n2)

(1) if (p1 ⊆ n2) then return ∅
(2) else if (p1 ∩ n2 = ∅) then return {Nor(<s(p1),s(n1-p2),e(n1-p2),s(p1)>)}
(3) else if (p1 ⊃ n2) then return {Nor(<s(p1),s(n1-p2),e(n1-p2),s(n2)>), Nor(<e(n2),s(n1-p2),e(n1-p2),e(p1)>)}
(4) else return {Nor(<s(p1-n2),s(n1-p2),e(n1-p2),e(p1-n2)>)} ■

The difference function accepts as parameters two time intervals for the minuend (p1 and n1) and two time intervals for the subtrahend (p2 and n2). p1 and p2 are the possible intervals, i.e., they contain the chronons that are in at least one scenario, and n1 and n2 are the necessary –certain– intervals, i.e., they contain the chronons that are in every scenario (thus n1⊆p1 and n2⊆p2). The function operates along the following idea (for space constraints, we will not go into the details): if a chronon is both in the minuend and in the subtrahend, and in the subtrahend such a chronon is (i) necessary (i.e., it belongs to n2), it will not be in the result, (ii) only possible (i.e., it belongs to p2 but not to n2), it will be possible in the result. From (i) and the fact that n1⊆p1, descends line (1) of the difference function, from (ii) descends line (2), from (i) and (ii) and the fact that n2⊈p1 descends line (3), from (i) and (ii) and the fact that n2⊆p1 descends line (4) and, in particular, since n2⊆p1 the minuend "breaks" into two (pairs of) intervals.

Property. The algebraic operators in Definition 11 are correct (with respect to the semantics defined so far) and are closed with respect to the representational model.

5 Conclusions and future work

In this paper, we propose an innovative approach in which a semantic-based AI-style methodology is proposed to cope with temporal indeterminacy in TDBs. Specifically:

(1) We propose a new semantic definition for indeterminate time in TDBs, in which the semantics of algebraic operators can be expressed in terms of their Codd's counterparts (thus formally providing a "*snapshot semantics*" for indeterminate time TDBs).
(2) We propose a new AI-style methodology to the treatment of TDBs, using it to develop a semantically-grounded 1NF approach (data model plus algebra) to cope with "interval-based" temporal indeterminacy.

Indeed, in this paper we have shown that, when introducing the temporal dimension, TDBs have to cope with ***implicit*** information, which has to be ***symbolically manipulated*** by algebraic operators to answer queries. As a consequence, we propose

an innovative AI-based methodology to cope with time in relational DBs. We are confident that our methodology can be fruitfully applied to other types of temporal information in TDBs (e.g., implicit representation of periodically repeated data [19, 20]), and possibly of other forms of indeterminacy, thus leading to a new AI stream of research to cope with *indeterminate\implicit* data in relational DBs.

References

1. Snodgrass, R.T.: Developing Time-oriented Database Applications in SQL. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2000).
2. Snodgrass, R.T.: The TSQL2 temporal query language. Kluwer (1995).
3. Wu, Y., Jajodia, S., Wang, X.S.: Temporal database bibliography update. In: Etzion, O., Jajodia, S., and Sripada, S. (eds.) Temporal Databases: Research and Practice. pp. 338–366. Springer Berlin Heidelberg, Berlin, Heidelberg (1998).
4. Liu, L., Özsu, M.T. eds: Encyclopedia of database systems. Springer (2009).
5. Dyreson, C.: Temporal Indeterminacy. In: Liu, L. and Ozsu, M.T. (eds.) Encyclopedia of Database Systems. pp. 2973–2976. Springer US, Boston, MA (2009).
6. Jensen, C.S., Snodgrass, R.T.: Semantics of Time-Varying Information. INFORMATION SYSTEMS. 21, 311–352 (1996).
7. Dyreson, C.E., Snodgrass, R.T.: Supporting valid-time indeterminacy. ACM Transactions on Database Systems (TODS). 23, 1–57 (1998).
8. Dekhtyar, A., Ross, R., Subrahmanian, V.: Probabilistic temporal databases, I: algebra. ACM Transactions on Database Systems (TODS). 26, 41–95 (2001).
9. Anselma, L., Terenziani, P., Snodgrass, R.T.: Valid-time indeterminacy in temporal relational databases: A family of data models. In: Proc. TIME. pp. 139–145. IEEE (2010).
10. Anselma, L., Terenziani, P., Snodgrass, R.T.: Valid-Time Indeterminacy in Temporal Relational Databases: Semantics and Representations. IEEE Transactions on Knowledge and Data Engineering. 25, 2880–2894 (2013).
11. Anselma, L., Piovesan, L., Terenziani, P.: A 1NF temporal relational model and algebra coping with valid-time temporal indeterminacy. Journal of Intelligent Information Systems. 47, 345–374 (2016).
12. Hobbs, J.R., Pan, F.: An ontology of time for the semantic web. ACM Transactions on Asian Language Information Processing. 3, 66–85 (2004).
13. Baumann, R., Loebe, F., Herre, H.: Axiomatic theories of the ontology of time in GFO. Applied Ontology. 9, 171–215 (2014).
14. Ermolayev, V., Batsakis, S., Keberle, N., Tatarintseva, O., Antoniou, G.: Ontologies of Time: Review and Trends. IJCSA. 11, 57–115 (2014).
15. McKenzie, L.E., Jr., Snodgrass, R.T.: Evaluation of Relational Algebras Incorporating the Time Dimension in Databases. ACM Comput. Surv. 23, 501–543 (1991).
16. Anselma, L., Bottrighi, A., Montani, S., Terenziani, P.: Extending BCDM to Cope with Proposals and Evaluations of Updates. IEEE Transactions on Knowledge and Data Engineering. 25, 556–570 (2013).
17. Anselma, L., Stantic, B., Terenziani, P., Sattar, A.: Querying now-relative data. Journal of Intelligent Information Systems. 41, 285–311 (2013).
18. Anselma, L., Piovesan, L., Sattar, A., Stantic, B., Terenziani, P.: A Comprehensive Approach to "Now" in Temporal Relational Databases: Semantics and Representation. IEEE Transactions on Knowledge and Data Engineering. 28, 2538–2551 (2016).
19. Terenziani, P.: Irregular Indeterminate Repeated Facts in Temporal Relational Databases. IEEE Transactions on Knowledge and Data Engineering. 28, 1075–1079 (2016).
20. Terenziani, P.: Nearly Periodic Facts in Temporal Relational Databases. IEEE Transactions on Knowledge and Data Engineering. 28, 2822–2826 (2016).