Mariusz Giero  
University of Białystok

QUERYING TEMPORAL DATABASE  
WITH THE LANGUAGE  
OF FIRST-ORDER TEMPORAL LOGIC¹

1. Introduction

A temporal database [Etz, Ste, Tan] is defined as a database maintaining object histories, i.e., past, present, and possibly future data. There are numerous application domains dealing with temporal data: Medical Systems (e.g. patient’s records), Computer Applications (e.g. history of file back ups), Archive Management Systems (e.g. sporting events, publications and journals), Reservation Systems (e.g. when was a flight booked) and many others [Sno]. Support for time-varying data within a traditional relational database is not straightforward. There have been more than two dozens extended relational data models proposed [JenSno]. Time-varying data is commonly represented by timestamping values [JenSno, Jen]. Timestamps can be time points, intervals or a set of intervals and can be added to tuples or attributes. There are also different considerations of what time stamps represent: valid time, i.e., time when data (tuple) is true in the universe of discourse, transaction time, i.e., time when data is stored in a database or both time references together.

In this paper we consider a temporal database model with tuple timestamping. Tuples are timestamped by a set of intervals which represent valid time.

¹ The research reported in this paper is part of the project entitled Temporal Representation of Knowledge and Its Implementation in the Computer Systems of Medical Conducts supported by the Polish Ministry of Science and Higher Education, grant no. 3 T11F 011 30.
2. Structure of Time

In this paper, we assume that the flow of time \((T, <)\) is a linear, discrete and ordered structure with no end points. \(T\) is a set of time points and \(<\) is a binary order relation defined on \(T\) which satisfies the following conditions:

- transitivity \(\forall x, y (x < y \land y < z \rightarrow x < z)\)
- irreflexivity \(\forall x \neg (x < x)\)
- totality \(x = y\) or \(x < y\) or \(y < x\), where \(x, y, z \in T\)

3. Relational Database

The relational data model was introduced in the 1970s by E. F. Codd [Cod, Dat]. Currently, it is the most widespread data model used for database applications. Formally, it can be defined as follows:

**Definition 1.**

A *relational database schema* is a quintuple \(S = (R, A, D, \text{attr}, \text{dom})\), where:

- \(R = \{R_1, \ldots, R_k\}\) is a set of relation names,
- \(A = \{A_1, \ldots, A_n\}\) is a set of attribute names,
- \(D = \{D_1, \ldots, D_m\}\) is a set of domains,
- \(\text{attr} : R \rightarrow \text{TUP}(A)\), where \(\text{TUP}(A)\) denotes a set of finite tuples of different elements of \(A\), is a mapping that assigns to each relation name a tuple of attribute names,
- \(\text{dom} : A \rightarrow D\) is a mapping that assigns to each attribute name a domain.

**Definition 2.**

An *instance of relational database* (or just a *relational database*) for schema \(S = (R, A, D, \text{attr}, \text{dom})\) is a set \(DB = \{R_1, \ldots, R_k\}\) where \(R_i\) is a relation instance (or just a relation) over the relation name \(R_i \in R\), i.e.,

\[
R_i \subseteq \text{dom}(A_1) \times \ldots \times \text{dom}(A_l),
\]

where \(\text{attr}(R_i) = (A_1, \ldots, A_l)\), \(l \leq n\) and \(\times\) is the Cartesian product operator.

**Example 1.**

Let us consider a database storing data about the patients at a hospital. For simplicity, we will use only two attributes and one relation name.
Querying Temporal Database with the Language of First-Order... 

\[ R = \{ \text{PATIENTS} \}, \ A = \{ \text{ID, NAME} \}, \ D = \{ \text{N, CHAR} \} \]

\[ \text{attr(PATIENTS)} = (\text{ID, NAME}), \]

\[ \text{dom}(\text{ID}) = \text{N}, \ \text{dom}(\text{NAME}) = \text{CHAR}. \]

\[ DB = \{ \text{PATIENTS} \} \]

\[ \text{PATIENTS} = \{(1, \text{Kowalski}), (2, \text{Kozłowski}), (3, \text{Piasecka})\} \]

The fact \((x, y) \in \text{PATIENTS}\) means that a person named \(y\) with an identifier \(x\) is a patient at a specific hospital. A database can also be represented (not formally, for the sake of readability) as a set of tables. A table represents a relation. In this example:

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Kowalski</td>
</tr>
<tr>
<td>2</td>
<td>Kozłowski</td>
</tr>
<tr>
<td>3</td>
<td>Piasecka</td>
</tr>
</tbody>
</table>

4. Temporal Database

**Definition 3.**

An *instance of temporal database* (or just a *temporal database*) for schema \(S = (R, A, D, \text{attr, dom})\) over the flow of time \((T, <)\) is a set \(TDB = \{ R_1, \ldots, R_k \}\) where \(R_i\) is a temporal relation instance (or just a temporal relation) over the relation name \(R_i \in R\), i.e.,

\[ R_i(\text{dom}(A_1) \times \ldots \times \text{dom}(A_l)) \times 2^T, \]

where \(\text{attr}(R_i) = (A_1, \ldots, A_l)\).

**Example 2.**

Let \(S\) be the same schema as in Example 1. We take the flow of time to be that of days \(T = \{ \ldots, 2007–03–01, 2007–03–02, 2007–03–03, \ldots \}\).

\[ TDB = \{ \text{PATIENTS} \} \]


\[ (2, \text{Kozłowski}), [2007–02–25, 2007–03–01]), \]


\[ ^2 \text{N denotes a set of natural numbers, CHAR a set of character sequences.} \]

\[ ^3 [a, b] = \{x : x \in T, a \leq x \leq b\}. \]
The set of time points (stamps) associated to a tuple describes when data represented by the tuple are true in modelled reality, i.e., in this example, when a person is (or was or is going to be) a patient at the hospital. The temporal database can also be represented as a set of tables:

**PATIENTS**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Kowalski</td>
<td>[2007–02–25, 2007–03–01]</td>
</tr>
</tbody>
</table>

5. The Query Language for Temporal Database

Let \( S = (R, A, D, \text{attr}, \text{dom}) \) be a relational database scheme, \((T, <)\) be the flow of time and \( TDB = \{ R_1, \ldots, R_k \} \) be a temporal database for \( S \) over \((T, <)\). The query language (QL) for \( TDB \) is based on the language of first-order temporal logic [Gab, ChoTom]. It has the following categories of basic symbols:

- Domain variables: \( x_1, x_2, \ldots \);
- Domain constants: \( c_1, c_2, \ldots \);
- time variables: \( t_1, t_2, \ldots \);
- time constants: \( e_1, e_2, \ldots \);
- elements of \( R \) as predicate symbols: \( R_1, R_2, \ldots, R_k \) and a predicate symbol \( \text{time} \);
- equality symbol: \( = \);
- logical connectives: \( \neg, \land \);
- existential quantifier: \( \exists \);
- temporal connectives: \( U, S \);
- punctuation symbols: \( (, ) \).

**Syntax**

A term is either a constant or a variable. The atomic formulas of the language are of the form:

- \( a_i = a_j \), where \( a_i \) and \( a_j \) are terms of the same sort, i.e., either domain terms or time terms,
- \( R_i(a_1, a_2, \ldots, a_n) \), where \( n \) is the length of the sequence \( \text{attr}(R_i) \) and \( a_j \) is a domain variable (constant) that ranges over (is element of) the domain \( \text{dom}(\text{attr}(R_i))[j] \),
- \( \text{time}(a) \), where \( a \) is a time term.
Formulas of $QL$ are finite strings of basic symbols defined in the following recursive manner:

1. Any atomic formula is a formula,
2. if $\phi, \psi$ are formulas, so also are $\neg \phi, \phi \land \psi, \exists a \phi, U(\phi, \psi), S(\phi, \psi)$, where $a$ is any variable $x_i$.

**Semantics**

We define interpretation $\Theta$ as follows: $\Theta(R_i) = R_i, \Theta(c_i) \in \bigcup D, \Theta(e_i) = T$, for every $i$. An assignment $v$ is a mapping that associates every domain variable $x_i$ with a domain value $v(x_i) \in \bigcup D$ and every time variable $t_i$ with a time point $v(t_i) \in T$. It is convenient to extend an assignment over constants by making $v(c_i) = \Theta(c_i)$ and $v(e_i) = \Theta(e_i)$, for every $i$. We define a formula $\phi$ to be true in $TDB$ at time $t$ under assignment $v$ (denoted by $TBD, v, t \models \phi$) by induction on the structure of the formula:

**Definition 3a.**

1. $TBD, v, t \models R_i(a_1, \ldots, a_s)$ iff $((v(a_1), \ldots, v(a_s)), \tau) \in \Theta(R_i)$, where $\tau \subseteq T$ and $t \in \tau$,
2. $TBD, v, t \models a_i = a_j$ iff $v(a_i) = v(a_j)$,
3. $TBD, v, t \models time(a_i)$ iff $v(a_i) = t$,
4. $TBD, v, t \models \neg \phi$ iff not $TBD, v, t \models \phi$,
5. $TBD, v, t \models \phi \land \psi$ iff $TBD, v, t \models \phi$ and $TBD, v, t \models \psi$,
6. $TBD, v, t \models \exists x_i \phi$ iff $TBD, v^*, t \models \phi$, where $v^*$ is an assignment which agrees with the assignment $v$ on the values of all variables except, possibly, on the values of $x_i$,
7. $TBD, v, t \models U(\phi, \psi)$ iff there exists a $t_1 \in T$ with $t < t_1$ and $TBD, v, t_1 \models \phi$ and for every $t_2 \in T$ such that $t < t_2 < t_1$ holds $TBD, v, t_2 \models \psi$,
8. $TBD, v, t \models S(\phi, \psi)$ iff there exists a $t_1 \in T$ with $t_1 < t$ and $TBD, v, t_1 \models \phi$ and for every $t_2 \in T$ such that $t_1 < t_2 < t$ holds $TBD, v, t_2 \models \psi$.

For convenience, we will introduce additional symbols: $\lor, \to, \leftrightarrow$ (other logical connectives), $\forall$ (universal quantifier) and $F, P, G, H, X, Y$ (other temporal connectives, (see Fig. 1)) defined as:

**Definition 3b.**

1. $\phi \lor \psi \equiv_{\text{def}} \neg(\neg \phi \land \neg \psi)$
2. $\phi \to \psi \equiv_{\text{def}} \neg \phi \lor \psi$
3. $\phi \leftrightarrow \psi \equiv_{\text{def}} (\phi \to \psi) \land (\psi \to \phi)$
4. $\forall x \phi \equiv_{\text{def}} \neg \exists x \neg \phi$
5. $F\phi \equiv_{\text{def}} U(\phi, T)$

89
Definition 4.

A temporal database query is a formula of $QL$ with at least one free variable. The answer of the query $\varphi$ (denoted by $\varphi(TDB)$) is the temporal relation it generates in the database:

$$\varphi(TDB) = \{((v(x_1), \ldots, v(x_s)), \tau) : TDB, v, t \models \varphi \text{ and } t \in \tau\},$$

where $x_1, \ldots, x_s$ are all free variables of the formulae $\varphi$. 

Fig. 1. Graphical Representation of Temporal Connectives
6. Queries

We will formulate four queries over the temporal database presented in Example 2 (we will assume that today is 2007–04–03).

Query 1.

Find those who were (but are no longer) patients at the hospital

\[ \varphi = (P\text{PATIENTS}(x_1, x_2)) \land \neg P\text{PATIENTS}(x_1, x_2) \land \text{time}(2007–04–03) \]

According to definition 4 the answer of the query is the set:

\[ \varphi(TDB) = \{ (((v(x_1), v(x_2)), \tau) : TBD, v, t \models (P\text{PATIENTS}(x_1, x_2)) \land \neg P\text{PATIENTS}(x_1, x_2) \land \text{time}(2007–04–03) \land t \in \tau \} \]

From definition 3a and 3b, we have:

\[ TBD, v, t \models (P\text{PATIENTS}(x_1, x_2)) \land \neg P\text{PATIENTS}(x_1, x_2) \land \text{time}(2007–04–03) \]

\[ \downarrow \text{(def. 3a, p. 5)} \]

(a) \( TBD, v, t \models P\text{PATIENTS}(x_1, x_2) \)
(b) and \( TBD, v, t \models \neg P\text{PATIENTS}(x_1, x_2) \)
(c) and \( TBD, v, t \models \text{time}(2007–04–03) \)

(a) \( TBD, v, t \models P\text{PATIENTS}(x_1, x_2) \)

\[ \downarrow \text{(def. 3b, p. 7)} \]

\[ TBD, v, t \models S(P\text{PATIENTS}(x_1, x_2), \top) \]

\[ \downarrow \text{(def. 3a, p. 8)} \]

there exists \( t_1 \in T \) with \( t_1 < t \)
and \( TBD, v, t_1 \models P\text{PATIENTS}(x_1, x_2) \)
and for every \( t_2 \in T \) such that \( t_1 < t_2 < t \) holds \( TBD, v, t_2 \models \top \)

\[ \downarrow \]

there exists \( t_1 \in T \) with \( t_1 < t \) and \( TBD, v, t_1 \models P\text{PATIENTS}(x_1, x_2) \)

\[ \downarrow \text{(def. 3a, p. 1)} \]

there exists \( t_1 \in T \) with \( t_1 < t \) and \( (v(x_1), v(x_2), \tau) \in P\text{PATIENTS} \),
where \( \tau \subseteq T \) and \( t_1 \in \tau \)

(b) \( TBD, v, t \models \neg P\text{PATIENTS}(x_1, x_2) \)

\[ \downarrow \text{(def. 3a, p. 4)} \]
not $TBD, v, t \models PATIENTS(x_1, x_2)$

\[ \upharpoonright \] (def. 3a, p. 1)

$\langle v(x_1), v(x_2), \tau \rangle \notin PATIENTS$, where $\tau \subseteq T$ and $t \in \tau$

(c) $TBD, v, t \models time(2007–04–03)$

\[ \upharpoonright \] (def. 3a, p. 3)

$t = 2007–04–03$

(a), (b) and (c) are satisfied by:

\[ t_1 = 2007–03–01, \]
\[ \tau = [2007–02–25, 2007–03–01], \]
\[ v(x_1) = 2, v(x_2) = \text{Kozłowski} \]

and

\[ t_1 = 2007–03–16, \]
\[ v(x_1) = 1, v(x_2) = \text{Kowalski}, \]

therefore,


Query 2.

Find those who stayed at the hospital more than once

$\varphi = P(PATIENTS(x_1, x_2) \land P(\neg PATIENTS(x_1, x_2) \land P PATIENTS(x_1, x_2))) \land time(2007–04–03)$


(It can be shown in an analogous way to the previous query).

Query 3.

When did Kowalski (id = 1) stay at the hospital? (in other words: show the past history of the tuple (1, Kowalski))

$\varphi = PATIENTS(1, x)$

Query 4.

Find those who were admitted to hospital between 2007–01–01 and 2007–04–03

\[ \varphi = \text{time}(2006–12–31) \land \neg \text{PATIENTS}(x_1, x_2) \land \]
\[ \text{F(PATIENTS}(x_1, x_2) \land \text{F(time}(2007–04–04))) \]

\[ \varphi(TBD) = \{(1, \text{Kowalski}), \]
\[ ((2, \text{Kozłowski}), [2007–02–25, 2007–03–01]), \]
\[ ((5, \text{Piasecka}), \]

References


