# Towards A Logic For Representation And Retrieval Of Incomplete Temporal Objects

**Ed Porto Bezerra**
Department of Computer Science
Federal University of Paraíba
João Pessoa, Brazil
porto@dsc.ufpb.br

**Ulrich Schiel**
Department of Systems and Computing
Federal University of Paraíba
Campina Grande, Brazil
ulrich@dsc.ufpb.br

**Bernardo Lula Júnior**
Department of Systems and Computing
Federal University of Paraíba
Campina Grande, Brazil
lula@dsc.ufpb.br

## Abstract

In order to obtain an adequate model of the universe of discourse of certain information systems, it is of fundamental importance to allow the description of both incomplete and temporal characteristics of certain objects. In particular, in some cases the two types of characteristics may be treated simultaneously. This article proposes a logic, based on modal temporal logic, for the description and retrieval of incomplete temporal objects. Experiments has been done with a prototype, called MITO (Manipulation of Incomplete Temporal Objects) which has given significant elements for the construction of the theory.

## 1. Introduction

The growing importance of the use of computers in modern life becomes more and more evident. In this process, an adequate modelling of the objects in an information system, is fundamental. In order to give support for the information needs, in some cases the system must be able to keep the history of its objects and, both the history as the attributes of the objects may contain imprecise data. These objects must be created, updated adequately, and queries must return the best answer possible.



**Proceedings of the Workshop on Computer Science and Information Technologies CSIT'99**
**Moscow, Russia, 1999**

Research on temporal databases has gained a certain maturity [4,6,8,26,27]. Incomplete information has also a long history, but continues as subject of several research directions [1,7,17,19,28]. The youngest theme is temporal indeterminacy [7,11,13,25]. As our knowledge, there are no work combining non-temporal and temporal indeterminacy in an unified approach.

Also temporal extensions to database languages have been defined as, for instance, TQUEL [24], SQL/Temporal [18], and TSQL2 [25]. The last language supports also some kinds of temporal incompleteness. Actual database languages are not able to deal with incomplete information but, some suggestions are coming up as, e.g. extensions to SQL [7,13].

We can distinguish three kinds of imperfect information, according to the survey of [21]. *Incomplete information* means the absence of some value, given by a 'null value'. This can mean that the value exists but is unknown or that the value is non-existent. For instance, a null value at the 'has-phone' attribute can signify that the object has an unknown phone number or it has no telephone. *Imprecise information* occurs when we have some information, but it is not precise. Imprecision can be interval valued (age between 20 and 25, salary less than 1000, name is 'Smith' or 'Schmidt') or fuzzy valued (age is 'young'). Finally, *uncertain information* associates an estimation of validity to the values. This estimation may be probabilistic or possibilistic.

According to the classification above, in this paper we treat incomplete information, and interval or set-valued imprecise information. We assume that these imperfections can occur at the attributes of an object and at the temporal information attached to each object.

Classical database systems are based on the so called Closed World Assumption (CWA), where all facts not stored in, or not deductible from the database are false. In contrast, in the

Open World Assumption (OWA) a fact is false only if its negation is deductible from the database. All other not true facts are assumed 'unknown' [14]. For systems dealing with imperfect information the CWA is not suitable, since the frontier between 'true' and 'false' facts is inexact. For this reason, false facts must explicitly be stored in the database, and we call them *negative information.* According to Levesque [15] in order to treat incomplete information effectively it must be possible to define exactly what is known (precisely or not), what is not known, and what is false.

As a first step for the validation of our ideas, a prototype has been implemented in LPA-Prolog for Windows [16], called MITO (Manipulation of Incomplete Temporal Information) [2]. Therefore, the rules governing the prototype are more based on intuition as on a formal system.

In this paper we work out an extended first-order logic which gives a sound foundation for systems dealing with incomplete temporal information.

In order to attend the requirements of the OWA and of temporal databases, the theory developed is a temporal modal logic [9,12] with special operators *K* (known) and *M* (maybe). All facts stored in the system are interpreted as known (K) [15] and possible worlds obtained by disjunctive information are classified as maybe (M) [10]. The specific temporal operators are *after, before, during, begin, end* and *never.*

The following section introduces the example used in this article, first in a database form and then in form of logical clauses. The example is used to introduce the several forms of imprecise information considered. In Section 3 we define the alphabet and axiomatics of our logic, including general axioms specific for query processing. The correct interpretation of imprecise temporal intervals associated to objects, is the matter of Section 4, and queries are discussed in Section 5. Section 6 concludes the paper.

## 2. Incomplete Temporal Objects in MITO

An object is something which makes sense in the context of an application and is distinguishable from other objects [22]. In MITO, objects are associated with an identification and a set of attributes. Since the database is temporal, the identification is a pair (*id, t*), where *id* is a unique code in the systems and *t* is the instant when the object has been created. Therefore the history of an object may be retrieved using the *id.* An update of some attribute generates a new entry in the database with the *id* of the updated object, composed with a new *t.* Each temporal object has a special attribute called *time* which contains a time interval determining the validity of the actual state or the object. We do not consider in this paper complex objects, where attribute values may contain other object Ids or complex values, such as sets, tuples, etc.

The following example lists four instances of a class of employment relating employees to companies with a corresponding salary. Each object has an identification and a list of attributes, including the *time* attribute. In our example we do not take into account the decomposition of the identification.

With the values of the example we define our three kinds of imprecision of the data part of the object and also at the temporal part.

| | |
|---|---|
| employment(O1):<br>employee = *antônio* ***or*** *joão*<br>company = *UFRN* ***or*** *UFPB*<br>salary = *1400*<br>time = *[1994, *]* | employment(O3):<br>employee = *joão*<br>company = *UFPB*<br>salary = *<1500*<br>time = *[1994, now]* |
| employment(O2):<br>employee = ***not****(joão)*<br>company = *UFPE*<br>salary = ***in***(2000,3000)<br>time = *[before(1983), now]* | employment(O4):<br>employee = *******<br>company = *UFBA*<br>salary = *none*<br>time = *[1970, 1990]* |

UNKNOWN VALUE: if the value of an attribute is unknown (null value of [5]), it is represented by an asterisk '*'. The case when an attribute is no existent, it contains the special value 'none'. For instance, employment(O4) states that there is an employee working at UFBA without salary, between 1970 and 1990, but we do not know who it is.

IMPRECISE VALUE: imprecision can be given by disjunctive values (object O1), intervals (salary of object O2) or semi-open domains (salary of O3).

NEGATIVE INFORMATION: negation is necessary for the explicit declaration of false facts. Therefore, employment(O2) states that there is an employee of UFPE, but not 'João'.

In the temporal part the endpoints of the interval may be unknown (as the asterisk of object O1), determined by semi-open time intervals *before(t)* and *after(t)*, or by a closed interval *during(I)*, where *t* is a time instant and *I* is a time interval.

The example database above must be converted in a clausal form in order to satisfy the notation of the predicates of the language to be defined in the next section.

| | |
|---|---|
| employment(O1) ∧<br>employee(O1, *or*(antônio, joao)) ∧<br>company (O1,*or*(UFRN, UFPB) ∧<br>salary (O1, 1400) ∧<br>time(O1, [1994, *]) ∧ | employment(O3) ∧<br>employee (O3, joao) ∧<br>company(O3, UFPB) ∧<br>salary(O3, *lt* 1500) ∧<br>time(O3, [1994, *now*]) ∧ |
| employment(O2) ∧<br>employee (O2, *not*(joao)) ∧<br>company (O2,UFPE) ∧<br>salary (O2, *in*(2000, 3000)) ∧<br>time(O2, [*before*(1983), *now*]) ∧ | employment(O4) ∧<br>employee (O4, *) ∧<br>company (O4, UFBA) ∧<br>salary(O4, none) ∧<br>time(O4, [1970, 1990]) |

Note that the object identifier makes the connection between the several predicates concerning properties of the same object. The name of the unary predicate is called the *name of the object.* In the following table we explain the meaning of the temporal operators used to model imprecise endpoints of

the *time* predicate. We distinct between time intervals I and time instants t which are intervals at the lowest granularity.

| Operator | Meaning |
|---|---|
| Before(t) | the instant occurs sometime before t |
| After(t) | the instant occurs sometime after t |
| During(I) | the instant occurs sometime during I |

## 3. LITO - A Logic for Incomplete Temporal Objects

We introduce now an extended first-order logic <*L, A, R*> for dealing with incomplete temporal objects, where *L* is the description of database language, *A* is the set of axioms which characterises the operators of *L*, and *R* is the set of rules to be used for the derivation of new facts (modus ponens) and processing of non-temporal and temporal queries.

### 3.1 Alphabet

logic symbols:

a) punctuation: ( , ), [ , ]

b) connectives: $\neg$, $\vee$

c) quantifier: $\forall$

d) modal operators: *M, K*

e) temporal operators: *never, before, after, during, begin, end*

f) non-temporal operators: *not, or, lt*

g) variables:

non-temporal: x, $x_1$, $x_2$,…

temporal instant: t, $t_1$, $t_2$,…

temporal interval: I, $I_1$, $I_2$,…

Non-logic symbols:

a) constants:

temporal: *, $\infty$, now

non-temporal: * and all character strings

b) predicative symbols: P, $P_1$, $P_2$,..., Time, < and =

### 3.2 Terms

We distinguish between non-temporal terms, temporal instant terms, and temporal interval terms.

*Non-temporal terms:*

a) non-temporal constants are non-temporal terms;

b) each non-temporal variable is a non-temporal term

c) if Tx, $Tx_1$ e $Tx_2$ are non-temporal terms then *or*($Tx_1$, $Tx_2$), *not*(Tx) and *lt* Tx are non-temporal terms

*Temporal instant terms:*

a) temporal constants are temporal instant terms

b) each temporal instant variable is a temporal instant term

c) if Tt is a temporal instant term, then before(Tt) and after(Tt) are temporal instant terms

d) if TI is a temporal interval term, then during(TI) is a temporal instant term

e) if TI is a temporal interval term, then begin(TI) and end(TI) are temporal instant terms

*Temporal interval terms:*

a) each temporal interval variable is a temporal interval term

b) if Tt1 and Tt2 are temporal instant terms, then [Tt1, Tt2] is a temporal interval term

c) if TI is a temporal instant term, then never(TI) is a temporal interval term

d) if Tt is a temporal instant term, then [$\infty$, Tt] e [Tt, $\infty$] are temporal interval terms.

### 3.3 Formulas

*A well formed formula (wff) is an expression defined recursively as:*

a) if x is a non-temporal constant, Tx, Tx1 and Tx2 are non-temporal terms and TI is a temporal interval term, then Tx1 = Tx2, Tx1 < Tx2, P(x), P(x, Tx) and Time(x, TI) are wffs;

b) if F and G are wffs, then $\neg$F, F $\vee$ G, MF and *K*F are wffs;

c) if F is a wff and v is a variable, then $\forall$v (F) is a wff;

d) if $Tt_1$ are $Tt_2$ are temporal instant terms, then $Tt_1 = Tt_2$ is a wff;

e) if $t_1$ are $t_2$ are temporal instant variables, then $t_1 < t_2$ is a wff;

f) if $TI_1$ and $TI_2$ are temporal interval terms then $TI_1 = TI_2$ is a wff.

### 3.4 Definitions

If F and G are wffs, we define the following new symbols:

1) $\exists x\,(F) \Leftrightarrow def\ \neg\forall x\,(\neg F)$
2) $F \wedge G \Leftrightarrow def\ \neg\,(\neg F \vee \neg G)$
3) $F \rightarrow G \Leftrightarrow def\ \neg F \vee G$
4) $F \leftrightarrow G \Leftrightarrow def\ (F \rightarrow G) \wedge (G \rightarrow F)$
5) $t1 \in [t2, t3] \Leftrightarrow def\ (t2 < t1 \vee t1 = t2) \wedge (t1 < t3 \vee t1 = t3)$
6) $I1 \subseteq I2 \Leftrightarrow def$ (begin(I1) > begin(I2) $\vee$ begin(I1) = begin(I2)) $\wedge$ (end(I1) < end(I2) $\vee$ end(I1) = end(I2))
7) $t_2 = succ\ t_1 \Leftrightarrow_{def} (t_1 < t_2 \wedge \forall t_3\,(t_1 < t_3 \rightarrow (t_2 < t_3) \vee (t_2 = t_3)))$
8) $t_2 = pred\ t_1 \Leftrightarrow_{def} succ\ t_2 = t_1$
9) $le$ Tx $\Leftrightarrow_{def} or$(lt Tx, Tx)
10) $gt$ Tx $\Leftrightarrow_{def} not$(le Tx)
11) $ge$ Tx $\Leftrightarrow_{def} not$(lt Tx)
12) $in(Tx_1, Tx_2) \Leftrightarrow_{def} (ge\ Tx_1 \wedge le\ Tx_2)$
13) $sometime$ Time(x, I) $\Leftrightarrow_{def} \exists t(t \in I \wedge$ Time(x, [t, t]))
14) $TI = [Tt_1, Tt_2] \Leftrightarrow_{def} Tt_1 = begin(TI) \wedge Tt_2 = end(TI)$

### 3.5 Axiomatics

The axioms of the systems are grouped in the following categories: basic, non-temporal, temporal and imprecise intervals.

BASIC AXIOMS:

1) All the axioms of first-order logic
2) $\forall x\ (F \vee G) \rightarrow MF$
3) $\forall x\ (F \vee G) \rightarrow MG$
4) $\forall x\ KF \rightarrow F$
5) $F \rightarrow MF$
6) $I_1 \subseteq I_2 \Leftrightarrow_{def} \forall t\ (t \in I_1 \rightarrow t \in I_2)$

NON-TEMPORAL AXIOMS:

7) $\forall x \forall x_1 \forall x_2\ (P(x, or(x_1, x_2)) \leftrightarrow P(x, x_1) \vee P(x, x_2))$
8) $\forall x \forall x_1 \forall x_2\ (P(x, x_1) \wedge P(x, x_2) \rightarrow x_1 = x_2)$
9) $\forall x \forall x_1 \forall x_2\ (K((P(x, or(x_1, x_2)) \rightarrow MP(x, x_1))$
10) $\forall x \forall x_1\ (KP(x, *) \rightarrow MP(x, x_1) \wedge \exists x_2\ P(x, x_2))$
11) $\forall x \forall x_1\ (KP(x, lt\ x_1) \rightarrow \exists x_2\ (x_2 < x_1 \wedge P(x, x_2)) \wedge \forall x_3\ (x_3 < x_1 \rightarrow MP(x, x_3))$
12) $\forall x \forall x_1\ (KP(x, not(x_1)) \rightarrow \neg P(x, x_1) \wedge KP(x, *))$
13) $\forall x\ (KP(x, not(*)) \rightarrow \neg \exists x_1\ P(x, x_1))$

TEMPORAL AXIOMS:

14) $\forall x \forall t_1 \forall t_2\ (Time(x, [t_1, t_2]) \rightarrow (t_1 < t_2) \vee (t_1 = t_2))$
15) $\forall x \forall I\ (Time(x, I) \leftrightarrow \forall t\ (t \in I \rightarrow Time(x, [t, t])))$
16) $\forall x \forall I\ (Time(x, never(I)) \leftrightarrow \forall t\ (t \in I \rightarrow \neg Time(x, [t, t])))$
17) $\forall x \forall t_1\ (Time(x, [\infty, t_1]) \leftrightarrow \forall t_2 (t_2 < t_1 \vee t_1 = t_2 \rightarrow Time(x, [t_2, t_2])))$
18) $\forall x \forall t_1\ (Time(x, [t_1, \infty]) \leftrightarrow \forall t_2 (t_1 < t_2 \vee t_1 = t_2 \rightarrow Time(x, [t_2, t_2])))$
19) $\forall x \forall t_2\ (KTime(x, [*, t_2]) \rightarrow \exists t_1 (Time(x, [t_1, t_2]) \wedge \neg KTime(x, [t_1, t_2])))$
20) $\forall x \forall t_1\ (KTime(x, [t_1, *]) \rightarrow \exists t_2 (Time(x, [t_1, t_2]) \wedge \neg KTime(x, [t_1, t_2])))$

IMPRECISE INTERVALS:

21) $\forall x \forall I \forall t\ \ (Time(x, [during(I), t_1]) \rightarrow \exists t\ (t \in I \wedge Time(x, [t, t_1]))$
22) $\forall x \forall t_1 \forall I\ (Time(x, [t_1, during(I)]) \rightarrow \exists t\ (t \in I \wedge Time(x, [t_1, t]))$
23) $\forall x\ \forall t \forall t_1\ (Time(x, [before(t), t_1]) \rightarrow \exists t_2\ (t_2 < t \wedge Time(x, [t_2, t_1])))$
24) $\forall x\ \forall t_1 \forall t\ (Time(x, [t_1, before(t)]) \rightarrow \exists t_2\ (t_2 < t \wedge Time(x, [t_1, t_2])))$
25) $\forall x\ \forall t \forall t_1\ (Time(x, [after(t), t_1]) \rightarrow \exists t_2\ (t < t_2 \wedge Time(x, [t_2, t_1])))$
26) $\forall x\ \forall t_1 \forall t\ (Time(x, [t_1, after(t)]) \rightarrow \exists t_2\ (t < t_2 \wedge Time(x, [t_1, t_2])))$

### 3.6 Rules

BASIC:

1) $\vdash F$ and $\vdash F \rightarrow G$ then $\vdash G$ (modus ponens)

QUERIES:

The rules for processing basic queries are given as Horn clauses or formulas reducible to Horn clauses. When a query is formulated one variable of the head predicate should be instantiated and the other is fulfilled by the result of the application of the rule and it will contain the answer. This variable can also be considered as a free variable, e.g. [6].

NON-TEMPORAL BASICS:

2) $P_1(x, x_1) \wedge P_2(x, x_2) \wedge KP_1(x, x_1) \wedge KP_2(x, x_2) \rightarrow P_2(x, x_2)$
3) $P_1(x, x_1) \wedge P_2(x, x_2) \wedge (\exists x_3\ KP_1(x, or(x_1, x_3)) \wedge KP_2(x, x_2) \rightarrow MP_2(x, x_2)$
4) $P_1(x, x_1) \wedge P_2(x, x_2) \wedge (\exists x_3\ KP_1(x, lt\ x_3) \wedge (x_1 < x_3)) \wedge KP_2(x, x_2) \rightarrow MP_2(x, x_2)$
5) $P_1(x, x_1) \wedge P_2(x, x_2) \wedge (\exists x_3\ \exists x_4\ KP_1(x, in(x_3, x_4) \wedge (x_3 < x_1 \vee x_3 = x_1) \wedge (x_1 < x_4 \vee x_1 = x_4)) \wedge KP_2(x, x_2) \rightarrow MP_2(x, x_2)$
6) $P_1(x, x_1) \wedge P_2(x, x_2) \wedge KP_1(x, *) \wedge KP_2(x, x_2) \rightarrow MP_2(x, x_2)$
7) $P_1(x, x_1) \wedge P_2(x, x_2) \wedge KP_1(x, not(x_1) \wedge KP_2(x, x_2) \rightarrow \neg P_2(x, x_2)$

TEMPORAL BASICS:

8) $Time(x, I) \wedge KTime(x, I) \rightarrow Time(x, I)$
9) $P(x, x_1) \wedge Time(x, I_1) \wedge (\exists I_2\ KP(x, x_1) \wedge KTime(x, I_2) \wedge I_1 \subseteq I_2) \rightarrow P(x, x_1)$
10) $P(x, x_1) \wedge Time(x, I_1) \wedge (\exists t\ KP(x, x_1) \wedge KTime(x, never([t, t])) \wedge t \in I_1) \rightarrow \neg P(x, x_1)$
11) $P(x, x_1) \wedge Time(x, I_1) \wedge (\neg \exists I_2\ KP(x, x_1) \wedge KTime(x, I_2) \wedge I_1 \subseteq I_2) \wedge (\neg \exists t\ KP(x, x_1) \wedge KTime(x, never([t, t]) \wedge t \in I_1) \rightarrow MP(x, x_1)$
12) $P(x, x_1) \wedge Time(x, never(I_1) \wedge (\exists I_2\ KP(x, x_1) \wedge KTime(x, never(I_2) \wedge I_1 \subseteq I_2) \rightarrow P(x, x_1)$
13) $P(x, x_1) \wedge Time(x, never(I_1) \wedge (\exists t\ KP(x, x_1) \wedge KTime(x, [t, t]) \wedge t \in I_1) \rightarrow \neg P(x, x_1)$
14) $P(x, x_1) \wedge (Time(x, never(I_1)) \wedge (\neg \exists I_2\ KP(x, x_1) \wedge KTime(x, never(I_2) \wedge I_1 \subseteq I_2) \wedge (\neg \exists t\ KP(x, x_1) \wedge KTime(x, [t, t]) \wedge t \in I_1) \rightarrow MP(x, x_1)$

## 4. Transformation Of Imprecise Temporal Intervals

Temporal objects must be correctly interpreted, even if the time interval is imprecise, in order to obtain correct answers to queries. Based on axioms 21-26 we can transform the imprecise time of an object defined by the predicate Time(s, TI) into a conjunction of precise time clauses Time(x, I) or *sometime* Time(x, I). The intervals *I* created must attend axiom 14. The table below shows the full list of these transformations. They were originally proposed in [20] and reformulated in [2,3]. In the appendix we proof one of the corresponding propositions. It is important to note that the interpretation for Time(x, TI) is weaker than TI, but adequate to response queries.

| *begin*(TI) | *end*(TI) | interpretation for Time(x, TI) |
|---|---|---|
| $t_1$ | $T_2$ | Time(x, [$t_1$, $t_2$]) |
| $t_1$ | before($t_2$) | Time(x, [$t_1$, $t_1$]) ∧ *sometime* Time(x, [$t_1$, pred $t_2$]) |
| $t_1$ | after($t_2$) | Time(x, [$t_1$, succ $t_2$]) ∧ *sometime* Time(x, [succ $t_2$, ∞]) |
| T | during(I) | Time(x, [t, begin(I)]) ∧ *sometime* Time(x, I) |
| T | * | Time(x, [t, t]) ∧ *sometime* Time(x, [t, ∞]) |
| Before($t_1$) | $t_2$ | *sometime* Time(x, [∞, pred $t_1$]) ∧ Time(x, [pred $t_1$, $t_2$]) |
| Before($t_1$) | before($t_2$) | *sometime* Time(x, [∞, pred $t_2$]) |
| Before($t_1$) | after($t_2$) | *sometime* Time(x, [∞, pred $t_1$]) ∧ Time(x, [pred $t_1$, succ $t_2$]) ∧ *sometime* Time(x, [succ $t_2$, ∞]) |
| Before(t) | during(I) | *sometime* Time(x, [∞, pred t]) ∧ Time(x, [t, begin(I)]) ∧ *sometime* Time(x, I) |
| Before(t) | * | *sometime* Time(x, [∞, ∞]) |
| After($t_1$) | $t_2$ | *sometime* Time(x, [succ $t_1$, $t_2$]) ∧ Time(x, [$t_2$, $t_2$]) |
| After($t_1$) | before($t_2$) | *sometime* Time(x, [succ $t_1$, pred $t_2$]) |
| After($t_1$) | after($t_2$) | *Sometime* Time(x, [succ $t_1$, ∞]) |
| After(t) | during(I) | *Sometime* Time(x, [succ t, end(I)]) |
| After($t_1$) | * | *Sometime* Time(x, [succ t, ∞]) |
| During(I) | t | *Sometime* Time(x, I) ∧ Time(x, [end(I), t]) |
| During(I) | Before(t) | *Sometime* Time(x, [begin(I), pred t]) |
| During(I) | After(t) | *Sometime* Time(x, I) ∧ Time(x, [end(I), succ t]) ∧ *Sometime* Time(x, [succ t, ∞]) |
| During($I_1$) | During($I_2$) | *sometime* Time(x, $I_1$) ∧ Time(x, [end($I_1$), begin($I_2$)]) ∧ *sometime* Time(x, $I_2$) |
| During(I) | * | *sometime* Time(x, [begin(I), ∞]) |
| * | T | *sometime* Time(x, [∞, t]) ∧ Time(x, [t, t]) |
| * | Before(t) | *sometime* Time(x, [∞, pred t]) |
| * | After(t) | *sometime* Time(x, [∞, ∞]) |
| * | During(I) | *sometime* Time(x, [∞, end(I)]) |
| * | * | *sometime* Time(x, [∞, ∞]) |

In order to understand the table, suppose the time interval associated to an object is [*during*(01/01/1992, 12/31/1992), *after*(12/31/1994)], i.e. it started on a day in 1992 and ended after 1994. Since, a fact associated with the operator *sometime* will return a truth-value *possible*, a query on the validity of the object will return *possible* for times in 1992 and after 1994. For times between 1.1.1993 and 1.1.1995 (the successor of 12.31.1994) the answer will be *true*. In the appendix we proof this proposition.

## 5. Querying Incomplete Temporal Objects

MITO allows two types of queries: logical queries and retrieval queries. In logical queries the user asks for the validity of some fact and obtains one of the answers *yes*, *no*, *possible* and *unknown*. Retrieval queries return values of the database matching partially or totally the query.

*The general form of a query is*

*object-name »(« <attribute> »:» <value>{[ «,» <attribute> »:» <value> ]} »)» [ «;» <interval>]».»*

where:

a) *object-name* is the name of an object (unary predicate);

b) *attribute* is the name of the attribute object of the query

c) the *value* may be one of the following:

. a constant;

. an unknown, imprecise or negative value;

. a variable 'x' which means that this value should be retrieved;

d) *interval* corresponds to the temporal part of the query. If it is not empty it must be a precise or imprecise time interval.

### 5.1 Logic queries

A logic query returns one of the following truth-values *yes*, *no*, *possible* or *unknown* depending on the matching between the query and the database. The value *unknown* is returned if there is not any matching. The following query illustrates this process. Suppose the query *employment(employee: 'joão', company: UFPB, salary: <1500); [1994,now])* over the database described in section 2. There are two objects *employment(O1)* and *employment(O3)* matching (partially) the query. The result of the processing is shown in the table below.

| object | employee | company | salary | time | intra-object truth value |
|---|---|---|---|---|---|
| employment(O1) | Possible | Possible | Yes | Yes | Possible |
| employment(O3) | Yes | Yes | Yes | Yes | Yes |

The final *inter-object value*, combining *Possible* with *Yes*, becomes *Yes*.

After the truth evaluation of each attribute an intra-object evaluation determines the truth value of each object, and an inter-object evaluation determines the final result of the query. Details of this evaluation are described in [2,23].

### 5.2 Retrieval queries

A retrieval query returns values stored in the database. In some cases to these values a possibility predicate is attached. Suppose we want to know who worked for UFPB, we state the query *employment(employee: x, company: UFPB)*. The

object *employment(O1)* returns *possible(joão) and possible(antônio)*, whereas *employment(O3)* returns *yes(joão)*.

## 6. Conclusion

In this article a modal temporal logic has been introduced to deal with incomplete temporal objects. By this, we have tried to obtain a formal model of the intuition of indeterminacy which normally occurs in some information systems. This logic should be the basis of the reasoning engine of the system called MITO.

We have intentionally excluded imperfect information associated with probabilistic, possibilistic or fuzzy functions. In our opinion, there are many application domains where the information is imprecise without knowledge about values of such functions. These can be project, planning, legal and historical databases in general.

One of the few approaches which considers temporal indeterminacy is the TSQL2 proposal [25]. They define an indeterminate period which coincides with our temporal intervals with imprecise endpoints (with *during, after* or *before*). The necessity to compare periods (or intervals) in order to process joins over temporal intervals (in TSQL2) or to answer queries (in MITO) needs a redefinition of the *Before* predicate. In TSQL2 this is obtained by using an parameter called *ordering plausibility* as lower bound of the probability that a period $\alpha$ is before a period $\beta$. In MITO we use the modal operators *K* and *M*, and the four-valued logic which is adequate for applications considered, where a *probability mass function* is absent.

We have implemented a first version of MITO in LPA-Prolog for Windows [16], which allows storing and querying incomplete temporal objects. A difficult task is updating such a system, and we should consider monotonic and non-monotonic updates [2].

From the formal standpoint, the semantics of the LITO must be defined and the completeness of the logic be proven.

## APPENDIX – Demonstration of a proposition of Section 4

**PROPOSITON:**

$\forall x \forall I \forall t$ (Time(x, [*during*(I), *after*(t)]) $\rightarrow$ *sometime* Time(x, I) $\wedge$ Time(x, [end(I), succ t]) $\wedge$ *sometime* Time(x, [succ t, $\infty$])).

Observation: the clause Time(x, [end(I), *succ* t]) is only applicable when end(I) $\leq$ t.

At the right side of each derivation step, we put the step number and the hypothesis (H), definition (D), axiom (A), rule (R) or lemma (L) number used.

We present firstly three lemmas.

**Lemma 1:** $\forall x \forall I$ Time(x, I) $\rightarrow$ *sometime* Time(x, I)

1. $\neg$ *sometime* Time(x, I) thesis's negation
2. $\neg \exists t(t \in I \wedge$ Time(x, [t, t])) 1 with D13
3. $\neg$ Time(x, I) $\leftrightarrow \neg \exists t(t \in I \wedge$ Time(x, [t, t])) 1 with A15
4. $\neg$ Time(x, I) 2,3 with R1.

**Lemma 2:** $\forall x \forall I_1 \forall I_2$ *sometime* Time(x, $I_1$) $\wedge$ ($I_1 \subseteq I_2$) $\rightarrow$ *sometime* Time(x, $I_2$)

1. $\exists t(t \in I_1 \wedge$ Time(x, [t, t])) $\wedge$ ($I_1 \subseteq I_2$) H with D13
2. $\exists t(t \in I_1 \wedge$ Time(x, [t, t])) $\wedge \forall t_1$ ($t_1 \in I_1 \rightarrow t_1 \in I_2$) 1 with A6
3. $\exists t(t \in I_1 \wedge$ Time(x, [t, t])) $\wedge$ ($t \in I_1 \rightarrow t \in I_2$) 2 with ($t_1$/t)
4. $\exists t(t \in I_2 \wedge$ Time(x, [t, t])) 3 with R1
5. *sometime* Time(x, $I_2$) 4 with D13.

**Lemma 3**: $\forall x \forall I_1 \forall I_2$ (Time(x, $I_2$) $\wedge$ ($I_1 \subseteq I_2$) $\rightarrow$ Time(x, $I_1$))

1. $\forall t \forall x \forall I_1 \forall I_2$ ($t \in I_1 \rightarrow t \in I_2$) $\wedge$ Time(x, $I_2$) H with A6
2. $\forall t \forall x \forall I_1 \forall I_2$ ($t \in I_1 \rightarrow t \in I_2$) $\wedge$ ($t \in I_2 \rightarrow$ Time(x, [t, t])) 1 with A15
3. $\forall t \forall x \forall I_1$ ($t \in I_1 \rightarrow$ Time(x, [t, t])) 2 with A1
4. $\forall x \forall I_1$ (Time(x, $I_1$) 3 with A15.

*Demonstration of the proposition:*

1st part: $\forall x \forall I \forall t$ (Time(x, [*during*(I), *after*(t)]) $\rightarrow$ *sometime* Time(x, I))

1. $\forall x \forall I \forall t$ ($\exists t_1$ ($t_1 \in$ I) $\wedge$ Time(x, [$t_1$, *after*(t)])) H with A21
2. $\forall x \forall I \forall t$ ($\exists t_1$ ($t_1 \in$ I) $\wedge$ Time(x, [$t_1$, $t_1$])) 1 with L3
3. $\forall x \forall I$ *sometime* Time(x, I) 2 with D13.

2nd part: $\forall x \forall I \forall t$ (Time(x, [*during*(I), *after*(t)] $\rightarrow$ Time(x, [end(I), *succ* t])

4. $\forall x \forall I \forall t \exists t_1$ ($t_1 \in$ I) $\wedge \exists t_2$($t < t_2 \wedge$ Time(x, [$t_1$, $t_2$])) 1 with A26
5. $\forall x \forall I \forall t \exists t_1$ ($t_1 \in$ [*begin*(I), *end*(I)] $\wedge \exists t_2$ ($t < t_2 \wedge$ Time(x, [$t_1$, $t_2$])) 4 with D14
6. $\forall x \forall I \forall t \exists t_1$ ($t_1 <$ *end*(I) $\vee$ ($t_1$ = *end*(I))]) $\wedge \exists t_2$ ($t < t_2 \wedge$ Time(x, [$t_1$, $t_2$])) 5 with D5
7. $\forall x \forall I \forall t \exists t_1$ ($t_1 <$ *end*(I) $\vee$ ($t_1$ = *end*(I))]) $\wedge \exists t_2$ ($t < t_2 \wedge t <$ *succ* t $\wedge \forall t_3$($t < t_3 \rightarrow$ *succ* $t < t_3 \vee$ *succ* $t = t_3$) $\wedge$ Time(x, [$t_1$, $t_2$])) 6 with D7
8. $\forall x \forall I \forall t \exists t_1$ ( $t_1 <$ *end*(I) $\vee$ ($t_1$ = *end*(I))]) $\wedge \exists t_2$ ($t < t_2 \wedge$ ($t < t_2 \rightarrow$ *succ* $t < t_2 \vee$ *succ* $t = t_2$) $\wedge$ Time(x, [$t_1$, $t_2$])) 7 with ($t_3$/$t_2$)

9. $\forall x \forall I \forall t \exists\ t_1$ ( $t_1 < end(I) \vee (t_1 = end(I))]) \wedge \exists t_2$ ($succ\ t < t_2 \vee succ\ t = t_2) \wedge$ Time(x, $[t_1, t_2]$)) 8 with R1

10. $\forall x \forall I \forall t \exists\ t_1 \exists\ t_2$ ($[end(I), succ\ t] \subseteq [t_1, t_2] \wedge$ Time(x, $[t_1, t_2]$)) 9 with D6

11. $\forall x \forall I \forall t$ Time(x, $[end(I), succ\ t]$)) 10 with L3.

$3^{th}$ part: $\forall x \forall I \forall t$ (Time(x, $[during(I), after(t)]) \rightarrow$ *sometime* Time(x, $[succ\ t, \infty]$))

12. $\forall x \forall I \forall t$ Time(x, $[succ\ t, succ\ t]$)) 11 with A15

13. $\forall x \forall I \forall t$ *sometime* Time(x, $[succ\ t, succ\ t]$)) 12 with L1

14. $\forall x \forall I \forall t$ *sometime* Time(x, $[succ\ t, succ\ t]) \wedge$ ($[succ\ t, succ\ t] \subseteq [succ\ t, \infty]$) 13 with A6

15. $\forall x \forall I \forall t$ *sometime* Time(x, $[succ\ t, \infty]$) 14 with L2.

## 7. References


1. Abiteboul S, Hull R, Vianu V. «*Foundations of Databases*». Addison-Wesley, 1995, pp 487-507
2. Bezerra E, Schiel U, Ferneda E. «MITO - Manipulation of Incomplete Temporal Objects». In: *Anais do XII Simpósio Brasileiro de Banco de Dados*, Fortaleza – Ce, 1997, pp 364-374
3. Bezerra E, Schiel U, Nóbrega H. «Uma abordagem formal à representação e manipulação de objetos temporais incompletos». In: *Anais da VII Semana de Informática da Bahia*, Salvador – Ba, 1998, pp 1-15
4. Böhlen M, Chomicki J, Snodgrass R, Toman D. «Querying TSQL2 Databases with Temporal Logic». In: Apers P, Bouzeghoub M, Gardarin G. (eds.) *Advances in Database Technology: EDBT'96*. Springer-Verlag, 1996, pp 325-341 (*Lecture Notes in Computer Science* No. 1057)
5. Codd E. «Extending the database relational model to capture more meaning». In: *ACM Transactions on Database Systems*, Vol. 4, No. 4 ,1979, pp 397-434
6. Chomicki J. «Temporal Query Languages: A Survey». In: Gabay D, Ohlbach H. (eds.) *Temporal Logic: ICTL'94*. Springer-Verlag, 1994, pp 506-534 (*Lecture Notes in Computer Science* volume 827)
7. Dey D, Sarkar S. «Extended SQL Support for Uncertain Embley D, Goldstein R. (eds.) *Conceptual Modeling: ER'97*. Springer-Verlag, 1997, pp 102-112 (*Lecture Notes in Computer Science* No. 1331)
8. Etzion O, Jajodia S, Sripada S. «*Temporal Databases: Research and Practice*». Springer-Verlag, 1998
9. Emerson E. «Temporal and Modal Logic». In: van Leeuwen J. (ed) *Handbook of Theoretical Computer Science*. Elsevier/MIT Press, 1990
10. Enderton H. «A Mathematical Introduction to Logic». Academic Press, 1972
11. Gadia S, Nair S, Peon Y. «Incomplete Information in Relational Temporal Databases». In: *Proc. 18th. Conference on Very Large Databases*, Vancouver, 1992
12. Gabbay D, Hodkinson I, Reynolds M. «*Temporal Logic: Mathematical Foundations and Computational Aspects*». Oxford University Press, 1994
13. Griffiths A, Theodoulidis B. «SQL+i: Adding Temporal Indeterminacy to the Database Language SQL». In: *Proc. 14th. British National Conference On Databases*, Edinburgh, 1996
14. Keller A, Wilkins M. «On the Use of an Extended Relational Model to Handle Changing Incomplete Information». In: *IEEE Transactions on Software Engineering*, No. 7, 1985
15. Levesque H. «The Logic of Incomplete Knowledge Bases». In: Mylopoulus K, Brodie M. (eds) *Readings of Artificial Inteligence & Database.* Morgan Kaufmann Publishers, Inc., 1994, pp 328-341
16. LPA-WIN-PROLOG 3.0. Programming Guide by Bria D. Steel, 1996
17. Liu K, Sunderraman R. «Indefinite and Maybe Information in Relational Databases». In: *ACM Transactions on Database Systems*, Vol. 15, No. 1, 1990, pp. 1-39
18. Melton J. «SQL/Temporal». ISO/IEC JTC 1/SC 21/WG 3 DBL-MCI-0012, 1996
19. Nakata M, Fresconi G, Mura T. «Handling Imperfection in Databases: A Modal Logic Approach». In: Hameurlain A, Tjoa M. (eds) *Database and Expert Systems Applications: DEXA'97*. Springer-Verlag, 1997, pp 613-622 (*Lecture Notes in Computer Science* No. 1308)
20. Oresotu B. *Um sistema de representação e recuperação de dados incompletos e informação temporal*. MSc thesis, Federal University of Paraíba, Campina Grande, 1988
21. Parsons S. «Current Approaches to Handling Imperfect Information in Data and Knowledge Bases». In: *IEEE Transactions on Knowledge and Data Engineering*, No. 3, 1996
22. Rambaugh J, Blaha M, Premerlani W, Eddy F, Lorensen W. «*Object-Oriented Modeling and Design*». Englewood Cliffs, Prentice-Hall, 1991
23. Schiel U. «Representação e recuperação de informação temporal e incompleta». In: *Anais do 3O. Simpósio Brasileiro de Inteligência Artificial*, Rio de Janeiro, 1986, pp 271-281
24. Snodgrass R. «The Temporal Query Language TQUEL». In: *ACM Transactions on Database Systems*, volume 12, No.2, 1987, pp 247-298
25. Snodgrass R. (ed). «*The TSQL2 Temporal Query Language*». Kluwer Academic Publishers, 1995
26. Tansel A, Clifford J, Gadia S, Jajodia S, Segev A, Snodgrass R. «*Temporal Databases*». The Benjamin/Cummings Publishing Company, Inc., 1993
27. Tsotras V, Kumar A. «Temporal Database Bibliography *ACM SIGMOD Record*, Vol. 25, No. 1, 1996, pp 41-51
28. Zicari R. «Incomplete Information in Object-Oriented *ACM SIGMOD Record*, Vol. 19, No. 3, 1990, pp 5-1