Aspects of Dealing with Imperfect Data in Temporal Databases

José Pons, Christophe Billiet, Olga Pons, and Guy De Tré

Abstract In reality, some objects or concepts have properties with a time-variant or time-related nature. Modelling these kinds of objects or concepts in a (relational) database schema is possible, but time-variant and time-related attributes have an impact on the consistency of the entire database. Therefore, temporal database models have been proposed to deal with this. Time itself can be at the source of imprecision, vagueness and uncertainty, since existing time measuring devices are inherently imperfect. Accordingly, human beings manage time using temporal indications and temporal notions, which may contain imprecision, vagueness and uncertainty. However, the imperfection in human-used temporal indications is supported by human interpretation, whereas information systems need extraordinary support for this. Several proposals for dealing with such imperfections when modelling temporal aspects exist. Some of these proposals consider the basis of the system to be the conversion of the specificity of temporal notions between used temporal expressions. Other proposals consider the temporal indications in the used temporal expressions to be the source of imperfection. In this chapter, an overview is given, concerning the basic concepts and issues related to the modelling of time as such or in (relational) database models and the imperfections that may arise during or as a result of this modelling. Next to this, a novel and currently researched technique for handling some of these imperfections is presented.

1 Introduction

The concept of time itself is very complex to handle and interpret [45], [68], though it is very natural and omnipresent. As information systems often attempt the modelling of natural objects, concepts or processes, they often require modelling temporal aspects or concepts. Thus, several proposals have been concerned

José Pons, Olga Pons
Department of Computer Science and Artificial Intelligence, Escuela Técnica Superior de Ingeniería Informática, Universidad de Granada, C/Periodista Daniel Saucedo Aranda s/n, E-18071, Granada, Spain. e-mail: jpons,opc@decsai.ugr.es

Christophe Billiet, Guy De Tré
Department of Telecommunications and Information Processing, Ghent University, St.-Pietersnieuwstraat 41, B-9000 Gent, Belgium. e-mail: Christophe.Billiet,Guy.DeTre@telin.ugent.be

with the obtaining of theoretical models that allow the modelling or representation of time [6], [10].

A very specific type of information systems are database systems, which are computer systems designed to manage databases. A database contains data representing real objects or concepts. Each (atomic) part of these data is a result value of a measurement of a property or a description of a property of a real object or concept. In reality, some aspects or properties of objects or concepts are time-variant or time-related. e.g., the moment of a bank transaction is traditionally a moment in time and thus a time-related notion, the function of an employee in a company can change through recorded history and is thus time-variant. A temporal database schema is a database schema that models real objects or concepts with time-related or time-variant properties. However, the modelling of temporal aspects has a direct impact on the consistency of the temporal database, because the temporal nature of these aspects imposes extra integrity constraints. An example. Consider a relation in a relational library database, modelling the presence of books in the library. Every physical book is represented by a unique identifier. Every record in the relation contains such an identifier, a date on which the corresponding book was loaned and a date on which it was subsequently returned (if it was returned). Without further precautions, a library employee could add several records with the same book identifier, different 'loaned'-dates and no 'returned'-dates. This group of records would represent the same physical book being loaned several times on different dates and never returned, which is of course impossible. A temporal database model will typically constrain record insertion and prevent similar modelling inconsistencies.

A lot of research concerns temporal database models and their approaches to the modelling of time. The first efforts were towards the representation of historical information related to objects represented by records in a database [8]. Some proposals tried to extend the Entity Relationship Model [46], without impact on any database standards like SQL [65].

Notably, in 1994, "A Consensus Glossary of Temporal Database Concepts" was published [29]. For this publication, 44 temporal database researchers, among which some of the main researchers in this field, cooperated to reach a consensus on the nature and definitions of some of the main temporal database concepts and terminology. This glossary was subsequently updated in 1998 [41].

An interesting issue in temporal modelling concerns relationships between temporal notions. Notably, Allen [1] studied temporal relationships between time intervals (and as a special case time points). Among others, the querying of temporal databases has greatly profited from these temporal relationships, because they allowed for richer and more complex user-specified temporal query demands, by allowing to express more complex relationships between the temporal notions in the temporal expressions in the query and the temporal indications in the database. e.g., in a relation modelling who was department head of an institution during which periods of time, a query like 'who were the department heads when Thomas worked for the institution' can be evaluated using similar relationships.

Humans handle temporal information using certain temporal notions like time intervals or time points [29], and they often have to deal with imperfections like

imprecision's, vagueness, uncertainties or inconsistencies possibly contained in the descriptions of these temporal notions. Among many others, these possible imperfections in descriptions of temporal notions determine an important issue in temporal modelling. e.g., the description of the temporal notion in a sentence like 'The Belfry of Bruges was finished on a day somewhere between 01/01/1201 A.D. and 31/12/1300 A.D.' contains imperfection because of the uncertainty in the used time-related expression. It is known that the building was finished on a single day, but it is not known precisely which day this was.

To allow information systems to cope with these and similar data imperfections, many approaches adopt fuzzy sets [52] for the representation of temporal information [54], [55], [5], [20]. The temporal relationships studied by Allen were fuzzified by several authors [58], [55], [67]. Garrido et al. [36] present different temporal operators, defined by a combination of regular fuzzy comparisons. Both [36] and [61] deal with uncertainty in temporal expressions concerning time intervals. Other approaches, like [63], use rough sets [59] to represent time intervals.

Next to temporal modelling, some attention has been spent on temporal reasoning [1]. Although temporal reasoning is not discussed in this chapter, it should be noted that, among others, Dubois and Prade et al. [20], [23] have dealt with fuzziness and uncertainty in temporal reasoning.

The aim of this chapter is to present and explain some main concepts regarding time in information systems and to present and discuss some issues and techniques concerned with handling data imperfections related to time. The rest of this chapter is structured as follows. Section 2 presents some basic concepts and terminology about temporal modelling and discusses some of its important aspects and issues, while section 3 presents some important issues concerning the combination of data imperfections and temporal modelling. In section 4, some basic concepts and terminology about temporal databases are presented, followed by an overview of some interesting issues concerning temporal databases and a survey of some commercial temporal database systems. Finally, in section 5, an approach to querying temporal databases containing imperfect temporal information is presented, followed by some conclusions and some suggestions for future work in section 6.

2 Basic Concepts and Issues in Time Modelling

Before considering the introduction of temporal modelling to information systems, it is necessary to define and explain some main concepts concerning temporal modelling and their corresponding terminology, to situate these and to discuss some properties and issues related to these concepts. In this section, several basic concepts and their corresponding terminology will be defined, explained and situated. Most of these concepts are widely used in the community of temporal databases and their definitions have been agreed upon in the context of [29]. For these concepts, in the entire chapter, the contents of [29] are followed (and often cited).

2.1 Basic Concepts and Properties

In information systems, time itself is usually perceived as a linear or cyclic concept [43]. Therefore, a time domain modelling time is usually represented by a set with an imposed partial order. In general, two main types of time models can be discerned: a *linear* model [3] and a *cyclic* model [51]. In the linear model, a total order is imposed on the set and the progress of time is seen as a linear matter, while cyclic models are mainly used in the modelling of recurrent processes. It should be noted that the majority of proposals use linear time models.

Data models used by information systems (and in specific, temporal database systems) may represent an underlying time axis using *chronons* [29], which can informally be described as the smallest distinguishable time units that can be used in the system. However, to explain what chronons are, an explanation of some other temporal concepts is necessary.

Definition 1. Instant [29]
An ***instant*** is a time point on an underlying time axis.

Thus, an instant is basically an instantaneous time point on the time axis underlying a time model. The term is used in the context of the time model too.

Orthogonal to the classification of time models as linear or cyclic, they can be classified as *discrete*, *dense* or *continuous* models[29], [41]. In a discrete model [8], the notion exists that every instant has a unique successor and the set of (modelled) instants is seen as a discrete one. Here, intuitively, the set of instants can be seen as isomorphic to the set of natural numbers $\mathbb{N}$. In a dense model, the notion exists that between any two instants always lies another. Here, intuitively, the set of instants can be seen as isomorphic to the set of rational numbers $\mathbb{Q}$ (when the set of (modelled) instants is a discrete one) or the set of real numbers $\mathbb{R}$ (when the set of (modelled) instants is a continuous one). In a continuous model, the notion also exists that between any two instants always lies another one, but the set of (modelled) instants is always seen as continuous and there are no "gaps" between successive instants.

Some other necessary concepts are:

Definition 2. Time Interval [29]
A ***time interval*** is the time between two instants.

Definition 3. Duration [29]
A ***duration*** is an amount of time with known length, but no specific starting or ending instants.

A time interval as such is bounded by two instants, whereas a duration is not. Also, it should be noted that an instant is in fact a singular case of a time interval.

Definition 4. Temporal Element [29]
A ***temporal element*** is a finite union of time intervals.

Definition 5. Event [29]
An ***event*** is an instantaneous fact, i.e. something occurring at an instant.

Definition 6. 'Temporal' as Modifier [29]
The modifier ***'temporal'*** is used to indicate that the modified concept concerns some aspect of time.

Data models used for time modelling might now represent an underlying time axis using chronons:

Definition 7. Chronon [29]
In a data model, a ***chronon*** is a non-decomposable time interval of some fixed, minimal duration.

A time model contained in a data model may now represent an underlying time axis by a sequence of consecutive chronons. These chronons have identical durations. A data model will typically not specify the exact chronon duration, so it can be fixed later by applications implementing the data model.

The fact that chronons are actually time intervals has a particular effect on the representation of instants and time intervals. In a time model using chronons, an instant is of course represented by a chronon. A time interval may be represented by a set of contiguous chronons, depending on the amount of time the time interval comprises.

Another classification of time models concerns the use of points or intervals to model time. The equivalence between interval-based and point-based time models is demonstrated in [**?**].

Restrictions on time range may exist, as time may be bounded orthogonally in the past and in the future [43].

2.2 Granularities

An important issue in time modelling concerns the concept of *temporal granularities*. A formal definition for this concept is given in [50]:

Definition 8. Granularity [74]
A ***granularity*** is an ordered set of non-overlapping and continuous temporal elements called *granules*.

Definition 9. Granule [74]
A ***granule*** is the basic time unit in a granularity.

A temporal granularity is in fact a partitioning of the time line (time model) used by a system, usually dependent on the application. For example, the age of an adult human being is usually expressed in years: one will use sentences like 'Laura is 21 years old' instead of sentences like 'Laura is 21 years, 3 months and 4 days old'. In this example any duration shorter than a year needs no representation and thus the used granularity allows no specification for durations shorter than a year. The granules are years.

As a granularity G is an ordered set, each granule may be represented by an integer. In this representation, to keep track of the granularity a granule is an element of, the corresponding granularity name is added in subscript:

$$G = \{i_G \mid i \in \mathbb{Z}\} \tag{1}$$

In a system, the granularity with the shortest granules is the *chronons granularity*, which is denoted by '$\perp$'. It is the granularity of which the granules are chronons.

Definition 10. Mapping function [50]
A mapping function f is a function that maps a given granule i_G, $i \in \mathbb{Z}$, in a given granularity G, to a set of corresponding chronons:

$$\begin{aligned} &f : G \to \mathscr{P}(\perp) \\ &\quad i_G \mapsto \{c_\perp \mid (c_\perp \text{ is contained in } i_G) \wedge (c_\perp \in \perp)\} \end{aligned}$$

Note that a mapping function f always maps from a given granularity G to the powerset of the set of chronons $\perp$. Therefore, the output for a mapping function is an element of $\mathscr{P}(\perp)$ and thus a subset of $\perp$.
A mapping function f requires that the following properties hold [50]:

- *G is an ordered set.*
- *G is a set of continuous granules.*
- *The granules in G do not overlap.*

The existence of mapping functions between granularities and the chronons granularity also allows comparing granularities with respect to the length of their granules. In this context, two important concepts can be discerned.

Definition 11. Finner Than [50]
Consider a mapping function f and let i_G and j_H be elements of granularities

G and H respectively. Granularity G is now said to be ***finner than*** granularity H if:

$$|f(i_G)| < |f(j_H)|$$

Definition 12. Coarser Than [50]
Consider a mapping function f and let i_G and j_H be elements of granularities G and H respectively. Granularity G is now said to be ***coarser than*** granularity H if:

$$|f(i_G)| > |f(j_H)|$$

It is also possible to describe the relation between different granularities. This is called a casting function:

Definition 13. Casting function [50]
Consider two different granularities G and H. A granularity-to-granularity ***casting function*** *cast* is then a function mapping granules from G to granules from H:

$$\begin{aligned} cast &: G \times \mathbb{G} \times \mathbb{G} \rightarrow H \\ &: (i_G, G, H) \rightarrow j_H \end{aligned}$$

where $i_G \in G$ and $j_H \in H$ and where $\mathbb{G}$ denotes the set of all granularities.

Thus, the function *cast* associates a granule i_G in G to a corresponding granule j_H in H. Two kinds of granularity-to-granularity mappings can now be discerned: an *upwards mapping* is a mapping from a granularity G to a coarser granularity H, whereas a downwards mapping is a mapping from a granularity K to a finner granularity L. Orthogonal to this classification, mappings between two granularities may be classified as *regular mappings*, *irregular mappings* or *congruent mappings* [50], [30].

- *Regular mapping*: A regular mapping is a granularity-to-granularity mapping where the mapping function value is calculated by means of multiplications and/or divisions and (maybe) an anchor adjustment. e.g., the mapping value of the mapping between hours and minutes is calculated using a multiplication by 60.
- *Irregular mapping*: An irregular mapping is a granularity-to-granularity mapping where the mapping function value can not be calculated by means of multiplications and/or divisions. e.g., the mapping value of the mapping between months and days is dependent on the exact month or day.
- *Congruent mapping*: A congruent mapping is a granularity-to-granularity mapping where the two granularities involved in the mapping have the same granules but a different anchor. e.g., the mapping between the days (Gregorian calendar days) and the academic days is a congruent mapping.

Different granularity-to-granularity mappings between several granularities can be represented using a granularity graph, which is a directed graph indicating the mapping conversions. The above is illustrated in the following example.

Example 1. Consider a system that models both Gregorian calendar dates as well as academic calendar dates. In this system, the chronons granularity is a set of milliseconds. Figure 1 shows the complete granularity graph corresponding to this example. The transition between the chronons granularity and the seconds granularity is an example of a regular mapping. Regular mappings are represented by thin arrows in the visualisation of the graph. The transition between the days granularity and the months granularity is an example of an irregular mapping. In the graph visualisation, irregular mappings are represented by a bold arrow. Finally, the transition between the Gregorian calendar day granularity and the academic day granularity is an example of a congruent mapping. Both concern the same days, but the academic year starts on October 1st, whereas the Gregorian calendar year starts on January 1st. In the graph visualisation, congruent mappings are visualised as straight lines without arrow heads.

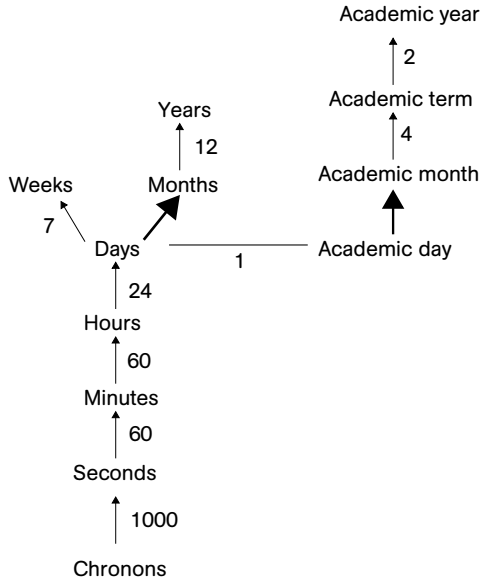

Fig. 1 The granularity graph corresponding to example 1.

2.3 Temporal Relationships

In this section, a brief introduction can be found, concerning *temporal relationships*, sometimes also called ‘temporal relations’[5]. Temporal relationships can be seen as

relationships between temporal elements belonging to the same time domain. These relationships express how the temporal elements are related to one another, with respect to temporal precedence and overlap.

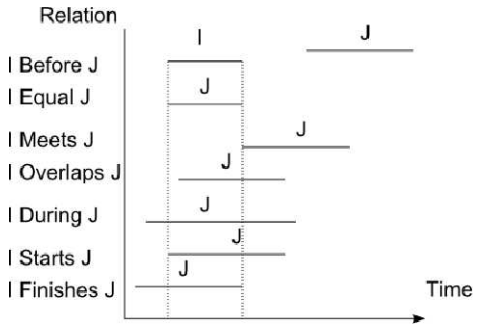

Fig. 2 Allen relations between two time intervals I and J.

Several (collections of) operators have been proposed in order to compare temporal elements and model the temporal relationships between them. Allen [1] most notably described such relationships between time intervals and as a special case, between instants. Figure 2 shows the temporal relationships Allen discerned. Some proposals can be applied to both crisp and other time intervals [58], [55], [67], [36].

3 Data Imperfections in Time Modelling

As explained in the introduction, humans handle temporal information using temporal notions like time intervals or time points [29]. While the used temporal notions may contain imperfections [16], [20], [55], [23], humans often gracefully deal with these, as their inherent interpretation capability accounts for a lot of them. This phenomenon has been studied a.o. in the field of artificial intelligence [13], [28] and language understanding [11], [55], [16]. An information system, however, cannot appeal to a similar interpretation functionality. Thus, many proposals have been concerned with the combination of time and imperfections in the context of information systems [55]. In this section, some main concepts and issues concerning this combination are presented.

3.1 Types of Imperfections in Temporal Modelling

Generally, in temporal modelling, a distinction is made between the following types of imperfections [55].

- *Uncertainty*: Temporal information or data may contain uncertainty. This means that the exact temporal value is (partially) unknown, however, generally some knowledge is present anyhow, possibly describing the value [**?**], [55], [**?**], [54]. E.g., the temporal notion described in a sentence like 'The Belfry of Bruges was finished on a day somewhere between 01/01/1201 A.D. and 31/12/1300 A.D.' contains uncertainty: it is known that the belfry of Bruges was finished on a single day and that this day lays somewhere between 01/01/1201 A.D. and 31/12/1300 A.D., but it is not known exactly which day it is.
- *Vagueness*: Temporal information or data might contain inherent vagueness, as a precise instant or time interval may be intended, but the description of it is certainly vague [67], [55], [16]. E.g., the temporal notion described in a sentence like 'It happened during summer.' contains vagueness, as even the boundaries of the mentioned temporal notion are not clearly expressed.
- *Subjectivity or ambiguity*: Temporal notions might be subject to subjectivity or ambiguity. In certain cases, the temporal notion concerns a historical period like 'late romanticism' or 'the early middle ages' and thus contains subjectivity [55]. In other cases, the interpretation of the temporal notion depends on extra factors. E.g., consider a person saying to another person 'Let's meet each other at six.' The person hearing these words doesn't now if 6 a.m. or 6 p.m. is intended, though the person saying the words does.

As to the sources of the imperfections in temporal information, most proposals consider no specific source [28], [67], [55], [23], [54], [58], [73]. Some proposals, however, deal with the imperfections specifically resulting from aspects of language [16] and other proposals consider transitions between different granularities to be the source of imperfection in temporal information [50]. Therefore, some proposals consider granularity as the base of the temporal model [10].

In an information system, temporal information is usually related to facts or events [7]. In light of this, a classification of temporal information can be considered, in which the following types of temporal information may be found:

- *Definite temporal information*: Definite temporal information contains information describing a situation in which all time indications associated with some fact are absolute time indications. The temporal information is precisely known.
- *Indefinite temporal information* [18]: Indefinite temporal information contains information describing a situation in which the time indication associated with some fact has not been fully defined. E.g., consider an event that in fact occurred but it is not known exactly when.
- *Infinite temporal uncertain information* [44]: Infinite temporal uncertain information contains information describing a situation in which an infinite number of time indications are associated with some fact. This is usually found in recurrent events like meetings. E.g., consider meetings that take place every Wednesday at noon. Some systems (usually with different information providers) may dispute the occurrence and/or the duration of a fact.

3.2 Representation of Imperfect Information

As mentioned before, information systems may have to deal with time indications which contain vagueness. Even for some specific events or facts, the temporal indications may become imprecise. Therefore, a time point might be specified by means of a time interval of which the boundaries may not be precisely known. An example.

Example 2. Consider a speaker and a hearer. The speaker wants to make an appointment with the hearer. Now, consider the speaker saying:

'We will meet each other tomorrow around 10'

The hearer will now usually instinctively agree that the appointment will be in e.g. the time interval between 9.55h and 10.05h.

The study of the semantics of 'around' in temporal [16] indications has shown that the size of the time interval associated with the imprecise specification of a time point depends on the distance with respect to the current time. E.g., consider now that the speaker is talking about something that happened *'during last week'*, then the hearer would consider a time interval of more or less 10 days.

Some proposals [47], [10], [55], [7] conclude that the best representation for incomplete temporal knowledge is therefore based on time intervals, even if they refer to a fact that happen at a time point. This means that, as Allen proposed in [1], the primitive units (the chronons) in a time domain, used in an information system should be intervals.

In order to represent and manage uncertain temporal information properly, several theoretical frameworks have been proposed:

- *Probability theory*: Probability theory [18], [49], [39] is usually employed when uncertainty concerning a time interval allows a probability to be associated to the time interval. The use of probability theory is very usual in logistics information systems. E.g., *'The package will arrive at its destination on Monday morning with a probability of 0.8'*.
- *Possibility theory*: Using possibility theory [**?**], a possibility degree is associated to the temporal fact or event. Possibility theory is widely used to model uncertainty and vagueness in time [20],[23],[17],[55]. Several works [67], [58] present fuzzy versions of the temporal relations proposed by Allen [1]. The aim of these works is generally to obtain a flexible way to compare uncertain, ill-known temporal intervals by means of temporal relationships. The study of imperfect temporal metadata is done in [**?**], [**?**]. In [**?**] a proposal to use in fuzzy databases temporal fuzzy linguistic terms is studied. Burney [**?**], [**?**] has studied recently the combination of fuzzy databases with temporal data.
- *Rough sets*: Rough set theory [59] has been used to represent uncertainty in time intervals. The two dimensional representation of time intervals and the temporal relationships between them has been studied in [63]. In [**?**] a rough set-based model for temporal databases is presented. The study of temporal relationships between rough time interval is studied in [**?**].

3.3 Imperfections in Temporal Relationships

As the existence of temporal relationships allows to compare temporal notions, many approaches have been concerned with finding similar temporal relationships, able to support imperfections in the temporal information which is described by temporal notions or even by the temporal relationships themselves [58], [55], [67], [20]. These approaches are often based on Allen's operators [1]. Example 3 presents a short example concerning one of Allen's relationships.

Example 3. Consider an event which takes place between time points A and B. Thus, the event comprises time interval $[A,B]$ (this is visualized in part (1) of figure 3). The classical Allen relationship 'after' returns an interval $]B,\infty[$ as shown in part (2) of figure 3. A fuzzified version of Allen's 'after' operator is illustrated in part (3) of figure 3. The comparison between two time intervals results in a possibility degree in the unit interval. The shape of the possibility distribution is shown in part (3) of figure 3. Note that all the points strictly after the point B results in a possibility degree of 1 whereas there is an area near the point B in which the possibility degree runs smoothly between 0 and 1.

Consider now the interval given by $[C,D]$, illustrated in part (4) of figure 3. The user wants to know if $[C,D]$ is after $[A,B]$. The crisp version of the 'after' operator would return *'no'* as an answer. The fuzzy version for the same operator would return *'yes, with a possibility of 0.5'*.

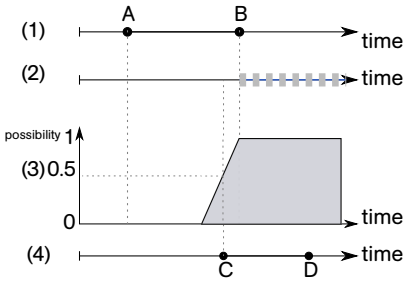

Fig. 3 Example for the Allen relationship *'after'*. (1) The event bounded within time points A and B. (2) The crisp version of the *'after'* operator. (3) A fuzzy version of the after operator. (4) Another event, bounded within time points $[C,D]$.

4 Basic Concepts and Issues in Temporal Databases

A temporal database can generally be seen as a database that manages some temporal aspects in its schema [32], [5]. In subsection 4.1, some main concepts and properties concerning temporal databases and their definitions are presented and explained.

In subsections 4.2 and 4.3, some main issues of relational temporal databases are presented and discussed. Finally, subsection 4.4 presents an overview of some commercial temporal database systems.

4.1 Basic Concepts and Properties

A database schema models some part of reality. As mentioned in the introduction, the part of reality a temporal database schema tries to model, contains some temporal aspects. For example, in this part of reality, some concepts or objects could have time-related or time-variant properties. The modelling of these temporal aspects has to be handled specifically in order for the database to maintain a consistent model of reality.

Thus, a temporal database will contain *temporal values*, i.e. values representing (indications of) time. Temporal values in a temporal database can be classified into the following types based on their interpretation and modelling purpose. The definitions and explanations of these types can be found in [29] and [56] and more information can be found in [42], [69] and [56].

Definition 14. Valid Time [29]
The ***valid-time*** (VT) of a fact is the time when the fact is true in the modeled reality.

Definition 15. Transaction Time [29]
A database fact is stored in a database at some point in time, and after it is stored, it is current until logically deleted. The ***transaction-time*** (TT) of a database fact is the time when the fact is current in the database and may be retrieved.

Definition 16. Decision Time [56]
Decision time (DT) denotes the time when an event was decided to happen.

Definition 17. User-defined Time [29]
User-defined time (UDT) is an uninterpreted attribute domain of date and time.

Valid times are usually provided by the user, whereas *transaction-times* are usually system-generated and -supplied [29]. Temporal values of type UDT are not given any extraordinary interpretation and have thus no extraordinary query language support [29].

A *temporal database* can now formally be defined as follows:

Definition 18. Temporal Database [29]
A ***temporal database*** supports some aspect of time, not counting user-defined time.

In a relational temporal database, temporal values will of course be in the tuples of the extensions of temporal relations:

Definition 19. Valid-time Relation [29]
A ***valid-time relation*** is a relation with exactly one system supported valid-time.

Definition 20. Transaction-time Relation [29]
A ***transaction-time relation*** is a relation with exactly one system supported transaction-time.

A *valid-time*, respectively *transaction-time relational database* is now defined as containing one or more valid-time, respectively transaction-time relations [29]. Next to this, *bitemporal* relational databases contain both valid-time and transaction-time [29] and tritemporal databases contain valid-time, transaction-time and decision-time [56].

A very extensive list of the most well-known temporal database models can be found in [75]. As it is of course necessary to define a consistent way to query the temporal data, there are several proposals concerned with query languages and query language adaptations for temporal databases like [57] and [70].

In the rest of the chapter, the focus will be on concepts and issues concerning valid-time relations and aspects of valid-time relations. For this reason, the next two sections will present and discuss some main issues concerning temporal databases, specifically applied to or presented in the context of valid-time relations.

4.2 Primary Keys in Valid-time Relation Design

Generally, when designing a relation based on a relational database model, a subset of the relation's attribute set is usually chosen as primary key. The values of a tuple for these attributes will then uniquely determine this tuple, hence no two distinct tuples may have the exact same values for every attribute in this primary key. Next to attributes unrelated to time, a valid-time relation schema will typically contain one or more attributes which model the valid-time aspects and behavior of the real objects and concepts modelled by the relation schema. In this work, these attributes are called *valid-time attributes*. In valid-time relation extensions, distinct tuples can exist containing the exact same values for every attribute except the valid-time attributes. These distinct tuples represent distinct versions of the same real object or

concept, valid during different time periods. To allow the existence of such tuples when designing a valid-time relation using a relational database model, the most common solution is to include the valid-time attributes in the primary key.

The following example illustrates this primary key issue.

Example 4. Consider the example valid-time relation visualized in table 1, which models when certain people worked as employees in a certain company and under whose supervision they worked during that time. The valid-time attributes 'Start' and 'End' describe the year when an employee started, respectively finished working for the company. For example, the last tuple visualized in table 1 represents that the employee represented by this tuple started working for the company in 2005 and finished in 2009. The attributes 'Name', 'Birthday' and 'Supervisor' describe respectively the name, birthday date and unique identifier of the supervisor of an employee in the time during which he or she worked for the company. When correct, the date of an employee's birthday never changes and as such, the modelling of birthday dates has no effect on the database consistency. The 'Birthday' attribute thus describes UDT values. The attribute 'ID' describes employee identifiers. For each tuple, this identifier (a number) uniquely identifies the employee represented by the tuple.

Now consider {ID} being the primary key and consider the company wanting to hire Sarah again in 2010. This would be represented by another tuple in the relation, containing value 4 for attribute 'ID'. The existence of such a tuple is of course not allowed by the primary key, because it would mean the existence of two distinct tuples containing value 4 for attribute 'ID'. This problem can now be solved by defining a new primary key: {ID, Start, End}, which allows for the existence of distinct tuples with value 4 for attribute 'ID', as long as they have different values for attributes 'Start' or 'End'. The resulting relation is shown in table 2.

Table 1 Example relation modelling the employees of a company. Values for the 'Birthday' attribute are visualized here in 'dd/mm/yyyy' format.

ID	Name	Birthday	Supervisor	Start	End
1	Peter	24/10/1985	3	2010	-
2	Maria	03/04/1984	3	2001	-
3	John	21/02/1964	-	1999	-
4	Sarah	29/11/1985	2	2005	2009

4.3 Consistency in Valid-time Relation Content Modification

The solution presented in subsection 4.2 concerns relation design and consists of including the valid-time attributes in the primary key. Unfortunately, implementing this solution as such allows for the existence of records whose values imply inconsistencies with respect to the modelling of reality.

Table 2 Example relation after including the valid-time attributes in the primary key and adding a tuple.

ID	Name	Birthday	Supervisor	Start	End
1	Peter	24/10/1985	3	2010	-
2	Maria	03/04/1984	3	2001	-
3	John	21/02/1964	-	1999	-
4	Sarah	29/11/1985	2	2005	2009
4	Sarah	29/11/1985	2	2010	-

Consider a valid-time relation of which the primary key can be partitioned into two sets of attributes. One set contains attributes totally unrelated to time, for which the values of a record allow to uniquely identify the object or concept represented by the record. The other set contains the valid-time attribute(s). Because the valid-time attribute(s) is(are) included in the primary key, the existence of distinct records with exactly the same values for all time-unrelated attributes and distinct values for at least one valid-time attribute is not prohibited. Thus, inserting such records into the relation is not prohibited either, even if the information represented by the values for the valid-time attributes shows clear inconsistencies. An example.

Example 5. Consider the example valid-time relation visualized in table 3, which is based on the relation visualized in table 1. The primary key is again {ID, Start, End}. The last record in the relation represents a person named 'Sarah' started working for the company in 2007 and finished in 2008, with supervisor 'John'. However, the fourth record represents the same person (the value for attribute 'ID' is the same) started working for the company in 2005 and finished in 2009, with supervisor 'Maria'. The intention is clear: Sarah worked in the company from 2005 to 2009, first for Maria, then for John, then again for Maria. It is of course possible for an employee to change supervisors, but it is of course impossible for a person to start working in the same company twice at different times, for different supervisors, without stopping to work for one in between, as it is impossible to stop working for a supervisor twice at different times, without working for another one in between. The valid-time information represented by the last record is clearly not consistent with the valid-time information represented by the fourth record, or vice versa.

Table 3 Example relation with records whose values for the valid-time attributes violate consistency.

ID	Name	Birthday	Supervisor	Start	End
1	Peter	24/10/1985	3	2010	-
2	Maria	03/04/1984	3	2001	-
3	John	21/02/1964	-	1999	-
4	Sarah	29/11/1985	2	2005	2009
4	Sarah	29/11/1985	3	2007	2008

The most usual approach to deal with this inconsistency problem is to adapt the DML used by the DBMS, as to enforce consistency towards time with respect to the modelled reality.

Example 6. Consider the problem presented in example 5. The inconsistency arises when the last record in table 3 is inserted. Because the record's values for the valid-time attributes differ from those of the fourth record, the last record is accepted. The DML statement used was (the table is called 'Employees'):

```
INSERT INTO Employees VALUES
 (4, 'Sarah', '29/11/1985', 3, 2007, 2008);
```

The inconsistency problem can now be solved by replacing this statement with:

```
UPDATE Employees SET 'End' = '2007' WHERE
 (ID = 4) AND (Start = 2005) AND (End = 2009);
INSERT INTO Employees VALUES
 (4, 'Sarah', '29/11/1985', 3, 2007, 2008);
INSERT INTO Employees VALUES
 (4, 'Sarah', '29/11/1985', 2, 2008, 2009);
```

The resulting relation is visualized in table 4.

Table 4 Example relation updated maintaining consistency.

ID	Name	Birthday	Supervisor	Start	End
1	Peter	24/10/1985	3	2010	-
2	Maria	03/04/1984	-	2001	-
3	John	21/02/1964	-	1999	2010
3	John	21/02/1964	-	2010	-
4	Sarah	29/11/1985	2	2005	2007
4	Sarah	29/11/1985	3	2007	2008
4	Sarah	29/11/1985	2	2008	2009

4.4 Commercial Temporal Database Systems

Several commercial temporal DBMS exist. Table 5 gives an overview of some of the more well-known temporal DBMS and provides references for more information.

Oracle workspace manager [9] and TimeDB [72] are libraries for dealing with time in OracleDB. On another note, TimeDB and Postgree Temporal [62] are similar: both are simple implementations that implement a subset of the Allen operators and some operations for the creation and manipulation of temporal attributes (valid-time, transaction-time or both times are supported). Teradata [71] is mainly a business intelligence system designed for data mining. Secondo [19] is an extensible

database system in which the core of the database may be replaced by a customized algebra. It is designed for non-standard applications and it supports both valid and transaction-times.

The most complete implementation is Workspace Manager.

Unfortunately, none of these systems take data imperfections into account, neither in data storage nor in querying.

Table 5 Commercial Temporal Database Systems.

Name	Time Supported	Comments	Reference
Oracle Workspace Manager	VT and TT.	Package for Oracle DB.	[9]
TimeDB	VT and TT.	Interface for Oracle DB.	[72]
Postgree Temporal	VT.	Package for Postgree SQL.	[62]
Teradata	VT and TT.	Used for data-mining.	[71]
Secondo	VT and TT.	Spatio-temporal database.	[38]

5 Data Imperfections in Temporal Databases

Consider a logistics company which transports packages. At the moment a package leaves, the time when it will arrive at its destination may be estimated, but will typically not be known precisely. For such companies and in many other situations, information systems able to handle imperfection with respect to certain temporal aspects of the objects modelled by the system are necessary.

5.1 Data Imperfections in Temporal Databases

Data and information imperfections and techniques to represent them correctly in databases and queries are usually the focus of research in fuzzy databases. Proposals from this field may present an approach based somehow on fuzzy set theory [52] or possibility theory [22], although other theories support information imperfections too. Comparably, many proposals concerning the introduction of data imperfections or information imperfections in temporal databases present approaches based somehow on fuzzy set theory [36], [35], [55], [5] or possibility theory [23], [61], [12], although proposals based on other theories exist [31], [15], [63]. As possibility theory is usually seen as a theory of confidence, aimed at dealing with uncertainty, in some proposals, possibility theory is used specifically to handle uncertainty in temporal information. In fuzzy databases [34], uncertainty is usually expected to appear in the database content, whereas other types of imperfection, notably imprecision, are usually expected to appear in querying.

Concerning temporal databases, there are several approaches to handle uncertainty in temporal data stored in a database. Many of these approaches concern sev-

eral different types of time notions (VT, TT or DT), but most of these approaches focus somehow on valid-time [36], [35].

In the following subsection, a novel approach to representing uncertainty concerning valid-time notions and a corresponding technique to query similar valid-time indications in a valid-time relation are proposed. The presented proposal is based on concepts introduced in [21] and on the framework proposed in [61].

5.2 Handling Uncertainty in a Valid-time Relation

A valid-time indication usually takes the form of a time interval. Such a valid-time interval can be described (and stored in a valid-time relation record) using its boundaries (endpoints) or using one endpoint and the interval length. Usually, a valid-time interval is represented using its endpoints, which is also the approach adopted by the presented proposal.

Generally, the uncertainty concerning a set of values might be described by a possibility distribution on the powerset of which one of the elements can be the intended set [21]. This representation, however, introduces some issues in practice or in practical applications. Therefore, in the presented proposal, possibility theory is used to model uncertainty, but only uncertainty concerning the exact values of the start and end point of a valid-time interval is considered and the uncertainty in both the start point and the end point are modelled using possibility distributions.

In fact, to model the uncertainty related to a valid-time interval using possibility theory, the presented proposal introduces so-called ill-known time intervals, relying on the concept of ill-known sets [21].

5.2.1 Ill-known Time Intervals

To represent valid-time indications which might contain uncertainty, the presented proposal introduces the concept of ill-known valid-time intervals, which relies on the concept of ill-known sets [21]. To correctly explain the concept of ill-known sets, the concept of possibilistic variables should be introduced first. In the presented proposal, the definition of possibilistic variables of [61] is followed. In [61], a *possibilistic variable* is defined as follows.

Definition 21. Possibilistic variable [61]
A possibilistic variable X over a universe U is defined as a variable taking exactly one value in U, but for which this value is (partially) unknown. Its possibility distribution π_X on U models the available knowledge about the value that X takes: for each $u \in U$, $\pi_X(u)$ represents the possibility that X takes the value u. In the presented work, this possibility is interpreted as a measure

of how plausible it is that X takes the value u, given (partial) knowledge about the value X takes.

Now, consider a set R, which contains single values (and not collections of values). When a possibilistic variable X_v is defined on such a set R, the unique value X_v takes, which is (partially) unknown, is called an *ill-known value* in this work [21].

When a possibilistic variable is defined on the powerset $\mathscr{P}(R)$ of some universe R, the unique value the variable takes will be a crisp set and its possibility distribution on the powerset $\mathscr{P}(R)$ will describe the possibility of each crisp subset of R to be the value the variable takes. This value (a crisp set) the variable takes, which is (partially) unknown, is now called an *ill-known set* [21].

Finally, consider a set R, which contains single values (and not collections of values) and its powerset $\mathscr{P}(R)$. Now consider a subset $\mathscr{P}_I(R)$ of $\mathscr{P}(R)$ and let this subset contain every element of $\mathscr{P}(R)$ that is an interval, but no other elements. When a possibilistic variable X_i is defined on the subset $\mathscr{P}_I(R)$ of the powerset $\mathscr{P}(R)$ of some set R, the unique value X_i takes will be a crisp interval and the possibility distribution π_{X_i} of X_i will be a possibility distribution on $\mathscr{P}_I(R)$. This π_{X_i} will define the possibility of each value of $\mathscr{P}_I(R)$ (a value of $\mathscr{P}_I(R)$ is a crisp interval which is a subset of R) being the value X_i takes. This exact value (a crisp interval) the variable takes, which is (partially) unknown, is called an *ill-known interval* here.

The presented proposal will deal with ill-known time intervals. Ill-known time intervals are ill-known intervals in a time domain. In the presented proposal, an ill-known time interval will be defined and represented via its start and end point, which will be ill-known values. The elements of the ill-known time interval are the values between its start and end point, including the start and end points themselves[1]. It should be clear that this approach to representing ill-known time intervals differs from the approach based on a single possibility distribution on a set $\mathscr{P}_I(R)$ of a set R. These approaches have a different behavior and can be used to describe different ill-known time intervals. The correspondences, interactions and transitions between these different representations of ill-known intervals and their interpretations are part of the authors current research.

In the presented proposal, a closed ill-known time interval with start point defined by possibilistic variable X and end point by possibilistic variable Y is noted $[X,Y]$. Figure 4 shows a closed ill-known time interval.

Several authors work with concepts very similar to these ill-known time intervals and some of them [36] propose transformations of the describing functions in order to optimize the storage of such ill-known valid-time intervals, though recent research might seem to indicate some minor issues with respect to some of these transformations [61]. A comparison between the transformations in [36] and the framework in [61] is presented in [60]. Figure 5 illustrates a transformation based on the 'convex hull' approach [36].

[1] The presented proposal only deals with closed ill-known time intervals. Dealing with halfopen or open ill-known intervals is part of the current research of the authors.

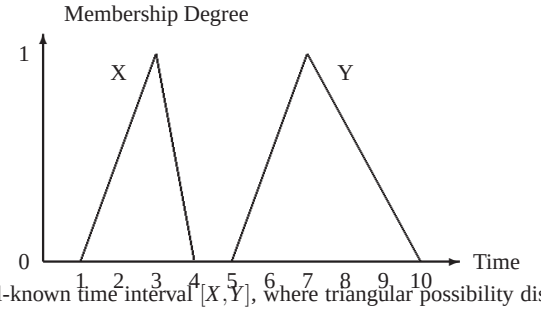

Fig. 4 A closed ill-known time interval $[X,Y]$, where triangular possibility distributions describe the ill-known values defining the start and end points.

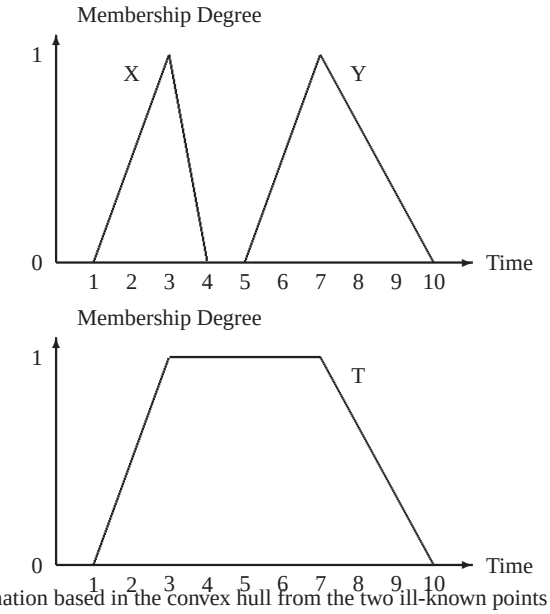

Fig. 5 Transformation based in the convex hull from the two ill-known points X and Y.

In the presented proposal, ill-known valid-time intervals will be used to represent valid-time indications in a valid-time relation and to model the uncertainty these may contain.

To evaluate the temporal demands in queries issued by users to query a valid-time relation containing ill-known valid-time intervals, the presented proposal introduces a technique based on the concept of ill-known time constraints, which is based on the concept of ill-known constraints as presented in [61]. Both concepts are treated in subsection 5.2.2.

Before ill-known time constraints can be introduced, another notion related to possibilistic variables shoud be paid attention to. In fact, a specific application of possibilistic variables is obtained when the set under consideration is the set of boolean values, denoted $\mathbb{B} = \{T,F\}$, where T denotes 'true' and F denotes 'false' [61]. Indeed, any boolean proposition p takes exactly one value in $\mathbb{B}$. If the

knowledge about which value this proposition p takes, is given by a possibility distribution π_p, then proposition p can be seen as a possibilistic variable. In the presented proposal, the interest lies with the case where the proposition holds (denoted $p = T$) and the possibility and necessity that $p = T$ demand most attention. In the following sections, the following notations are used, based on previous notations:

$$\text{Possibility that } p = T: \qquad Pos(p) = \pi_p(T) \tag{2}$$

$$\text{Necessity that } p = T: \qquad Nec(p) = 1 - \pi_p(F) \tag{3}$$

5.2.2 Ill-known Time Constraints

The presented proposal contains a technique for evaluating user queries used to query a valid-time relation in which the valid-time indications are represented by ill-known time intervals. Part of this query evaluation technique relies on the concept of ill-known time constraints, which is based on the concept of ill-known constraints as presented in [61]. These concepts are presented below. Following [61], an *ill-known constraint* is defined as follows.

Definition 22. Given a universe U, an *ill-known constraint* C is specified by means of a binary relation $R \subseteq U^2$ and a fixed, ill-known value defined by its possibilistic variable V on U, i.e.:

$$C \triangleq (V, R)$$

A set $A \subseteq U$ now satisfies this constraint C if and only if:

$$\forall a \in A : (V, a) \in R$$

An example of an ill-known constraint is given by:

$$C_< \triangleq (X, <)$$

Some set A then satisfies $C_<$ if

$$\forall a \in A : X < a$$

The satisfaction of a constraint $C \triangleq (V, R)$ by a set A is basically a Boolean matter (either the set satisfies the constraint or not) and can thus be seen as a boolean proposition, but due to the uncertainty inherent to the ill-known value V, it can be uncertain whether C is satisfied by A or not [61]. Based on the possibility distribution π_V of V, the possibility and necessity that A satisfies C can be found. This proposition can thus be seen as a possibilistic variable on $\mathbb{B}$. The required possibility and necessity are calculated using the following formulas [61].

$$Pos(A \text{ satisfies } C) = \min_{a \in A} \left(\sup_{(w,a) \in R} \pi_V(w)) \right) \tag{4}$$

$$Nec(A \text{ satisfies } C) = \min_{a \in A} \left(\inf_{(w,a) \notin R} 1 - \pi_V(w) \right) \tag{5}$$

Now, to calculate the possibility or necessity of a set A satisfying multiple constraints, the min t-norm operator is used to express a conjunction of constraints. For example:

$$Pos((A \text{ satisfies } C_1) \text{ and } (A \text{ satisfies } C_2)) =$$
$$\min_{a \in A}(Pos(A \text{ satisfies } C_1), Pos(A \text{ satisfies } C_2))$$
$$Nec((A \text{ satisfies } C_1) \text{ and } (A \text{ satisfies } C_2)) =$$
$$\min_{a \in A}(Nec(A \text{ satisfies } C_1), Nec(A \text{ satisfies } C_2))$$

Accordingly, the max t-conorm operator is used to express a disjunction of constraints. For example:

$$Pos((A \text{ satisfies } C_1) \text{ or } (A \text{ satisfies } C_2)) =$$
$$\max_{a \in A}(Pos(A \text{ satisfies } C_1), Pos(A \text{ satisfies } C_2))$$
$$Nec((A \text{ satisfies } C_1) \text{ or } (A \text{ satisfies } C_2)) =$$
$$\max_{a \in A}(Nec(A \text{ satisfies } C_1), Nec(A \text{ satisfies } C_2))$$

In the presented proposal, *ill-known time constraints* are considered, which are ill-known constraints of which the considered universe is a time domain.

In the next subsection, the core of the presented proposal is described.

5.2.3 Querying Valid-time Relations containing Ill-known Valid-time Intervals

One of the main purposes of the existence of (relational) databases is to allow information retrieval. The standard query language for (relational) databases is SQL [53], but several proposals to extend the SQL language for transaction-time databases [66], valid-time databases [33] and bitemporal databases [57] exist and some authors have studied how to support temporal querying in standard SQL [70].

As mentioned before, the presented proposal deals with querying a valid-time relation. In this subsection, the core of the presented proposal is described. First the particular structure of the relation is described, along with the nature and structure

of its supposed contents. Next, the particular query structure is presented. Finally, the particular method for evaluating queries and for ranking the result records are presented.

Relation Structure

In the presented proposal, a valid-time relation is considered, in which every record contains just one valid-time indication. This valid-time indication is represented by a closed ill-known time interval, to allow uncertainty in the valid-time information. As explained above, the ill-known time intervals used here will be defined and represented via their start and end points, which are ill-known values in the valid-time domain.

Query Structure

In the presented proposal, a query consists of two separate constructs of user demands.

Definition 23. Query
A query $\tilde{Q}$ is given by:

$$\tilde{Q} = \left(Q^{time}, Q\right) \tag{6}$$

Here, Q denotes the construct of all (possibly fuzzy) non-temporal user query demands. These comprise all constraints and demands unrelated to valid-time and thus unrelated to the valid-time indications in the queried relation. Q^{time} denotes the temporal demand specified by the user.

The presented query structure allows the user to specify a single temporal demand, denoted by Q^{time}.

Definition 24. Temporal demand
A temporal demand Q^{time} is defined by:

$$Q^{time} = (I, AR) \tag{7}$$

Here, I denotes a crisp time interval (which can be specified in any way required) and AR denotes one of the Allen relations (cf. section 2.3).

The interpretation of such a temporal demand $Q^{time} = (I, AR)$ is that, for a record with an ill-known valid-time interval J, the user demands that I AR J holds.

Query evaluation

Query satisfaction in a fuzzy relational database is usually a matter of degree. Typically, the evaluation of the query demands for a record results in a *satisfaction degree s*, which is typically in the unit interval, i.e. $s \in [0,1]$. This satisfaction degree then models the extent to which the record satisfies the query demands. As such, a satisfaction degree of 0 indicates total dissatisfaction while a degree of 1 indicates total satisfaction and every level of satisfaction between total satisfaction and total dissatisfaction is indicated by a satisfaction degre between 0 and 1.

In the presented approach, for every record r, each part of a query $\tilde{Q} = \left(Q^{time}, Q\right)$ is evaluated independently:

- The user preferences expressed in the non-temporal part Q, are evaluated, resulting in a satisfaction degree denoted $e_Q(r)$. The presented approach accepts any valid, sound method of calculating this evaluation, as long as the method is well-founded and $e_Q(r) \in [0,1]$.
- The evaluation of the temporal demand expressed in the temporal part, $Q^{time} = (I, AR)$, depends on AR. A specific construct of ill-known constraints (cf. section 5.2.2) is considered, depending on the Allen relation denoted by AR. The exact construct of constraints is an instantiation based on the formulas which can be found in table 6, for every possible value of AR. The form and capacity of these constraints are based on [61]. Then, using equations (4) and (5), the exact formulas to calculate the possibility $\text{Pos}_{Q^{time}}(r)$ and the necessity $\text{Nec}_{Q^{time}}(r)$ that record r satisfies this construct of ill-known time constraints are derived from this construct of constraints. As mentioned, $\text{Pos}_{Q^{time}}(r)$ and $\text{Nec}_{Q^{time}}(r)$ denote the possibility, respectively the necessity that the considered record r satisfies the construct of constraints corresponding to the temporal demand Q^{time} and thus the possibility, respectively the necessity that r satisfies Q^{time}.

Table 6 Constructs of constraints related to their respective Allen relations, as used in the presented work. In this table, the ill-known time interval $J = [X,Y]$ in a record r has a start point described by possibilistic variable X and an end point described by possibilistic variable Y. The crisp time interval in the user's temporal demand is denoted I.

Allen Relation	Construct of constraints
I before J	$C_1 \triangleq (<, X)$
I equal J	$\left(C_1 \triangleq (\geq, X)\right) \wedge \neg\left(C_2 \triangleq (\neq, X)\right) \wedge \left(C_3 \triangleq (\leq, Y)\right) \wedge \neg\left(C_4 \triangleq (\neq, Y)\right)$
I meets J	$\left(C_1 \triangleq (\leq, X)\right) \wedge \neg\left(C_2 \triangleq (\neq, X)\right)$
I overlaps J	$\left(C_1 \triangleq (<, Y)\right) \wedge \neg\left(C_2 \triangleq (\leq, X)\right) \wedge \neg\left(C_3 \triangleq (\geq, X)\right)$
I during J	$\left(\left(C_1 \triangleq (>, X)\right) \wedge \left(C_2 \triangleq (\leq, Y)\right)\right) \vee \left(\left(C_3 \triangleq (\geq, X)\right) \wedge \left(C_4 \triangleq (<, Y)\right)\right)$
I starts J	$\left(C_1 \triangleq (\geq, X)\right) \wedge \neg\left(C_2 \triangleq (\neq, X)\right)$
I finishes J	$\left(C_1 \triangleq (\leq, Y)\right) \wedge \neg\left(C_2 \triangleq (\neq, Y)\right)$

Aggregation and Ranking

In this subsection, the notations used in the previous subsection are followed. To be able to present the most appropriate results to the user most prominently, for every record r, an aggregation method is used to aggregate $\text{Pos}_{Q^{time}}(r)$ and $\text{Nec}_{Q^{time}}(r)$ into a temporal record rank $e_{Q^{time}}(r)$ and after this, a convex combination combining $e_{Q^{time}}(r)$ and $e_Q(r)$ will provide the final record rank $e_{final}(r)$.

To calculate $e_{Q^{time}}(r)$, an a simple and crude method is used:

$$e_{Q^{time}}(r) = \left(\frac{\text{Pos}_{Q^{time}}(r) + \text{Nec}_{Q^{time}}(r)}{2} \right) \tag{8}$$

This method aims to provide the result records with a natural ranking based on the users temporal constraint. $e_{Q^{time}}(r)$ will of course be a value in $[0,1]$, as both $\text{Pos}_{Q^{time}}(r) \in [0,1]$ and $\text{Nec}_{Q^{time}}(r) \in [0,1]$. The purpose is that records which fit the users temporal demand better get a higher score than records fitting the temporal demand worse. Here, this aim is reached because the necessity degree $\text{Nec}_{Q^{time}}(r)$ cannot exceed 0 unless the possibility degree $\text{Pos}_{Q^{time}}(r)$ equals 1.

The final ranking $e_{final}(r)$ for a record r is now given by a convex combination of both temporal and non-temporal evaluation scores.

$$e_{final}(r) = \omega * e_Q(r) + (1-\omega) * e_{Q^{time}} \tag{9}$$

A convex combination is used mainly for 2 reasons:

- The use of this convex combination allows a record to make up for a low temporal evaluation score with a high non-temporal evaluation score and vice versa.
- The exact value of ω can now be modified to ascribe more value to either the fulfillment of the user's temporal demands or the fulfillment of the user's non-temporal constraints.

In the next subsection, some main concepts and issues concerning bipolarity in the context of temporal databases are presented and discussed.

5.3 Bipolarity in Temporal Databases

Humans express their preferences using both positive and negative statements, where positive statements express what is desired or acceptable and negative statements express what is undesired or unacceptable [5]. This realization is interesting with regard to database querying, because sometimes a user does not exactly know his or her preferences or can't express them in only positive statements, but prefers to use negative statements to express what he or she dislikes or doesn't need. This introduces the need for *bipolar querying*, a technique to model both positive and negative user preferences in a database query. Sometimes positive and negative preferences are clearly symmetric, making it possible to derive one from the other. For

example, a person may define the concept of 'tall' as '1.80 meters or higher'. The negative would then be the opposite: not tall would be 'anything less than 1.80 meters'. However, in some cases, positive preferences can not be directly obtained from negative preferences or vice versa. E.g., when a person prefers to buy a black motorbike, this does not necessarily mean the person would totally reject a very dark blue motorbike. This phenomenon is called *heterogeneous bipolarity* [26], [27].

The use of imprecise query preference formulation in bipolar querying is well discussed in existing literature [14], [27], [48]. In [48], desired and mandatory query conditions are used, instead of positive and negative preferences. However, the inverse of a mandatory preference expresses what should be rejected and this could be seen as negative information, whereas desired query conditions can be seen as positive preferences. However, the combination of bipolar querying and the use of imprecise query preferences in the context of temporal databases is not so well discussed in existing literature. A proposal for the bipolar querying of valid-time databases has been made by Billiet et al. [5]. The model presented there deals with a fuzzy valid-time specification based on [36].

Bipolarity can be handled using different concepts, such as intuitionistic fuzzy sets [2], interval valued fuzzy sets [76] Grattan-Guiness [37], Janh [40], Sambuc [64], [25] or two fold fuzzy sets [24].

From a theoretical point of view, bipolarity might be found either in the queries presented to a database system or in a database managed by a database system.

When bipolarity is found in queries, it is possible to distinguish between:

- Bipolarity inside query criteria: each individual query criterion may be specified using both positive and negative preferences. For example when querying a car database, the user can express that he or she wants a black car, but definitely not a red neither a blue one. Bipolarity resides here within the car color criterion.
- Bipolarity outside query criteria: the query is specified using a global positive and a global negative preference part. For example when querying a car database, the user can express that he or she wants a black car, but definitely not a car with a fuel consumption of 6 liters or more.

Concerning bipolarity inside a database, it should be possible to specify both positive and negative real world object or concept aspects, even at record level. Nevertheless, not so much research exists concerning bipolarity in databases.

6 Conclusions and Further Research

In this chapter, some of the main concepts concerning information imperfections in temporal modelling and information imperfections in temporal modelling in information systems and the terminology corresponding with these concepts are introduced and explained and some of the main properties of and issues with these concepts are presented and discussed. An overview of some commercial temporal

DBMS is briefly introduced. Finally, a novel technique for querying valid-time relations using imperfect query specifications is presented.

Further research work could follow several general directions. First of all, a theoretical model for dealing with uncertainty in both the database and the query at the same time could be researched and defined. Next, implementations including both DDL and DML could be proposed and constructed.

Acknowledgements

Part of this research is supported by the grant BES-2009-013805 within the research project TIN2008-02066: *Fuzzy Temporal Information treatment in relational DBMS*.

References

1. Allen, J.F.: Maintaining knowledge about temporal intervals. Commun. ACM **26**, 832–843 (1983)
2. Atanassov, K.T.: Intuitionistic fuzzy sets. Fuzzy Sets and Systems **20**, 87–96 (1986)
3. Benthem, J.F.K.A.V.: The Logic of Time: A Model-Theoretic Investigation into the Varieties of Temporal Ontology and Temporal Discourse. Reidel, Hingham, MA (1982)
4. Bhlen, M.H., Busatto, R., Jensen, C.S.: Point- versus interval-based temporal data models
5. Billiet, C., Pons, J.E., Matthé, T., De Tré, G., Pons Capote, O.: Bipolar fuzzy querying of temporal databases. In: Lecture Notes in Artificial Intelligence, vol. 7022, pp. 60–71. Springer, Ghent, Belgium (2011)
6. Bolour, A., Anderson, T.L., Dekeyser, L.J., Wong, H.K.T.: The role of time in information processing: a survey. ACM SIGMOD Record **12**, 27–50 (1982)
7. Chountas, P., Petrounias, I.: Modelling and representation of uncertain temporal information. Requirements Engineering **5**, 144–156 (2000)
8. Clifford, J., Tansel, A.U.: On an algebra for historical relational databases: two views. SIGMOD Rec. **14**, 247–265 (1985)
9. Corp., O.: Oracle database 11g. workspace manager overview.
10. Van der Cruyssen, B., De Caluwe R.and De Tré, G.: A theoretical fuzzy time model based on granularities. EUFIT'97 pp. 1127–1131 (1997)
11. De Caluwe, R., Van der Cruyssen, B., De Tré, G., Devos, F., Maesfranckx, P.: Fuzzy time indications in natural languages interfaces, pp. 163–185. Kluwer Academic Publishers, Norwell, MA, USA (1997)
12. De Caluwe, R., De Tré, G., Van Der Cruyssen, B., Devos, F., Maesfranckx, P.: Time management in fuzzy and uncertain object-oriented databases, *Knowledge management in fuzzy databases*, vol. 39, pp. 67–88. Physica-Verlag (2000)
13. De Tré, G., De Caluwe, R., Van der Cruyssen, B.: Dealing with time in fuzzy and uncertain object-oriented database models. EUFIT'97 pp. 1157–1161 (1997)
14. De Tré, G., Zadrozny, e.a.: Dealing with Positive and Negative Query Criteria in Fuzzy Database Querying Bipolar Satisfaction Degrees. In: Proceedings of 8th Int. Conf. FQAS, pp. 593–604. Springer Verlag Berlin, Denmark (2009)
15. Dekhtyar, A., Ross, R., Subrahmanian, V.S.: Probabilistic temporal databases, i: algebra. ACM Trans. Database Syst. **26**, 41–95 (2001)

16. Devos, F., Maesfranckx, P., De Tré, G.: Granularity in the interpretation of around in approximative lexical time indications. Journal of Quantitative Linguistics **5**, 167–173 (1998)
17. Devos, F., Van Gyseghem, N., Vandenberghe, R., De Caluwe, R.: Modelling vague lexical time expressions by means of fuzzy set theory. Journal of Quantitative Linguistics **1**(3), 189–194 (1994)
18. Dey, D., Sarkar, S.: A probabilistic relational model and algebra. ACM Trans. Database Syst. **21**(3), 339–369 (1996)
19. Dieker, S., Güting, R.H.: Plug and play with query algebras: Secondo-a generic dbms development environment. In: Proceedings of the 2000 International Symposium on Database Engineering & Applications, IDEAS '00, pp. 380–392. IEEE Computer Society, Washington, DC, USA (2000)
20. Dubois, D., HadjAli, A., Prade, H.: Fuzziness and uncertainty in temporal reasoning. Journal of Universal Computer Science **9**(9), 1168–1194 (2003)
21. Dubois, D., Prade, H.: Incomplete conjunctive information. Computers & Mathematics with Applications **15**, 797–810 (1988)
22. Dubois, D., Prade, H.: Possibility Theory: An Approach to Computerized Processing of Uncertainty. Plenum Press, New York (1988)
23. DuBois, D., Prade, H.: Processing fuzzy temporal knowledge. IEEE Transactions on Systems, Man, and Cybernetics **19**, 729–744 (1989)
24. Dubois, D., Prade, H.: Bipolarity in flexible querying. In: Proceedings of the 5th International Conference on Flexible Query Answering Systems, FQAS '02, pp. 174–182. Springer-Verlag, London, UK, UK (2002)
25. Dubois, D., Prade, H.: Interval-valued fuzzy sets, possibility theory and imprecise probability. In: In Proceedings of International Conference in Fuzzy Logic and Technology, pp. 314–319 (2005)
26. Dubois, D., Prade, H.: Rough Sets and Current Trends in Computing, *Lecture Notes in Computer Science*, vol. 4259, chap. Bipolar Representations in Reasoning, Knowledge Extraction and Decision Processes, pp. 15–26. Springer, Heidelberg, Germany (2006)
27. Dubois, D., Prade, H.: Handbook of Research on Fuzzy Information Processing in Databases, chap. Handling bipolar queries in Fuzzy Information Processing, pp. 97–114. Information Science Reference, New York, USA (2008)
28. Dutta, S.: An event based fuzzy temporal logic. In: Multiple-Valued Logic, 1988., Proceedings of the Eighteenth International Symposium on, pp. 64 –71 (1988)
29. Dyreson, C., Grandi, F.e.a.: A consensus glossary of temporal database concepts. SIGMOD Rec. **23**, 52–64 (1994)
30. Dyreson, C., Snodgrass, R.: Temporal granularity and indeterminacy: Two sides of the same coin. Technical report tr 94-06, Computer Science Department, University of Arizona, Tucson, U.S.A (1994)
31. Dyreson, C.E., Snodgrass, R.T.: Supporting valid-time indeterminacy. ACM Trans. Database Syst. **23**, 1–57 (1998)
32. Etzion, O., Jajodia, S., Sripada, S.: Temporal databases: research and practice. Lecture notes in computer science. Springer (1998)
33. Gadia., S.K.: A seamless generic extension of sql for querying temporal data (1992)
34. Galindo, J.: Fuzzy Databases: Modeling, Design, and Implementation. IGI Publishing, Hershey, PA, USA (2006)
35. Galindo, J., Medina, J.M.: Ftsql2: Fuzzy time in relational databases. In: EUSFLAT Conf. '01, pp. 47–50 (2001)
36. Garrido, C., Marin, N., Pons, O.: Fuzzy intervals to represent fuzzy valid time in a temporal relational database. Int. J. Uncertainty Fuzziness Knowlege-Based Syst. **17**, 173–192 (2009)
37. Grattan-Guinness, I.: Fuzzy membership mapped onto intervals and many-valued quantities. Mathematical Logic Quarterly **22**(1), 149–160 (1976)
38. Güting, R.H., Schneider, M.: Moving objects databases. URL `http://dna.fernuni-hagen.de/Lehre-offen/Kurse/1675/KE1.pdf`
39. Haddawy, P.: Believing change and changing belief. IEEE Transactions on Systems, Man, and Cybernetics Special Issue on Higher-Order Uncertainty **26** (1996)

40. Jahn, K.U.: Intervall-wertige mengen. Mathematische Nachrichten **68**(1), 115–132 (1975)
41. Jensen, C., Dyreson, C., Böhlen, M., Clifford, J., Elmasri, R., Gadia, S., Grandi, F., et al.: The consensus glossary of temporal database concepts - february 1998 version. In: Lecture Notes in Computer Science, pp. 367 – 405 (1998)
42. Jensen, C.S., Mark, L., Roussopoulos, N.: Incremental implementation model for relational databases with transaction time. IEEE Trans. Knowl. Data Eng. **3**, 461–473 (1991)
43. Jensen, C.S., Snodgrass, R.T., Soo, M.D.: The tsql2 data model. In: The TSQL2 Temporal Query Language, pp. 153–238 (1995)
44. Kabanza, F., m. Stevenne, J., Wolper, P.: Handling infinite temporal data. In: Journal of Computer and System Sciences, pp. 392–403 (1990)
45. Klein, W.: Time in Language. Routledge, London, U.K. (1994)
46. Klopprogge, M.R., Lockemann, P.C.: Modelling information preserving databases: Consequences of the concept of time. In: Proceedings of the 9th International Conference on Very Large Data Bases, pp. 399–416. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1983)
47. Knight, B., Ma, J.: Time representation: A taxonomy of temporal models. Artificial Intelligence Review **7**, 401–419 (1993)
48. Lacroix, M., Lavency, P.: Preferences; putting more knowledge into queries. In: Proceedings of the 13th International Conference on Very Large Data Bases, VLDB '87, pp. 217–225. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1987)
49. Lakshmanan, L.V.S., Leone, N., Ross, R., Subrahmanian, V.S.: Probview: A flexible probabilistic database system. ACM TRANSACTIONS ON DATABASE SYSTEMS **22**(3), 419–469 (1997)
50. Lin, H., (codirector, C.S.J., Ohlen, M.H.B., Busatto, R., Gregersen, H., Torp, K., (codirector, R.T.S., Datta, A., Ram, S.: Efficient conversion between temporal granularities (1997)
51. Lorentzos, N.A.: A formal extension of the relational model for the representation of generic intervals. Ph.D. thesis, Birkbeck College, University of London (1988)
52. Lotfi Zadeh: Fuzzy sets. Information and Control **8**, 338–353 (1965)
53. Melton, J., Simon, A.R.: Understanding the new SQL: a complete guide. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1993)
54. Mitra, D., et al.: A possibilistic interval constraint problem: Fuzzy temporal reasoning. In: Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proc. of the Third IEEE Conference on, vol. 2, pp. 1434–1439 (1994)
55. Nagypál, G., Motik, B.: A fuzzy model for representing uncertain, subjective, and vague temporal knowledge in ontologies. In: On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE, *LNCS*, vol. 2888, pp. 906–923. Springer, Heidelberg (2003)
56. Nascimento, M.A., Eich, M.H.: Decision time in temporal databases. In: Proceedings of the Second International Workshop on Temporal Representation and Reasoning, pp. 157–162 (1995)
57. Navathe, S., Ahmed, R.: Tsql—a language interface for history databases (1987)
58. Ohlbach, H.J.: Relations between fuzzy time intervals. International Symposium on Temporal Representation and Reasoning **0**, 44–51 (2004)
59. Pawlak, Z., Grymala-Busse, J., Slowinski, R., Ziarko, W.: Rough Sets. Communications of the ACM **38**(6) (1995)
60. Pons, J., Bronselaer, A., Pons, O., de Tre, G.: Possibilistic evaluation of fuzzy temporal intervals. In: G. Sainz, J. Alcalá (eds.) Actas del XVI Congreso Español sobre Tecnologías y Lógica Fuzzy. Valladolid, Spain (2012)
61. Pons, J.E., Bronselaer, A., De Tré, G., Pons, O.: Possibilistic evaluation of sets (2011). Submitted to the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems
62. Postgree: Temporal postgreesql (2011). URL `http://pgfoundry.org/projects/temporal/`
63. Qiang, Y., Asmussen, K., Delafontaine, M., De Tré, G., Stichelbaut, B., De Maeyer, P., Van de Weghe, N.: Visualising rough time intervals in a two-dimensional space. In: 2009 IFSA World Congress / EUSFLAT Conference, Proceedings (2001)

64. Sambuc, R.: Fonctions ϕ-floues. application l'aide au diagnostic en pathologie thyroidienne. Ph.D. thesis, Univ. Marseille, France (1975)
65. Sarda, N.L.: Extensions to sql for historical databases. IEEE Trans. Knowl. Data Eng. **2**, 220–230 (1990)
66. Sarda, N.L.: Extensions to sql for historical databases. IEEE Trans. Knowl. Data Eng. **2**, 220–230 (1990)
67. Schockaert, S., De Cock, M., Kerre, E.: Fuzzifying allen's temporal interval relations. Fuzzy Systems, IEEE Transactions on **16**(2), 517 –533 (2008)
68. Shackle, G.: Decision, order and time in human affairs. Cambridge University Press (1961)
69. Snodgrass, R.: The temporal query language tquel. In: Proceedings of the 3rd ACM SIGACT-SIGMOD symposium on Principles of database systems, PODS '84, pp. 204–213. ACM, New York, NY, USA (1984)
70. Snodgrass, R.T., Richard, C., Snodgrass, T., Snodgrass, T., Jensen, C.S., Jensen, C.S., Jensen, C.S., Steiner, A., Steiner, A., Bhlen, M.H., Bhlen, M.H., Bhlen, M.H., Bhlen, M.H., Busatto, R., Gregersen, H., (codirector, C.S.J., Torp, K., Datta, A., (codirector, R.T.S., Dyreson, C.E.: Transitioning temporal support in tsql2 to sql3. In: O. Etzion, S. Jajodia, S. Sripada (eds.) Temporal Databases: Research and Practice, *Lecture Notes in Computer Science*, vol. 1399. Springer Berlin / Heidelberg (1998)
71. Teradata: Teradata temporal (2011). URL `http://www.teradata.com/database/teradata-temporal/`
72. TimeDB: A temporal relational dbms. (2011). URL `http://www.timeconsult.com/Software/Software.html`
73. Virant, J., Zimic, N.: Attention to time in fuzzy logic. Fuzzy Sets and Systems **82**(1), 39 – 49 (1996)
74. Wang, X.S., Jajodia, S., Subrahmanian, V.S.: Temporal modules: An approach toward federated temporal databases. In: INFORMATION SYSTEMS, pp. 227–236 (1993)
75. Wu, Y., Jajodia, S., Wang, X.: Temporal database bibliography update. In: O. Etzion, S. Jajodia, S. Sripada (eds.) Temporal Databases: Research and Practice, *Lecture Notes in Computer Science*, vol. 1399, pp. 338–366. Springer Berlin / Heidelberg (1998)
76. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning - i. Inf. Sci. **8**(3), 199–249 (1975)