Acta Informatica manuscript No.
(will be inserted by the editor)

Mining Approximate Interval-based Temporal Dependencies

Carlo Combi · Pietro Sala

Received: date / Accepted: date

Abstract Temporal functional dependencies (TFDs) add valid time to classical functional dependencies (FDs) in order to express data integrity constraints over the flow of time. If the temporal dimension adopted is an interval, we have to deal with interval-based temporal functional dependencies (ITFDs for short), which consider different interval relations between tuple valid times. The related approximate problem is when we want to check wheter our data satisfy, without any constraint for the schema, a given ITFD under a given error threshold $0 \leqslant \epsilon \leqslant 1$. This can be rephrased as: given a relation instance r, is it possible to delete at most $\epsilon \cdot |r|$ tuples from it in such a way that the resulting instance satisfies the given ITFD? This optimization problem, *ITFD-Approx* for short, may represent a way to discover (i.e., mine) important dependencies among attribute values in a database. In this paper we analyze the complexity of problem *ITFD-Approx* restricting ourselves to Allen's interval relations: we shall see how the complexity of such a problem may significantly change, depending on the considered interval relation.

Keywords Temporal Database · Functional Dependencies · Interval Relations

1 Introduction

Temporal databases allow the description of the temporal evolution of information by associating one or more temporal dimensions with stored data [20]. The fundamental temporal dimension, we shall consider in this paper, associated with any stored fact is *valid time*, which describes the time when the fact is true in the

Carlo Combi
Department of Computer Science,
University of Verona
E-mail: carlo.combi@univr.it

Pietro Sala
Department of Computer Science,
University of Verona
E-mail: pietro.sala@univr.it

modeled reality. Several different kinds of (temporal) constraints may be expressed on temporal data: temporal constraints are usually expressed through languages based on first-order logic [35]. Among temporal integrity constraints for temporal data, a special kind of constraints, namely *temporal functional dependencies*, has been introduced [34]. Temporal functional dependencies (TFDs) add a temporal dimension to classical functional dependencies (FDs), to deal with temporal data. As an example, while FDs model constraints like "*professors with the same role get the same salary*", TFDs can represent constraints like "*for any given month, professors with the same role have the same salary, but their salary may change from one month to the next one*" or "*current salaries of professors uniquely depend on their current and previous roles*" [11].

Most TFDs proposed in literature rely on some point-based semantics, possibly extended to consider some fixed point-based temporal grouping when different temporal granularities, i.e., time partitions, are considered [3,11,34]. Recently, in [13,14] we focused on interval-based temporal constraints expressed through interval-based temporal functional dependencies (ITFDs), considering the well known Allen's interval relations to constrain data valid times. ITFDs allow us to express constraints as "*Salaries of professors hired the same day with the same role are the same (but professors hired with the same role in different days may have different salaries)*", which cannot be expressed through point-based TFDs. We already showed how to efficiently manage the incremental verification of different ITFDs, by proposing suitable data structures based on B-trees for interval-based indexing of data [13,14].

On the other hand, functional dependencies may be viewed, instead of constraints, as a way of representing some (possibly) unknown features of collected data. In this case, we do not impose any constraint on temporal data, as we are interested in mining some specific (temporal) features highlighted in most (but not all) data. Some thresholds are usually given on the number of data items that do not satisfy the considered feature. The concept of approximate functional dependency (AFD) is defined upon the concept of plain FD: given some data where an FD holds for *most* of data, we may identify the remaining data items, for which that FD does *not* hold. Consequently, we can define different measures to quantify the error we make in considering the FD to hold on the given data set [18,19].

Recently, approximate functional dependencies have been extended to consider different kinds of point-based temporal functional dependencies, which can be expressed according to the framework proposed in [11], and have been applied in some clinical domains, to allow physicians to mine new knowledge from data [9, 12]. At the best of our knowledge, the issue of dealing with approximate interval-based temporal functional dependencies has been previously dealt with only in [29], which is a preliminary short version of this paper.

In this paper we shall specifically focus on the discovery of interval-based temporal functional dependencies (ITFDs), according to their definition proposed in [13,14]. More precisely, even according to the previously cited contributions, we shall use the relational model extended to consider the temporal data dimension as the theoretical framework for representing and reasoning on (temporal) data.

The main original aspects of this paper may be summarized as in the following.

– We formally introduce the concept of *Approximate Interval-Based Temporal Functional Dependency* (AITFD) within the relational framework and discuss its

meaning by explicitly considering Allen's interval relations. A general clinical scenario related to patient therapies will be discussed to motivate and exemplify our approach.
– We address the complexity for the problem of deriving AITFDs given a (relational) dataset, namely *ITFD-Approx*. More specifically we focus on the data complexity of the *Maximal Consistency* problem. Such a problem consists of determining the maximum cardinality of the subsets of a (data) relation that satisfy a given ITFD. For each Allen's interval relation we provide the class of computational complexity for the related problem. For each result, we provide detailed proofs and, in some cases, we introduce and discuss some auxiliary data structures, we shall use to build proofs.

In the following, Section 2 introduces related work dealing with temporal functional dependencies and approximate functional dependencies. Section 3 discusses a motivating scenario, we shall use throughout the paper to motivate and exemplify our proposal. Section 4 introduces the basic definitions regarding approximate ITFDs and then introduces the formalization of the Maximal Consistency problem that will be analyzed in the remainder of the paper. Section 5 deals with the Maximal Consistency problems for approximate ITFDs restricted to the *equal*, *starts*, and *finishes* Allen's interval relations, for which solutions are similar and straightforward. Section 6 deals with the Maximal Consistency problem restricted to the *before* interval relation: it is less straightforward than the problems discussed for the previous relations, but still simple. In Section 7 we provide a polynomial-time algorithm for the problem considering the *during* interval relation. This is one of the main results of this paper: the *during* relation has been extensively studied in literature and both positive and negative results from the point of view of decidability have been provided [4,5,27]. Here we give a positive result from the point of view of complexity, showing that the problem involving the *during* relation is polynomially tractable. Section 8 completes the analysis by dealing with interval relations *meets* and *overlaps*: this is the second important result of the paper and it shows that the problem for these two cases is NP-complete, by means of two closely related reductions from the classical $Max2Sat$ decision problem. Section 9 introduces and discusses some analogies between $AITFD$ discovery and the theoretical research topic on database repairing. Finally, Section 10 draws some conclusions and sketches out some possible directions for future research.

2 Background and Related Work

We recall here the definition of functional dependency (FD), and then introduce its extensions: point-based and interval-based temporal functional dependencies (TFD) and approximate functional dependency (AFD). Such concepts will lead to the definition of approximate interval temporal functional dependency ($AITFD$) of Section 4, where $AITFD$ inherits the properties both from AFD and from interval-based TFDs.

The concept of functional dependency (FD) comes from the relational database theory and is defined as follows [7]:

Definition 1 (*Functional Dependency*) Let r be a relation over the relational schema R: let $X, Y \subseteq R$ be attributes of R. We assert that r satisfies functional

dependency $X \rightarrow Y$ (written as $r \models X \rightarrow Y$) if the following condition holds: $\forall t, t' \in r(t[X] = t'[X] \Rightarrow t[Y] = t'[Y])$.

Informally, for all the couples of tuples t and t' showing the same value(s) on X, the corresponding value(s) on Y for those tuples are identical.

2.1 Temporal Functional Dependencies

Moving closer to the main kind of temporal features we shall consider here, several kinds of temporal functional dependencies (TFDs) have been proposed in the literature, usually as temporal extensions of the widely know (atemporal) functional dependencies [34].

In the following, we provide a short overview of the main formalisms for TFDs proposed in the literature. All the introduced temporal data models consider some kind of (temporal) extensions to the classical relational model. Jensen et al. propose a bitemporal data model that allows one to associate both valid and transaction times with data [21]. They define TFDs as FDs that must be satisfied at any bitemporal point (i.e., representing both valid and transaction times: *chronon* in the authors' terminology). As an example, let *Profs* be a temporal relation schema with the set of atemporal attributes $U = \{profId, salary, role, project\}$. The condition "*at any time, the salary of a professor uniquely depends on his role*" can be expressed by TFD $role \rightarrow^{T} salary$.

Bettini, Jajodia, and Wang's notion of TFD takes advantage of time granularity [32]: time granularity is a partition of a time domain in groups of indivisible units called *granules*. Bettini, Jajodia, and Wang's TFDs allow one to specify conditions on tuples associated with granules of a given granularity and grouped according to a coarser granularity. As an example, if we consider the temporal relation schema $Profs$ with attributes $\{profId, salary, role, project\}$ and associated with granularity Month, the constraint "*for any given year, professor with the same role cannot have different salaries the same year; however, their salary may change from one year to the next one*" is captured by TFD $role \rightarrow_{\mathsf{Year}} salary$.

A general formalism for TFDs on complex (temporal) objects has been proposed by Wijsen [33]. It is based on a data model that extends the relational model with the notion of object identity, which is preserved through updates, and with the ability of dealing with complex objects, that is, objects that may have other objects as components. It has been shown that the class of Wijsen's TFDs subsumes the class of Bettini et al.'s TFDs [33]. More precisely, Bettini et al.'s TFDs are exactly TFDs on chronologies (i.e.time relationships between time points representing granularities). As an example, it is possible to express the condition "*professors cannot have different salaries over two consecutive time points if their role does not change*" by means of TFD $Emp : profId, role \ \mathbf{N} \ salary$.

Vianu proposes a simple extension to the relational model in order to describe the evolution of a database over time [31]. According to it, a temporal database is viewed as a sequence of instances (states) over time. A change in the state of the database is produced by the execution of an update, an insertion or a deletion. A *database sequence* is a sequence of consecutive instances of the database, together with "*update mappings*" from one instance (the "old" one, with attributes Y denoted as $\overset{\vee}{Y}$) to the next instance (the "new" one, with attributes Y denoted as

$\hat{Y}$). Tuples are viewed as representations of domain objects. Since a tuple and its updated version represent the same object, tuples preserve their identity through updates. As an example, condition: "*new salaries of professors depend uniquely on their current and previous roles*" is captured by the DFD $\overset{\vee}{role}\ \overset{\wedge}{role} \rightarrow \overset{\wedge}{salary}$ over the set of attributes $U = \{profId, salary, role\}$.

Recently, Combi et al. proposed a framework for TFDs that subsumes and extends the considered previous proposals [11]. The proposed framework is based on a simple temporal relational data model based on the notion of *temporal relation*, i.e. a relation extended with a timestamping temporal attribute `VT`, representing *valid time*, i.e. the time when the fact is true in the represented real world [10].

Two temporal views have been introduced: they allow one to join tuples that represent relevant cases of (temporal) evolution. On the base of the introduced data model, and leveraging the introduced temporal views, TFDs may be expressed by the syntax $[E\text{-}Exp(R), t\text{-}Group]X \rightarrow Y$ where $E\text{-}Exp(R)$ is a relational expression on R, called *evolution expression*, $t\text{-}Group$ is a mapping $\mathbb{N} \rightarrow 2^{\mathbb{N}}$, called *temporal grouping*, and $X \rightarrow Y$ is a functional dependency. As for the semantics, similarly to the case of standard FDs, a TFD is a statement about admissible temporal relations on a temporal relation schema R with attributes $U \cup \{VT\}$. Four different classes of TFD have been identified in [11]:

- *Pure temporally grouping* TFD: $E\text{-}Exp(R)$ returns the original temporal relation r. Rules of this class force the FD $X \rightarrow Y$, where $X, Y \subseteq U$, to hold over all sets which include all tuples whose `VT` belongs to the same temporal group;
- *Pure temporally evolving* TFD: $E\text{-}Exp(R)$ collects all the tuples modelling the evolution of an object. No temporal grouping exists: that is, the temporal grouping collects all the tuples of r in one unique set;
- *Temporally mixed* TFD: the expression $E\text{-}Exp(R)$ collects all the tuples modelling the evolution of the object. The temporal grouping is applied to the set of tuples generated by $E\text{-}Exp(R)$;
- *Temporally hybrid* TFD*s*. First, the evolution expression $E\text{-}Exp(R)$ selects those tuples of the given temporal relation that contribute to the modelling of the evolution of a real-world object (that is, it removes isolated tuples); then, temporal grouping is applied to the resulting set of tuples.

2.2 Interval-based Temporal Functional Dependencies

Let us now introduce the concept of Interval-based Functional Dependency and its underlying relational data model, we proposed in [14]. Given a linear order $\mathbb{O} = \langle O, < \rangle$, i.e. the time domain, an interval I over $\mathbb{O}$ is a pair $I = [b, e]$ where $b, e \in O$ and $b \leqslant e$. While the possible distinct relations between two points considering only the linear order are reduced to three (equality, successor, and predecessor), considering the order among two endpoints of two intervals leads us to have thirteen possible relations. Such a number is given by the possible ways one could arrange points of two distinct intervals $[b, e]$ and $[b', e']$ $(b \neq b' \vee e \neq e')$ on a total linear order, which are 6, plus their respective inverses and the equality relation $(b = b' \wedge e = e')$ for a total of thirteen distinct interval relations. Figure 1 depicts these relations, according to the notation proposed by Allen in [2]. It is worth to note that every relation has its dual, which is obtained by switching the

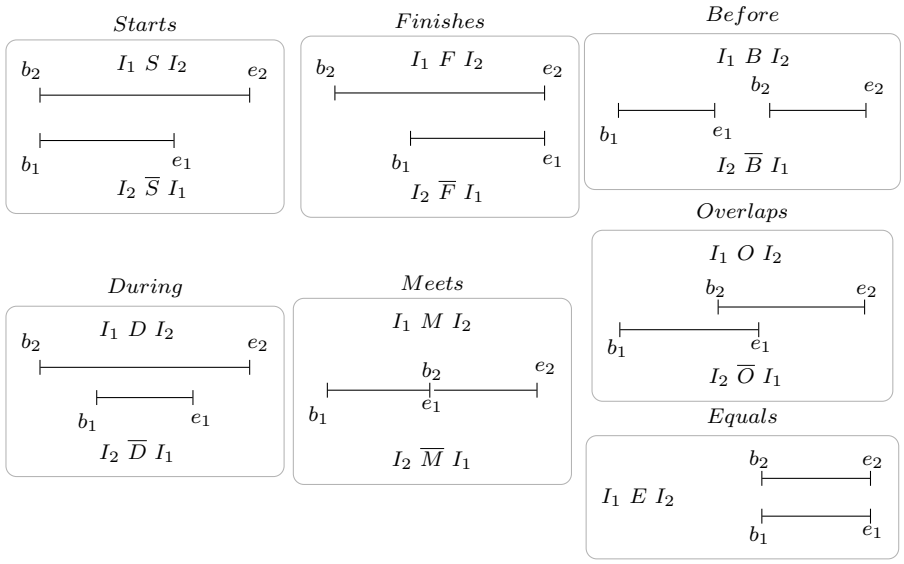


Fig. 1 The thirteen Allen's relations between intervals.

position of the two intervals. More precisely, given two intervals $I_1 = [b_1, e_1]$ and $I_2 = [b_2, e_2]$ we say that:

(1) I_1 E I_2 iff $b_1 = b_2$ and $e_1 = e_2$;
(2) I_1 M I_2 iff $e_1 = b_2$;
(3) I_1 S I_2 iff $b_1 = b_2$ and $e_1 < e_2$;
(4) I_1 F I_2 iff $b_1 > b_2$ and $e_1 = e_2$;
(5) I_1 O I_2 iff $b_1 < b_2$ and $b_2 < e_1 < e_2$;
(6) I_1 D I_2 iff $b_2 < b_1$ and $e_1 < e_2$;
(7) I_1 B I_2 iff $e_1 < b_2$.

The adopted data model is a simple temporal (relational) data model based on the concept of temporal relation. A temporal relation $\mathbf{r}$ is a relation on a temporal relation schema $\mathcal{R}$ defined on attributes $U \cup \{B, E\}$, where U represents a set of atemporal attributes and B, E are the temporal attributes describing the valid interval of a tuple. We assume that the domain of both attributes B and E is a totally ordered set $\mathbb{O}$. Clearly, a tuple $t \in \mathbf{r}$ satisfies $t[B] \leqslant t[E]$. To avoid ambiguities in the used terminology, we use *(temporal) instance* for "(temporal) relation" and *relation* for Allen's interval relations. Let us now consider the basic definition of *Interval-based Temporal Functional Dependency* (ITFD). We can consider only interval relations in the set $\mathcal{A} = \{E, S, F, B, D, M, O\}$. Indeed, in this case it is not meaningful to distinguish between a relation and its dual, as it will be clear from the following definition of interval-based temporal functional dependency.

Definition 2 Let X and Y be sets of atemporal attributes of a temporal relation schema $\mathcal{R} = R(U, B, E)$ and $\sim$ an Allen's Interval relation. An instance $\mathbf{r}$ of $\mathcal{R}$ satisfies an ITFD $X \rightarrow_{\sim} Y$ iff for each pair of tuples t_1 and t_2 such that $[t_1[B], t_1[E]] \sim [t_2[B], t_2[E]]$ and $t_1[X] = t_2[X]$, it is also true that $t_1[Y] = t_2[Y]$

For sake of brevity, when an instance $\mathbf{r}$ satisfies an ITFD $X \rightarrow_{\sim} Y$, we write $\mathbf{r} \models X \rightarrow_{\sim} Y$. Basically, ITFDs group tuples whose B and E attribute values satisfy interval relation $\sim$. In the above definition all the possible tuples having as valid interval either $[b, e]$ or $[b', e']$, where $[b, e] \sim [b', e']$, are considered together. If there exist two tuples where the B and E attribute values, match exactly points b, e, b', and e', respectively, and both tuples agree on the values of atemporal attributes X, then the ITFD imposes that both tuples must agree on the values of atemporal attributes Y.

2.3 Approximate Functional Dependencies

The concept of approximate functional dependency (AFD) derives from the concept of plain FD. Given an instance r where an FD holds for *most* of the tuples in r, we may identify *some* tuples, for which that FD does *not* hold. Consequently, we define some measurements over the error we make in considering the FD to hold on r. One measurement [22] is known as G_1 and considers the number of violating couples of tuples. Another measurement [22], known as G_2, considers the number of tuples which violate the functional dependency. In other words, G_1 counts the number of violations in the whole instance $\mathbf{r}$ with respect to the given FD, while G_2 counts the number of tuples, who participate in at least one violation of the given FD in $\mathbf{r}$. It is easy to see that G_1 is bounded by $|\mathbf{r}|^2$ while G_2 is bounded by $|\mathbf{r}|$. The most common measurement [22], known as G_3, considers the minimum number of tuples in r to be *deleted* for the FD to hold. Formally, $G_3(X \rightarrow Y, r) = |r| - max\{|s| \mid s \subseteq r \wedge s \models X \rightarrow Y\}$.

The related *scaled measurement* g_3 is defined as $g_3(X \rightarrow Y, r) = G_3(X \rightarrow Y, r)/|r|$.

We can now introduce here the definition of approximate functional dependency AFD as:

Definition 3 (*Approximate Functional Dependency*) Let r be an instance over the relational schema R: let $X, Y \subseteq R$ be attributes of R. Instance r fulfills an approximate functional dependency $X \xrightarrow{\varepsilon} Y$ (written as $r \models X \xrightarrow{\varepsilon} Y$) if $g_3(X \rightarrow Y, r) \leqslant \varepsilon$, where ε is the maximum acceptable error defined by the user.

Among the several AFDs that can be identified over an instance r, the minimal AFD is of particular interest, as many other AFDs can then be derived from the minimal one. We thus define the minimal AFD as follows:

Definition 4 (*Minimal* AFD) Given an AFD over r, we define $X \xrightarrow{\varepsilon} Y$ to be minimal for r if $r \models X \xrightarrow{\varepsilon} Y$ and $\forall X' \subset X$ we have that $r \not\models X' \xrightarrow{\varepsilon} Y$.

2.4 Approximate Temporal Functional Dependencies

According the the taxonomy proposed in [11] and discussed in Section 2.1, Combi et al. in [9] proposed approximate pure temporally grouping TFDs, where grouping is based either on *granularities* or on *sliding windows* (SW).

Definition 5 (*ATFD with Gran grouping*) Let r be an instance over the relational schema R with attributes $U \cup \{VT\}$: let $X, Y \subseteq U$ be attributes of R. Let $Gran$ be the reference granularity. Instance r satisfies the approximate temporal functional dependency on X and Y (written as $r \models [r, Gran]X \xrightarrow{\varepsilon} Y$) iff $g_3([r, Gran]X \rightarrow Y, r) \leqslant \varepsilon$.

That is, the percentage of tuples in the entire instance r to be *deleted* for an $ATFD$ to hold on all the tuples of r is less than ε; tuples of r are then grouped according to the granule of $Gran$ their `VT` value belongs to, to evaluate the considered $ATFD$. We recall that the count of tuples in r to be deleted refers to the entire instance r, and not to a sigle group - and one tuple may belong to one group only, if we use a $Gran$ grouping.

Definition 6 (*ATFD with SW grouping*) Let r be an instance over the relational schema R with attributes $U \cup \{VT\}$: let $X, Y \subseteq U$ be attributes of R. Let $\{i \dots i + k - 1\}$ be a sliding window (SW) of length k. Instance r satisfies an approximate temporal functional dependency on X and Y (written as $r \models [r, \{i \dots i+k-1\}]X \xrightarrow{\varepsilon} Y$) iff $g_3([r, \{i \dots i + k - 1\}]X \rightarrow Y, r) \leqslant \varepsilon$.

In [9] the authors considered as many SWs as possible, every SW sizing k elements: thus, the first considered sliding window is $i \dots i + k - 1$, the second considered sliding window is $i + 1 \dots i + k$, the third considered sliding window is $i + 2 \dots i + k + 1$, and so on. Every SW sets up a group (or *chain*) over which the $ATFD$ is checked. The $ATFD$ must hold, with an acceptable amount of error smaller than ε, over the entire database: we recall that, if we *delete* a tuple inside a SW, that tuple will remain *deleted* in *all* the SWs (either preceding or following the current SW) which include that tuple.

As for plain AFD, the concept of minimality has been introduced also for $ATFD$s [9].

3 A motivating scenario

In this section, we briefly introduce a real-world, yet general, example taken from clinical medicine, namely that of patient therapies, in order to provide a little insight on understanding both ITFDs and AITFDs. Most health care institutions collect a large quantity of clinical information about patient and physician actions, such as therapies and surgeries, as well as about health care processes, such as admissions, discharges, and exam requests. All these pieces of information are temporal in nature and the associated temporal dimension needs to be carefully considered in order to be able to properly represent clinical data and to reason about them [8].

Suppose we have patients who undergo several different therapies: each therapy can be supervised by a physician, and consists of the administration of some drug to the patient. Information about patients and therapies is stored in an instance according to the schema $PatTherapies(TherType, PatId, DrugCode, Qty, Phys, B, E)$, where *TherType* identifies a type of pharmacological therapy, *PatId* represents a patient ID, *DrugCode* and *Qty* the prescribed drug and its quantity, respectively, and *Phys* the physician who made the prescription (and is responsible for the therapy). Finally, attributes B and E represent the beginning and end time points

#	TherType	PatId	Phys	DrugCode	Qty	B	E
1	antiviral	1	House	0458	300	2	16
2	analgesics	1	Wilson	0976	200	3	11
3	cardiovascular	1	Wilson	0118	100	4	8
4	antipyretics	1	Wilson	0976	100	10	12
5	sedative	1	Foreman	0345	10	14	15
6	anxiolytic	1	House	0345	10	18	20
7	antiviral	2	Chase	0458	200	1	12
8	cardiovascular	2	Chase	0118	100	4	8
9	analgesics	2	Cameron	0976	150	6	11
10	antiviral	2	Cameron	0458	300	10	15
11	antiviral	1	House	0789	200	2	19

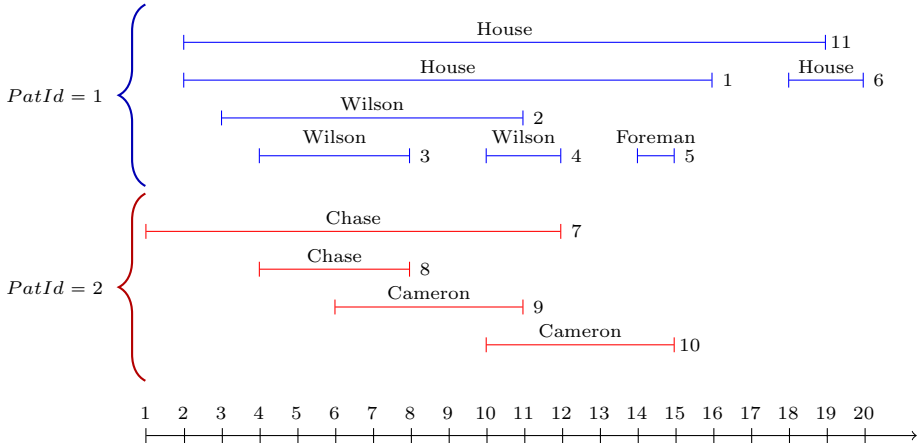


Fig. 2 An instance of relational schema *PatTherapies*, storing data about patient therapies, and its representation on the time line with values for attribute *Phys* (for each tuple its id is reported at the end of the interval).

of the tuple valid interval, respectively: they represent the bounds of the interval specified by the physician for each therapy. An instance of *PatTherapies* is provided in Figure 2.

A patient may have several drug administrations in the same period prescribed by different physicians. One may be interested in discovering some qualitative behaviour of such data. As an example, it may be asked if *in general* therapies occurring within the same period for the same patient are administered by the same physician. This is not a strict constraint for our instance but simply one possible behaviour that can be expressed by the ITFD $PatId \rightarrow_D Phys$. Thus, we are interested in finding the largest possible portion $\mathbf{r}' \subseteq \mathbf{r}$ such that $\mathbf{r}' \models PatId \rightarrow_D Phys$. In the example of Figure 2 this may be achieved by taking $\mathbf{r}' = \{t_2, t_3, t_4, t_5, t_6, t_8, t_9, t_{10}\}$. In this case, we may say that $PatId \rightarrow_D Phys$ holds on the 8/11 of the given instance. Another behaviour we may want to discover could be whether patient therapies are extended by the same physician. Such a behaviour is captured by the ITFD $PatId \rightarrow_O Phys$. This is a trickier behaviour than the one shown before, because a physician may extend a therapy that he/she has given to a patient but, implicitly, he/she cannot exceed the duration of a therapy given by another physician in the same period to the same patient. In Figure 2 we have that therapies concerning the first patient ($PatId = 1$) satisfy ITFD $PatId \rightarrow_O Phys$ without any error. On the contrary, therapies concerning

the second patient ($PatId = 2$) violate ITFD $PatId \rightarrow_O Phys$ because of the pair of tuples (t_7, t_{10}) and (t_8, t_9) (i.e Dr. Cameron is actually extending a therapy of Dr. Chase in both cases). Clearly both instances $\mathbf{r}' = \{t_1, t_2, t_3, t_4, t_5, t_6, t_9, t_{10}, t_{11}\}$ and $\mathbf{r}' = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_{11}\}$ satisfy ITFD $PatId \rightarrow_O Phys$ (solutions are not unique in general) and thus we may say that $PatId \rightarrow_O Phys$ holds on the 9/11 of the given instance. Even if the examples shown here use only one attribute in the set X for ITFD $X \rightarrow_\sim Y$, it does make no difference if $|X| > 1$. As a matter of fact, by Definition 2, tuples are grouped at the beginning according to their value on X and then each group is considered separately. For instance if one considers ITFD $TherType, PatId \rightarrow_\sim Y$, we have that tuples in Figure 2 are partitioned into 10 groups instead of two (i.e. we have then distinct values for the pair of attributes $TherType, PatId$). The rest of the paper is devoted to the study of such approximate ITFDs that we have seen here in an informal way, and especially to the study of the computational complexity involved in their calculation.

4 Approximate Interval-based Temporal Functional Dependencies

The following definition introduces the concept of Approximate ITFD ($AITFD$).

Definition 7 Let X and Y be sets of atemporal attributes of a temporal relation schema $\mathcal{R} = R(U, B, E)$, $\sim$ an Allen's Interval relation, and ϵ a real number $0 \leqslant \epsilon \leqslant 1$. An instance $\mathbf{r}$ of $\mathcal{R}$ satisfies an ITFD $X \rightarrow_\sim Y$ with approximation ϵ iff there exists a subset $\mathbf{r}' \subseteq \mathbf{r}$ for which $\mathbf{r} \backslash \mathbf{r}' \models X \rightarrow_\sim Y$ and $|\mathbf{r}'| \leqslant \epsilon \cdot |\mathbf{r}|$.

As for ITFDs, we write $\mathbf{r} \models_\epsilon X \rightarrow_\sim Y$ when $\mathbf{r}$ satisfies $X \rightarrow_\sim Y$ with approximation ϵ.

Given an instance $\mathbf{r}$, an ITFD $X \rightarrow_\sim Y$, and a real number $0 \leqslant \epsilon \leqslant 1$, we say that ITFD $X \rightarrow_\sim Y$ is ϵ-*maximal* on $\mathbf{r}$ iff $\mathbf{r} \models_\epsilon X \rightarrow_\sim Y$ and, for every $X' \rightarrow_\sim Y'$ such that $\mathbf{r} \models_\epsilon X' \rightarrow_\sim Y'$, we have either $X \nsubseteq X'$ or $Y \nsubseteq Y'$.

Given a set of Allen's interval relations $\mathcal{A}' \subseteq \mathcal{A}$, an instance $\mathbf{r}$, and a real number $0 \leqslant \epsilon \leqslant 1$, we define the set $ITFD(\mathcal{A}', \mathbf{r}, \epsilon) = \{X \rightarrow_\sim Y \ : \sim \in \mathcal{A}'$ and $\mathbf{r} \models_\epsilon X \rightarrow_\sim Y$ and $X \rightarrow_\sim Y$ is ϵ-maximal on $\mathbf{r}\}$. When $\mathcal{A}' = \mathcal{A} = \{E, S, F, B, D, M, O\}$ we simply omit it and we write $ITFD(\mathbf{r}, \epsilon)$.

Clearly, for every $\mathcal{A}', \mathcal{A}'' \subseteq \mathcal{A}$, for every instance $\mathbf{r}$, and for a every real number $0 \leqslant \epsilon \leqslant 1$, it holds that $ITFD(\mathcal{A}', \mathbf{r}, \epsilon) \cup ITFD(\mathcal{A}'', \mathbf{r}, \epsilon) = ITFD(\mathcal{A}' \cup \mathcal{A}'', \mathbf{r}, \epsilon)$ (e.g. $ITFD(\{S, F\}, \mathbf{r}, \epsilon) \cup ITFD(\{S, D, M\}, \mathbf{r}, \epsilon) = ITFD(\{S, F, D, M\}, \mathbf{r}, \epsilon)$).

It is worth to point out a crucial difference that holds in general between FDs and $AFDs$, and in particular between $ITFDs$ and $AITFDs$. Given an instance $\mathbf{r}$ of a schema $\mathcal{R}(U)$ and two non-empty set of attributes $X, Y \subseteq U$, it is easy to see that $\mathbf{r} \models X \rightarrow Y$ holds over $\mathbf{r}$ if and only if $\mathbf{r} \models X \rightarrow \{A\}$ for each attribute $A \in Y$. This does not hold in general if we move to $AFDs$. Suppose now that $\mathbf{r} \models_\epsilon X \rightarrow Y$ for some real number $0 \leqslant \epsilon \leqslant 1$. It implies that $\mathbf{r} \models X \rightarrow_\epsilon \{A\}$ for each $A \in Y$. However the opposite direction is not true in general. Indeed, for each $A \in Y$ let $\mathbf{r}_A$ be a set for which $\mathbf{r} \backslash \mathbf{r}_A \models X \rightarrow \{A\}$ and $|\mathbf{r}_A| \leqslant \epsilon \cdot |\mathbf{r}|$. It may be the case that $\mathbf{r}_A \neq \mathbf{r}_{A'}$ for each $A, A' \in Y$ with $A \neq A'$ and thus $|\bigcup_{A \in Y} \mathbf{r}_A| > \epsilon \cdot |\mathbf{r}|$.

To better explain this crucial difference, we may consider instance $\mathbf{r} = \{(1, 2, 3), (1, 1, 3), (1, 2, 1)\}$ over schema $\mathcal{R}(\{A_1, A_2, A_3\})$. It is easy to verify that both $\mathbf{r} \models_{1/3} \{A_1\} \rightarrow \{A_2\}$ ($\mathbf{r}_{A_2} = \{(1, 1, 3)\}$) and $\mathbf{r} \models_{1/3} \{A_1\} \rightarrow \{A_3\}$ ($\mathbf{r}_{A_3} = \{(1, 2, 1)\}$) hold, but $\mathbf{r} \models_{1/3} \{A_1\} \rightarrow \{A_2, A_3\}$ does not hold (i.e. you can keep only one tuple if you

want to satisfy such an FD and this means $\epsilon \geqslant 2/3$). Obviously, this distinction holds between $ITFDs$ and $AITFDs$ as well. For the sake of simplicity, in the following we will consider only $(A)ITFDs$ $X \rightarrow_{\sim} Y$ where Y is a singleton set (e.g. $Y = \{A\}$) and for the sake of readability we shall omit the curly brackets around sets when writing an $AITFD$.

4.1 Inferring AITFDs

In this section we introduce the problem of inferring approximate ITFDs and some closely related problems.

Problem 1 Given an instance $\mathbf{r}$ of some temporal relation schema $\mathcal{R} = R(U, B, E)$ and a real number $0 \leqslant \epsilon \leqslant 1$, the problem $ITFD\text{-}Infer(\mathbf{r}, \epsilon)$ consists of determining the set $ITFD(\mathbf{r}, \epsilon)$.

Given a set of interval relations $\mathcal{A}' \in \mathcal{A}$, the $\mathcal{A}'$-restricted version of the $ITFD\text{-}Infer(\mathbf{r}, \epsilon)$ problem is addressed as $\mathcal{A}'\text{-}ITFD\text{-}Infer(\mathbf{r}, \epsilon)$. For instance, the problem $\{E, S, F\}\text{-}ITFD\text{-}Infer(\mathbf{r}, \epsilon)$ takes as input an instance $\mathbf{r}$ for a temporal relation schema $\mathcal{R} = R(U, B, E)$ and a real number $0 \leqslant \epsilon \leqslant 1$, and returns the set $ITFD(\{E, S, F\}, \mathbf{r}, \epsilon)$. By means of a straightforward adaptation of results given by Mannila et al. in [25], we can give the following result.

Theorem 1 *For every temporal relational schema $\mathcal{R} = R(U, B, E)$, there exists an instance $\mathbf{r}$ of it and a real number $0 \leqslant \epsilon \leqslant 1$ for which $|ITFD(\mathbf{r}, \epsilon)| = \Omega(2^{\frac{|U|}{2}})$.*

Theorem 1 states that there exists $k \in \mathbb{R}^+$, for which, for every temporal relational schema $\mathcal{R} = R(U, B, E)$ we may build an instance $\mathbf{r}$ of $\mathcal{R}$ and determine a real number $0 \leqslant \epsilon \leqslant 1$ for which $|ITFD(\mathbf{r}, \epsilon)| \geqslant k \cdot 2^{\frac{|U|}{2}}$.

Theorem 1 has serious implications when searching for $ITFD(\mathbf{r}, \epsilon)$. Indeed, due to this result, we know that the worst case complexity of determining $ITFD(\mathbf{r}, \epsilon)$ cannot be less than exponential in the number of attributes of relational schema $\mathcal{R}$. As a consequence of this result, we have that every algorithm that solves $ITFD\text{-}Infer(\mathbf{r}, \epsilon)$ cannot avoid to test an exponential number of $ITFDs$. However, reasonable temporal relational schemata $\mathcal{R} = R(U, B, E)$ feature a small number of atemporal attributes: usually $|U| \leqslant 100$. Moreover, several heuristics have been studied to keep the number of tested dependencies feasible in the context of standard FDs [18,26] and such methods can be directly applied in solving $ITFD\text{-}Infer(\mathbf{r}, \epsilon)$.

To the best of our knowledge, every approach proposed in literature progressively sharpens the set of dependencies to be tested by pruning it on the base of results of some already tested dependencies. Indeed, testing a given dependency (approximate or not) is assumed to be an easier task w.r.t. to the asymptotic (data) complexity. This is true for FDs (approximate or not) [19,22] and even for non-approximate ITFDs [14], but, as we shall see, it is not true in general for approximate ITFDs. Since it is hopeless to avoid the test of an exponential number of ITFDs, our attention moves on assessing the (data) complexity of the following problem.

Problem 2 Given an instance $\mathbf{r}$ of some temporal relation schema $\mathcal{R} = R(U, B, E)$, a real number $0 \leqslant \epsilon \leqslant 1$, and an ITFD $X \rightarrow_{\sim} Y$, problem $ITFD\text{-}Approx(X \rightarrow_{\sim} Y, \mathbf{r}, \epsilon)$ consists of determining if $\mathbf{r} \models_{\epsilon} X \rightarrow_{\sim} Y$.

We shall see that the chosen interval relation $\sim$ plays a crucial role for the complexity of problem $ITFD\text{-}Approx(X \rightarrow_{\sim} Y, \mathbf{r}, \epsilon)$.

Given a temporal relation schema $\mathcal{R} = R(U, B, E)$, a singleton set of atemporal attributes $Y \in U$ and an interval relation $\sim \in \mathcal{A}$, an instance $\mathbf{r}$ is $\sim$-*consistent* with respect to Y iff for every pair of tuples $t, t' \in \mathbf{r}$ we have $I(t) \sim I(t')$ implies $t[Y] = t'[Y]$. Given an instance $\mathbf{r}$ of a temporal relation schema $\mathcal{R} = R(U, B, E)$ and a (singleton) set of atemporal attributes $Y \in U$, we say that a subset $\mathbf{r}' \subset \mathbf{r}$ is *monochromatic* with respect to Y if and only if for every pair of tuples $t, t' \in \mathbf{r}'$ we have $t[Y] = t'[Y]$. Monochromatic property of sets will appear often in the result given in the rest of the paper. In order to simplify the problem and reduce it to a simplest representation to deal with we introduce the following definition.

Definition 8 Given a singleton set of atemporal attributes $Y \subseteq U$, an interval relation $\sim \in \mathcal{A}$, and an instance $\mathbf{r}$, we define the set of all the $\sim$-consistent subsets of $\mathbf{r}$ w.r.t. Y as $\mathcal{S}^{\mathbf{r}}_{\sim,Y} = \{\mathbf{r}' : \mathbf{r}' \subseteq \mathbf{r}$ *and* $\mathbf{r}'$*is* $\sim -$*consistent w.r.t.* $Y\}$. The following problem is closely related to Problem 2.

Problem 3 Given an instance $\mathbf{r}$ of some temporal relation schema $\mathcal{R} = R(U, B, E)$, an interval relation $\sim \in \mathcal{A}$, and a singleton set $Y \subseteq U$, problem $\sim$-$Max\text{-}Consistent(Y, \mathbf{r})$ consists of determining the value of $\max_{\mathbf{r}' \in \mathcal{S}^{\mathbf{r}}_{\sim,Y}} |\mathbf{r}'|$.

Basically, considering all the possible $\sim$-*consistent* subsets $\mathbf{r}'$ of $\mathbf{r}$ w.r.t. Y, we are looking for the ones with maximum cardinality $|\mathbf{r}'|$, and in particular we are interested in determining such value. A simpler problem than Problem 3 is the problem $\sim$-$Consistent$.

Problem 4 Given an instance $\mathbf{r}$ of some temporal relation schema $\mathcal{R} = R(U, B, E)$, an interval relation $\sim \in \mathcal{A}$, and a singleton set $Y \subseteq U$, problem $\sim$-$Consistent(Y, \mathbf{r})$ consists of determining whether $\mathbf{r} \in \mathcal{S}^{\mathbf{r}}_{\sim,Y}$ holds.

Problem 4 consists of verifying if the whole instance is $\sim$-*consistent* w.r.t. Y. A simple algorithm to solve such a problem consists of comparing each pair of tuples t, t' with $I(t) \sim I(t')$. If a pair with $t[Y] \neq t'[Y]$ is found, the algorithm returns NO otherwise it returns YES. Such an algorithm has $\mathcal{O}(|\mathbf{r}|^2)$ complexity. However, for this and some related problems, faster algorithms have been proposed (see, for example, [14]) and the resulting complexity is $\mathcal{O}(n \cdot \log n)$. The strong connection between Problem 2 and Problem 3 is provided by the following theorem.

Theorem 2 *For every temporal relation schema* $\mathcal{R} = R(U, B, E)$*, every instance* $\mathbf{r}$ *of* $\mathcal{R}$*, every real number* $0 \leqslant \epsilon \leqslant 1$*, and every ITFD* $X \rightarrow_{\sim} Y$*, given a super-additive*[1] *function* $f : \mathbb{N} \rightarrow \mathbb{N}$*, we have that the following two properties hold:*

1. *if* $\sim$-$MaxConsistent$ $(Y, \mathbf{r})$ *is decidable with complexity* $\mathcal{O}(f(|\mathbf{r}|))$*, then* $ITFD\text{-}Approx(X \rightarrow_{\sim} Y, \mathbf{r}, \epsilon)$ *is decidable with complexity* $\mathcal{O}(f(|\mathbf{r}|))$*;*
2. *if* $ITFD\text{-}Approx(X \rightarrow_{\sim} Y, \mathbf{r}, \epsilon)$ *is decidable with complexity* $\mathcal{O}(f(|\mathbf{r}|))$*, then* $\sim$-$MaxConsistent$ $(Y, \mathbf{r})$ *is decidable with complexity* $\mathcal{O}(\log(|\mathbf{r}|) \cdot f(|\mathbf{r}|))$.

[1] A function $f : \mathbb{N} \rightarrow \mathbb{N}$ is *super-additive* if for every pair of elements $x, y \in \mathbb{N}$ we have that $f(x + y) \geqslant f(x) + f(y)$

Proof The proof of both properties is straightforward. For property 1 suppose that we have a function $F_\sim$ that takes as input an instance $\mathbf{r}$ of $\mathcal{R} = R(U, B, E)$ and a singleton set of attributes $Y \subseteq U$ and returns $\max_{\mathbf{r}' \in \mathcal{S}^{\mathbf{r}}_{\sim,Y}} |\mathbf{r}'|$. Moreover the complexity of $F_\sim$ is $\mathcal{O}(f(|\mathbf{r}|))$.

Consider the following algorithm:

1. let $x_1, \ldots, x_n$ be all the distinct values of attributes in X and let $\mathbf{r} = \mathbf{r}_{x_1} \cup \ldots \cup \mathbf{r}_{x_n}$ be the partition of $\mathbf{r}$ into n instances, for which for each $1 \leqslant i \leqslant n$ we have that $\pi_X(\mathbf{r}_{x_i}) = x_i$ (i.e tuples are partitioned according to their values for the attributes in X);
2. let $N = \Sigma_{1 \leqslant i \leqslant n} F_\sim(\mathbf{r}_{x_i}, Y)$;
3. if $|\mathbf{r}| - N \leqslant \epsilon \cdot |\mathbf{r}|$ the answer to $ITFD\text{-}Approx(X \rightarrow_\sim Y, \mathbf{r}, \epsilon)$ is YES, otherwise the answer is NO.

It is easy to see that the above procedure solves correctly the problem $ITFD\text{-}Approx(X \rightarrow_\sim Y, \mathbf{r}, \epsilon)$. The complexities of each step are:

1. $\mathcal{O}(|\mathbf{r}|)$ for step 1;
2. $\Sigma_{1 \leqslant i \leqslant n}\ \mathcal{O}(f(|\mathbf{r}_{x_i}|))$ for step 2, which is equal to $\mathcal{O}(f(|\mathbf{r}|))$ since f is super-additive;
3. $\mathcal{O}(1)$ for step 3.

Summing up, the total complexity is $\mathcal{O}(|\mathbf{r}| + f(|\mathbf{r}|))$ which turns out to be $\mathcal{O}(f(|\mathbf{r}|))$ since f is super-additive.

For property 2, suppose that we have a function $G_\sim$ that takes as input an instance $\mathbf{r}$ of $\mathcal{R} = R(U, B, E)$, an ITFD $X \rightarrow_\sim Y$, and a real value ϵ, and answers correctly to the problem $ITFD\text{-}Approx(X \rightarrow_\sim Y, \mathbf{r}, \epsilon)$. Moreover, the complexity of $G_\sim$ is $\mathcal{O}(f(|\mathbf{r}|))$.

Suppose now that, given an instance $\mathbf{r}$ of $\mathcal{R} = R(U, B, E)$ and $Y \subseteq U$, we want to solve the problem $\sim\text{-}MaxConsistent\ (Y, \mathbf{r})$.

Consider the following algorithm:

1. let $\mathbf{r}'$ the instance of relation $\mathcal{R} = R(U \cup \{X'\}, B, E)$ where $X' \notin U$ and for each $t \in \mathbf{r}'$ we have $t[U, B, E] \in \mathbf{r}$ and $t[X'] = 0$;
2. set $min = 0$ and $max = |r|$;
3. if $max = min$ return max;
4. if $G_\sim(X' \rightarrow_\sim Y, \mathbf{r}', (\lceil (max + min)/2 \rceil)/|\mathbf{r}'|) = YES$, put $min = \lceil (max + min)/2 \rceil$, otherwise put$max = \lceil (max + min)/2 \rceil$;
5. goto step 3.

It is easy to prove that such an algorithm solves the problem $\sim\text{-}MaxConsistent\ (Y, \mathbf{r})$. This procedure basically applies a dichotomic search of the maximal value ϵ, for which $G_\sim(X' \rightarrow_\sim Y, \mathbf{r}', \epsilon)$ holds. Once the desired ϵ is found, it is easy to provide the answer to the problem $\sim\text{-}MaxConsistent\ (Y, \mathbf{r})$ by means of a simple calculation. For the complexity evaluation we have that the dichotomic search implies the application of the function $G_\sim$ a number of times which is logarithmic in the size of $|\mathbf{r}|$. Thus the overall complexity turns out to be $\mathcal{O}(\log(|\mathbf{r}|) \cdot f(|\mathbf{r}|))$.

Such result allows us to focus on properties and computational complexity of Problem 3, knowing in advance that they hold also for Problem 2.

Theorem 3 *For every temporal relation schema $\mathcal{R} = R(U, B, E)$, every instance $\mathbf{r}$ of it, every interval relation $\sim \in \mathcal{A}$, and every set $Y \subseteq U$, problem $\sim\text{-}MaxConsistent(Y, \mathbf{r})$ belongs to the complexity class NP.*

Proof It is sufficient to provide a non deterministic polynomial algorithm that, given $\sim \in \mathcal{A}$, solves the problem $\sim$-$MaxConsistent(Y, \mathbf{r})$, for every instance $\mathbf{r}$ of a relation $\mathcal{R} = R(U, B, E)$ with $Y \in U$. Recall that we have for free from [14] that $\sim$-$Consistent(Y, \mathbf{r})$ can be checked in $\mathcal{O}(|\mathbf{r}| \cdot \log(|\mathbf{r}|))$ for every $\sim \in \mathcal{A}$. Then we can claim that there exists a function $F_\sim$ in NP that takes as input an instance $\mathbf{r}$ of relation $\mathcal{R} = R(U, B, E)$, an attribute $Y \in U$, and a number k, and answers correctly whether there exists a subset $\mathbf{r}' \subset \mathbf{r}$ which is $\sim$-$Consistent$ w.r.t. Y and $|\mathbf{r}'| \geqslant k$. To prove this it suffices to consider the following non-deterministic algorithm for calculating $F_\sim$: (i) guess a subset $\mathbf{r}' \subset \mathbf{r}$ with $|\mathbf{r}'| \geqslant k$ (ii) if $\sim$-$Consistency$ holds for $\mathbf{r}'$ the answer is YES, otherwise the answer is NO. It is easy to see that step (i) works in polynomial-time in $|\mathbf{r}|$, while step (ii) works in deterministic polynomial-time in $|\mathbf{r}|$. Using a dichotomic search, we can thus provide the following algorithm:

1. set $min = 0$ and $max = |r|$;
2. if $max = min$ return max;
3. if $F_\sim(\mathbf{r}, Y, \lceil (max + min)/2 \rceil) = YES$ put $min = \lceil (max + min)/2 \rceil$; otherwise, put $max = \lceil (max + min)/2 \rceil$;
4. goto step 2.

Such a procedure applies $\log(|\mathbf{r}|)$ times function $F_\sim$ and thus its complexity turns out to belong to the NP class.

For our purposes it is better to focus on the *evaluation* version of problem $\sim$-$MaxConsistent$, while in classical complexity theory usually problems are given in their *recognition* version [28]. The recognition version of $\sim$-$MaxConsistent$ adds a natural number $L \geqslant 0$ to the input of the problem and consists of simply answering YES or NO to the following question: *"Does a $\sim$-consistent w.r.t. Y subset $\mathbf{r}' \subseteq \mathbf{r}$ exist, such that $|\mathbf{r}'| \geqslant L$?"*. Under the assumption that the solution of the *evaluation* version can be logarithmically encoded, then both the evaluation and the recognition versions belong to the same complexity class and one of them is NP-complete if and only if even the other one is NP-complete [28]. It is straightforward to show that the logarithmic representation of the solution of $\sim$-$MaxConsistent$ is bounded by the size of input (i.e., it is bounded by $\lfloor \log |\mathbf{r}| \rfloor + 1$ which is better than polynomial). As the previous statement directly applies, from now on we can focus only on the evaluation version of problem $\sim$-$MaxConsistent$. In the following, we shall see that the tractability of problem $\sim$-$MaxConsistent$ depends on the chosen interval relation $\sim$.

We shall see that for interval relations S, E and F the $\sim$-$MaxConsistent$ problem turns out to be polynomial in the size of the input instance $\mathbf{r}$. The proposed algorithms for solving problem $\sim$-$MaxConsistent$ with $\sim \in \{S, E, F\}$ are given in Section 5, and each of them has complexity $\mathcal{O}(|\mathbf{r}| \cdot \log |\mathbf{r}|)$. In Section 6 we deal with problem B-$MaxConsistent$ which has still polynomial complexity but requires a more involved treatment. The proposed algorithm for solving the problem B-$MaxConsistent$ turns out to have complexity $\mathcal{O}(|\mathbf{r}|^4)$. In Section 7 we deal with problem D-$MaxConsistent$. Such a problem turns out to be polynomially solvable and we provide an algorithm that works in $\mathcal{O}(|\mathbf{r}|^{10})$ deterministic time. This is the most extended section of the paper since proving the underlying ideas and the soundness and completeness of the proposed algorithm for D-$MaxConsistent$ requires the introduction of a spatial representation of intervals as well as some preliminary results. Finally, in Section 8 we deal with the

Algorithm 5.1: E-MaxConsistency$(Y, \mathbf{r})$

```
r̄ ← r lexicographically ordered on B, E, Y
Max ← 0
YCount ← 1
MaxYCount ← 0
for i = 2, ..., |r̄|
  do
    if I(t̄_i) ≠ I(t̄_{i−1})
      then
        if YCount > MaxYCount
          then Max ← Max + YCount
          else Max ← Max + MaxYCount
        YCount ← 1
        MaxYCount ← 0
      else
        if t̄_i[Y] ≠ t̄_{i−1}[Y]
          then
            if MaxYCount < YCount
              then MaxYCount ← YCount
            YCount ← 1
          else YCount ← YCount + 1
if YCount > MaxYCount
  then Max ← Max + YCount
  else Max ← Max + MaxYCount
return (Max)
```

complexity of $\sim$-*MaxConsistent* problem when $\sim \in \{M, O\}$. With two very similar reductions to $Max2Sat$ we prove that both the problems are NP-Complete.

5 Maximal Consistency for E, S, and F cases

In this section we deal with (the complexity of) problem $\sim$–*MaxConsistent* when $\sim \in \{E, S, F\}$. These three cases are dealt with in the very same way and allow us to introduce some basic concepts concerning the verification of approximate functional dependencies in a gentle way.

Let us consider interval relation E. The key ingredient for the low-complexity of such problem consists of considering a partition for $\mathbf{r}$ as $\mathbf{r} = \mathbf{r}_{x_1,y_1} \cup \ldots \cup \mathbf{r}_{x_n,y_n}$, where, for each $1 \leqslant i \leqslant n$ and each tuple $t \in \mathbf{r}_i$, we have $(t[B], t[E]) = (x_i, y_i)$. It is easy to observe that, according to Definition 2, *E-Consistency* may be violated only by pairs of tuples t, t' inside the same partition $\mathbf{r}_i$ with $t[Y] \neq t'[Y]$. Then it is sufficient to select for each partition the value $M_i = \max_{y \in Dom(Y)} |\{t \in \mathbf{r}_i : t[Y] = y\}|$, and thus $E\text{-}MaxConsistent(Y, \mathbf{r}) = \Sigma_{1 \leqslant i \leqslant n} M_i$. For the S and F cases it suffices to build the partition taking into account values $t[B]$ and $t[E]$, respectively, instead of pair $(t[B], t[E])$ used for interval relation E. More precisely, let $\mathbf{r} = \mathbf{r}_{x_1} \cup \ldots \cup \mathbf{r}_{x_n}$, where for each $1 \leqslant i \leqslant n$ and each tuple in $t \in \mathbf{r}_i$, we have $t[B] = x_i$, for interval relation S. For interval relation F, $\mathbf{r} = \mathbf{r}_{y_1} \cup \ldots \cup \mathbf{r}_{y_n}$ and for each $1 \leqslant i \leqslant n$ and each tuple in $t \in \mathbf{r}_i$, we have $t[E] = y_i$.

The algorithm for *E-MaxConsistent* is described through the pseudocode of Algorithm 5.1. The algorithm first sorts the input instance $\mathbf{r}$ lexicographically

on attributes B, E, Y (for attributes B and E we make use of the order $\mathbb{O}$, for attribute Y any order will be suitable). Instance $\overline{\mathbf{r}}$ keeps the result of such sorting operation: its tuples $\bar{t}_1, \ldots, \bar{t}_{|\mathbf{r}|}$ are sorted lexicographically on B, E, Y. Then, we perform a linear parsing of $\overline{\mathbf{r}}$, collecting for each interval $[b, e] \in Intervals(\overline{\mathbf{r}})$ the cardinality of a maximal subset of $\overline{\mathbf{r}}$, for which all its tuples $\bar{t}$ have $I(t) = [b, e]$ and feature the same value for attributes in Y. It is easy to see that the sum of all these values represents the solution for problem E-$MaxConsistent(Y, \mathbf{r})$. The algorithm for problem S-$MaxConsistent(Y, \mathbf{r})$ (resp. F-$MaxConsistent(Y, \mathbf{r})$) is similar. It is enough to lexicographically sort $\mathbf{r}$ on B, Y (resp. E, Y) and to replace the test $I(\bar{t}_i) \neq I(\bar{t}_{i-1})$ with $\bar{t}_i[B] \neq \bar{t}_{i-1}[B]$ (resp. $\bar{t}_i[E] \neq \bar{t}_{i-1}[E]$) in Algorithm 5.1. We conclude this section with the following result.

Theorem 4 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$, any attribute $Y \in U$, and any instance $\mathbf{r}$ of $\mathcal{R}$, the complexity of problem $\sim$-$MaxConsistent(Y, \mathbf{r})$ with $\sim \in \{E, S, F\}$ is $\mathcal{O}(|\mathbf{r}| \cdot \log |\mathbf{r}|)$.*

Proof Consider Algorithm 5.1, which solves the problem of E-$MaxConsistent(Y, \mathbf{r})$. The S and F cases are dealt with by algorithms that behave in a similar way, leaving the complexity unchanged. The algorithm features one cycle that iterates $|\overline{\mathbf{r}}| - 1$ times on $\overline{\mathbf{r}}$, where $\overline{\mathbf{r}}$ is the lexicographically ordered counterpart of $\mathbf{r}$ on attributes B, E, Y. Such an ordering requires $\mathcal{O}(|\mathbf{r}| \cdot \log(|\mathbf{r}|))$ steps and thus its complexity dominates the linear complexity of the rest of the algorithm.

6 Maximal Consistency for the B case

In this section we deal with the complexity of problem B-$MaxConsistency$. Let $\mathbf{r}_B = \{t \ : \ t \in \mathbf{r} \ \wedge \ \exists t' \in \mathbf{r}$ such that $I(t) \ B \ I(t') \ \vee \ I(t) \ \overline{B} \ I(t')\}$ be the set of all tuples t in $\mathbf{r}$ such that there exists a tuple t' for which either $I(t) \ B \ I(t')$ or $I(t) \ \overline{B} \ I(t')$. Instance $\mathbf{r}_B$ turns out to be crucial for verifying if the whole instance $\mathbf{r}$ is $B - Consistent$ as pointed out by the following lemma.

Lemma 1 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$ and any instance $\mathbf{r}$ of $\mathcal{R}$, we have that $\mathbf{r}$ is B-consistent with respect to an attribute $Y \in U$ if and only if for each pair of tuples $t, t' \in \mathbf{r}_B$ we have $t[Y] = t'[Y]$.*

Proof We begin with the left to right direction. Suppose by contradiction that there exists a pair of tuples $t, t' \in \mathbf{r}_B$ with $t[Y] \neq t'[Y]$. Since $\mathbf{r}$ is B-*consistent*, we have that neither $I(t) \ B \ I(t')$ nor $I(t) \overline{B} \ I(t')$ hold. Thus, the following cases may arise (for a graphical account of them, refer to Figure 3):

- $I(t) = I(t')$. By definition of $\mathbf{r}_B$ we have that there exists a tuple t'' with either $I(t'') \ B \ I(t)$ or $I(t) \ B \ I(t'')$ and for B-$Consistency$ it holds $t''[Y] = t[Y]$. However we have that either $I(t'') \ B \ I(t')$ or $I(t') \ B \ I(t'')$, since $I(t) = I(t')$. Being $t''[Y] = t[Y] \neq t'[Y]$, it contradicts the B-$Consistency$ assumption;
- $I(t) \ S \ I(t')$. By definition of $\mathbf{r}_B$ we have that there exists a tuple t'' with either $I(t'') \ B \ I(t)$ or $I(t) \ B \ I(t'')$ and from B-$Consistency$ we know that $t''[Y] = t[Y]$ holds. If $I(t'') \ B \ I(t)$, we have that $I(t'') \ B \ I(t')$ and thus $t''[Y] \neq t[Y]$, which contradicts the B-$Consistency$ assumption. If $I(t') \ B \ I(t'')$, then we have

Algorithm 6.1: B-MaxConsistency$(Y, \mathbf{r})$

```
Max ← SingleValueMaxConsistencyB(Y, r)
NoBefore ← NoBeforeMaxConsistency(Y, r)
if Max < NoBefore
  then Max ← NoBefore
Endpoints ← π_B(r) ∪ π_E(r)
Pairs ← {(b, e) ; b, e ∈ Endpoints ∧ b ⩽ e}
for each (b, e) ∈ Pairs
  do
    MaxBeforeValue ← MaxBefore(Y, r, b, e)
    DuringValue ← DuringOrEqual(r, b, e)
    if Max < MaxBeforeValue + DuringValue
    then
      Max ← MaxBeforeValue + DuringValue
return (Max)
```

procedure SingleValueMaxConsistencyB$(Y, \mathbf{r})$

```
r̄ ← r lexicographically ordered on Y
Max ← 0
Current ← 1
for i = 2, ..., |r̄|
  do
    if t_i[Y] ≠ t_{i−1}[Y]
      then
        if Current > Max
          then Max ← Current
        Current ← 1
      else Current ← Current + 1
if Current > Max
  then Max ← Current
return (Max)
```

procedure MaxBefore$(Y, \mathbf{r}, b, e)$

```
Max ← 0
Yvalues ← π_Y(r)
for each v ∈ Yvalues
  do
    Current ← 0
    for each t ∈ r
      do
        if ( t[Y] = v ∧ t[B] ⩽ e
             ∧t[E] ⩾ b )
        then
          Current ← Current + 1
    if Max < Current
      then Max ← Current
return (Max)
```

procedure DuringOrEqual$(\mathbf{r}, b, e)$

```
Value ← 0
for each t ∈ r
  do
    if t[B] ⩽ b ∧ e ⩽ t[E]
      then Value ← Value + 1
return (Value)
```

procedure NoBeforeMaxConsistency$(Y, \mathbf{r})$

```
Endpoints ← π_B(r) ∪ π_E(r)
Max ← 0
for each e ∈ Endpoints
  do
    Current ← 0
    for each t ∈ r
      do
        if t[B] ⩽ e ⩽ t[E]
        then
          Current ← Current + 1
    if Max < Count
      then Max ← Current
return (Max)
```

$I(t)$ B $I(t'')$ and thus $t''[Y] \neq t[Y]$, which again contradicts the *B-Consistency* assumption.

The F case is completely symmetric and thus omitted;

- $I(t)$ D $I(t')$. As for the S case we have that there exists a tuple t'' with either $I(t'')$ B $I(t')$ or $I(t')$ B $I(t'')$ and from *B-Consistency* we know that $t''[Y] = t'[Y]$ holds. Since $I(t')$ contains $I(t)$, we have that if $I(t'')$ B $I(t')$ then $I(t'')$ B $I(t)$ and if $I(t')$ B $I(t'')$ then $I(t)$ B $I(t'')$. Since $t''[Y] \neq t[Y]$, we have a contradiction with respect to the *B-Consistency* assumption;
- $I(t)$ M $I(t')$. Since both t and t' belong to $\mathbf{r}_B$, there exists a pair of tuples t'' and t''', for which either $I(t'')$ B $I(t)$ or $I(t)$ B $I(t'')$ and either $I(t''')$ B $I(t')$ or $I(t')$ B $I(t''')$ hold. From *B-Consistency* we have that both $t''[Y] = t[Y]$ and $t'''[Y] = t'[Y]$ hold. Two main cases may arise: i) $I(t'')$ B $I(t)$ or $I(t')$ B $I(t''')$; ii) $I(t)$ B $I(t'')$ and $I(t''')$ B $I(t')$. Suppose that $I(t'')$ B $I(t)$ (the case where $I(t')$ B $I(t''')$ is symmetric and thus omitted). The case is depicted in Figure 3 M-case i). In such a case we have $I(t'')$ B $I(t')$ and $t''[Y] \neq t'[Y]$,

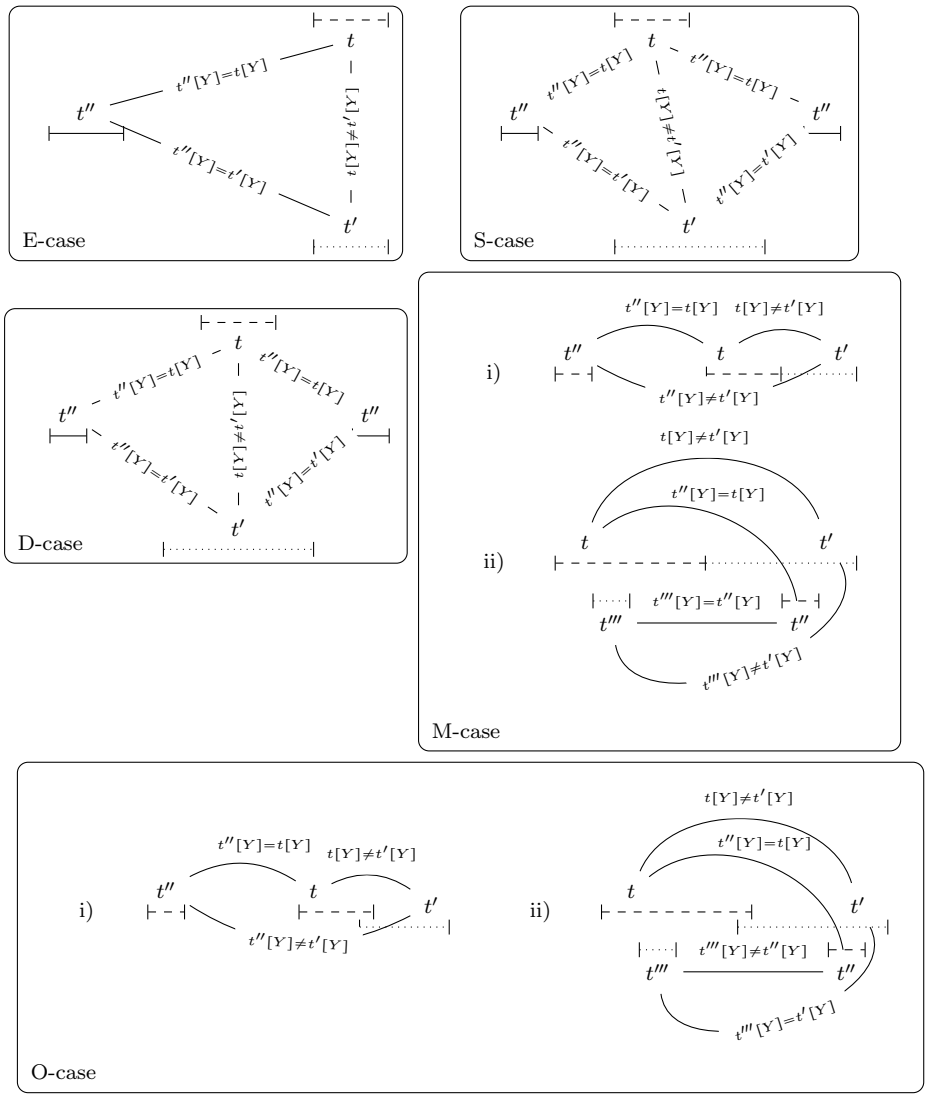


Fig. 3 A graphical account of the five cases of Lemma 1.

which contradicts the *B-Consistency* assumption. Let us consider now the case where $I(t)$ B $I(t'')$ and $I(t''')$ B $I(t')$ (depicted in Figure 3 M-case ii)). Since $I(t)$ M $I(t')$, we have $I(t''')$ B $I(t'')$. Thus, since $t'''[Y] \neq t''[Y]$, the *B-Consistency* assumption is contradicted;

– $I(t)$ O $I(t')$ (very similar to the M case). Since both t and t' belong to $\mathbf{r}_B$, there exists a pair of tuples t'' and t''' for which either $I(t'')$ B $I(t)$ or $I(t)$ B $I(t'')$ and either $I(t''')$ B $I(t')$ or $I(t')$ B $I(t''')$ hold. From *B-Consistency* we have both $t''[Y] = t[Y]$ and $t'''[Y] = t'[Y]$. Two cases may arise: i) $I(t'')$ B $I(t)$ or $I(t')$ B $I(t''')$; ii) $I(t)$ B $I(t'')$ and $I(t''')$ B $I(t')$. Suppose that $I(t'')$ B $I(t)$ (the case where $I(t')$ B $I(t''')$ is symmetric and thus omitted). The case is depicted

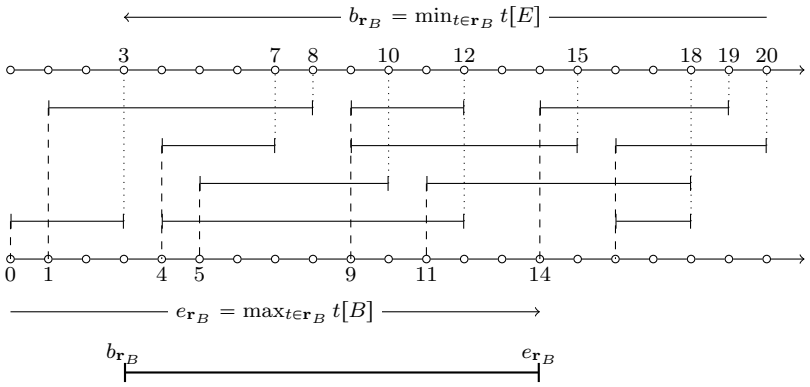


Fig. 4 An example of how interval $[b_{\mathbf{r}_B}, e_{\mathbf{r}_B}]$ is drawn from a set $\mathbf{r}_B$ (for the sake of readability the flow of time is reported on the top and on the bottom of the picture).

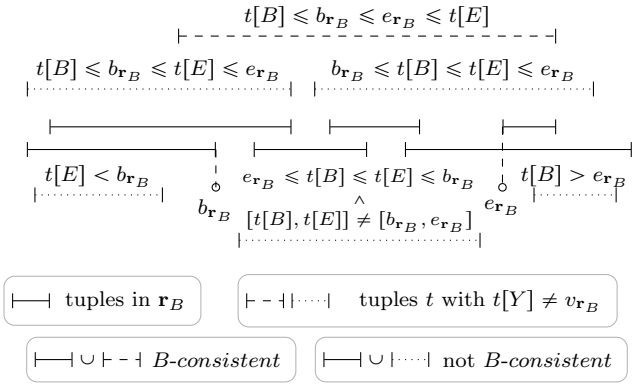


Fig. 5 A graphical account of the constraints on intervals in a *B-consistent* instance.

in Figure 3 *O*-case i). In such a case we have $I(t'')$ B $I(t')$ and $t''[Y] \neq t'[Y]$, which contradicts the *B-Consistency* assumption. Let us consider now the case where $I(t)$ B $I(t'')$ and $I(t''')$ B $I(t')$ (depicted in Figure 3 *O*-case ii)). Since $I(t)$ O $I(t')$, we have $I(t''')$ B $I(t'')$. Moreover, since $t'''[Y] \neq t''[Y]$, the *B-Consistency* assumption is contradicted.

Since we have reached a contradiction in each case, we can consider the left to right direction proved. For the right to left direction, suppose by contradiction that for every pair of tuples $t, t' \in \mathbf{r}_B$ we have $t[Y] = t'[Y]$ and $\mathbf{r}$ is not *B-Consistent*. This means that there exists two tuples $t, t' \in \mathbf{r}$ with either $I(t)$ B $I(t')$ or $I(t')$ B $I(t)$ and $t[Y] \neq t'[Y]$. By the definition of $\mathbf{r}_B$, we have that $t, t' \in \mathbf{r}_B$ and thus $t[Y] = t'[Y]$ (contradiction).

Given a *B-consistent* instance $\mathbf{r}$ w.r.t. Y, if $\mathbf{r}_B \neq \varnothing$, we define $v_{\mathbf{r}_B}$ as the value $v_{\mathbf{r}_B} = t[Y]$ for some $t \in \mathbf{r}_B$ (this definition is correct since such a value is unique for Lemma 1). Moreover, we define $b_{\mathbf{r}_B} = \min_{t \in \mathbf{r}_B} t[E]$ and $e_{\mathbf{r}_B} = \max_{t \in \mathbf{r}_B} t[B]$. In Figure 4 we provide a graphical account of interval $[b_{\mathbf{r}_B}, e_{\mathbf{r}_B}]$. Lemma 1 is used to

proved the following result which basically constrains the arrangement of intervals of *B-consistent* instances.

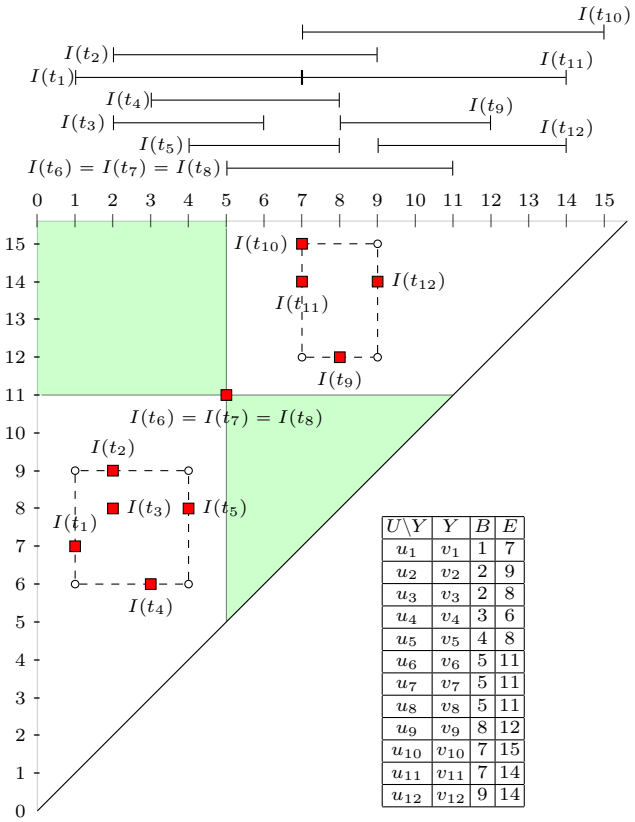
Lemma 2 *For every temporal relation schema $\mathcal{R} = R(U, B, E)$ and every instance $\mathbf{r}$ of $\mathcal{R}$, if $\mathbf{r}_B$ is monochromatic and $\mathbf{r}_B \neq \emptyset$, then for every tuple $t \in \mathbf{r}$ with $t[Y] \neq v_{\mathbf{r}_B}$ we have $t[B] \leqslant b_{\mathbf{r}_B} \leqslant e_{\mathbf{r}_B} \leqslant t[E]$.*

Proof Suppose by contradiction that there exists a tuple $t \in \mathbf{r}$ with $t[Y] \neq v_{\mathbf{r}_B}$ and either $t[B] > b_{\mathbf{r}_B}$ or $e_{\mathbf{r}_B} > t[E]$. Since $b_{\mathbf{r}_B} = \min_{t \in \mathbf{r}_B} t[E]$ by definition, we have that there exists a tuple $t' \in \mathbf{r}_B$ with $t'[E] = b_{\mathbf{r}_B}$. Thus we have $I(t')\ B\ I(t)$, which means that $\mathbf{r}$ is not *B-Consistent*. This is a contradiction since $\mathbf{r}_B$ is monochromatic and, for Lemma 1, we have that $\mathbf{r}$ is *B-Consistent*. Analogously, we have that $e_{\mathbf{r}_B} = \max_{t \in \mathbf{r}_B} t[B]$ by definition and thus there exists a tuple $t' \in \mathbf{r}_B$ with $t'[B] = e_{\mathbf{r}_B}$. We have $I(t)\ B\ I(t')$, which means that $\mathbf{r}$ is not *B-Consistent*. This is a contradiction, since $\mathbf{r}_B$ is monochromatic and, for Lemma 1, we have that $\mathbf{r}$ is *B-Consistent*.

Lemma 2 gives the following property on *B-consistent* instances w.r.t. Y. If $\mathbf{r}_B \neq \emptyset$, then every tuple $t \in \mathbf{r}$ with $t[Y] \neq v_{\mathbf{r}_B}$ must satisfy either $[b_{\mathbf{r}_B}, e_{\mathbf{r}_B}]\ D\ I(t)$, or $[b_{\mathbf{r}_B}, e_{\mathbf{r}_B}]\ S\ I(t)$, or $[b_{\mathbf{r}_B}, e_{\mathbf{r}_B}]\ F\ I(t)$, or $I(t) = [b_{\mathbf{r}_B}, e_{\mathbf{r}_B}]$. Figure 5 provides a graphical account of a set of intervals associated with tuples that either satisfy (dashed intervals) or violate (dotted intervals) conditions of Lemma 2.

Algorithm 6.1 makes use of Lemma 2 to solve problem *B-MaxConsistent* in polynomial time. It takes as input an instance $\mathbf{r}$ and an atemporal attribute Y, and selects the maximum value among three values, which represent an equal number of cases. The first value is calculated by procedure *SingleValueMaxConsistency*, which basically returns the maximum cardinality among the subsets of $\mathbf{r}$ that share the same value for attribute Y. The second value is calculated by procedure *NoBeforeMaxConsistency*: it returns the maximum cardinality among the subsets of $\mathbf{r}$ that do not feature two or more tuples t, t' with $I(t)\ B\ I(t')$. Then, the main procedure considers every pair of values (b, e) with $b \leqslant e$ and $b, e \in \pi_B(\mathbf{r}) \cup \pi_E(\mathbf{r})$. First, by means of procedure *MaxBefore*, it computes the maximum cardinality of a subset $\mathbf{r}' \in \mathbf{r}$, such that, for each tuple $t \in \mathbf{r}'$, it holds $b \leqslant t[B] \leqslant t[E] \leqslant e$ and for each pair of tuples $t, t' \in \mathbf{r}'$ it holds $t[Y] = t'[Y]$. Second, by means of procedure *DuringOrEqual*, it computes how many tuples t, regardless of value $t[Y]$, satisfy $t[B] \leqslant b \leqslant e \leqslant t[E]$. The sum of these two values is chosen and its value is compared with the values obtained from the previous two cases: then, the maximum over such values is returned. It is easy to see that the complexity function of the loop in the main procedure asymptotically dominates the complexities of the two procedures *MaxBefore* and *DuringOrEqual*. In particular, the main procedure iterates the function *MaxBefore*, which is quadratic, a quadratic number of times (the worst case for the cardinality of the set *Pairs*). The resulting (data) complexity of procedure *B-MaxConsistency* is $\mathcal{O}(|\mathbf{r}|^4)$. This analysis basically proves the following result.

Theorem 5 *For every temporal relation schema $\mathcal{R} = R(U, B, E)$, every attribute $Y \in U$, and every instance $\mathbf{r}$ of $\mathcal{R}$, the complexity of problem B-MaxConsistent$(Y, \mathbf{r})$ is $\mathcal{O}(|\mathbf{r}|^4)$.*



$U \backslash Y$	Y	B	E
u_1	v_1	1	7
u_2	v_2	2	9
u_3	v_3	2	8
u_4	v_4	3	6
u_5	v_5	4	8
u_6	v_6	5	11
u_7	v_7	5	11
u_8	v_8	5	11
u_9	v_9	8	12
u_{10}	v_{10}	7	15
u_{11}	v_{11}	7	14
u_{12}	v_{12}	9	14

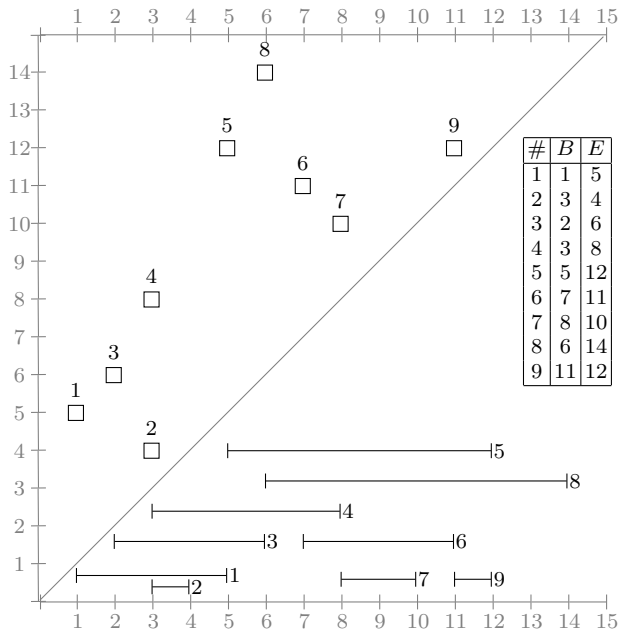
Fig. 6 A spatial representation of (*D-clusters* of) intervals.

7 Maximal Consistency for the D case

In this section we deal with the complexity of problem *D-MaxConsistency*. In order to make the explanation easier, let us introduce a spatial representation of intervals called *compass structure* [30]. Basically intervals $[b, e]$ are interpreted as points in the euclidean plane and, since $b \leqslant e$, all points/intervals belong to the half plane $y \geqslant x$. Figure 6 depicts an example of compass structure. It is easy to see that relations between intervals can be directly translated into spatial relations between their respective representations as points in the compass structure. For our purposes we need only to describe how interval relation D is mapped into the compass structure.

As an example, let us consider tuple t_6 in Figure 6. We have that, if a tuple t features an interval $I(t)$ such that $I(t)\ D\ I(t_6)$, the corresponding point of t must belong to the shaded triangle below $I(t_6)$ in the compass structure. On the other hand, if a tuple t features an interval $I(t)$ such that $I(t_6)\ D\ I(t)$, the corresponding point of t must belong to the shaded rectangle above $I(t_6)$ in the compass structure. Given a temporal relation schema $\mathcal{R} = R(U, B, E)$ and an instance $\mathbf{r}$ of it, we define the binary relation $\xrightarrow{D}$: $\mathbf{r} \times \mathbf{r}$ among tuples of $\mathbf{r}$. For every pair of tuples $t, t' \in \mathbf{r}$ we

x	$\overset{D*}{\rightarrow} y$
1	2
2	3
1	3
6	7
6	8
7	8
5	6
5	7
5	8
6	9
7	9
8	9
5	9
2	1
3	2
3	1
7	6
8	6
8	7
6	5
7	5
8	5
9	6
9	7
9	8
9	5



#	B	E
1	1	5
2	3	4
3	2	6
4	3	8
5	5	12
6	7	11
7	8	10
8	6	14
9	11	12

Fig. 7 An example of relation $\overset{D*}{\rightarrow}$.

have $t \overset{D}{\rightarrow} t'$ if and only if either $I(t)$ D $I(t')$ or $I(t')$ D $I(t)$. It is easy to see that relation $\overset{D}{\rightarrow}$ is symmetric.

Let $\overset{D*}{\rightarrow}$ be the transitive and reflexive closure of relation $\overset{D}{\rightarrow}$. We have that $\overset{D*}{\rightarrow}$ is an equivalence relation over $\mathbf{r}$. Figure 7 depicts an example of relation $\overset{D*}{\rightarrow}$. It is easy to observe that we may have $I(t) \overset{D*}{\rightarrow} I(t')$, when neither $I(t)$ D $I(t')$ nor $I(t')$ D $I(t)$ hold. For instance, it is the case of tuples t_1 and t_3 in Figure 7. In such example, the existence of t_2 determines that $t_1 \overset{D*}{\rightarrow} t_3$, since $t_1 \overset{D}{\rightarrow} t_2 \overset{D}{\rightarrow} t_3$. Let us now consider set $\{t_6, t_7, t_8\}$, containing only tuples which share the same interval. Clearly, we have that $t_i \overset{D}{\rightarrow} t_j$ does not hold for every $i, j \in \{6, 7, 8\}$. As a matter of fact, $t_6 \overset{D*}{\rightarrow} t_7$, $t_7 \overset{D*}{\rightarrow} t_8$, and $t_8 \overset{D*}{\rightarrow} t_6$ hold thanks to the reflexivity of relation $\overset{D*}{\rightarrow}$.

A subset $\mathbf{r}' \subseteq \mathbf{r}$ is a *D-closed set* of $\mathbf{r}$, if for every couple of tuples $t, t' \in \mathbf{r}'$, we have $t \overset{D*}{\rightarrow} t'$. Given a temporal relation schema $\mathcal{R} = R(U, B, E)$ and an instance $\mathbf{r}$ of it, a subset $\mathbf{r}^c \subseteq \mathbf{r}$ is a *D-cluster* of $\mathbf{r}$ if it is a maximal non-empty *D-closed set* of $\mathbf{r}$. In other words, a *D-cluster* of $\mathbf{r}$ is an equivalence class with respect to relation $\overset{D*}{\rightarrow}$. Given an instance $\mathbf{r}$, let $\mathcal{C}_\mathbf{r} = \{\mathbf{r}' \ : \ \mathbf{r}' \subseteq \mathbf{r}$ and $\mathbf{r}'$ is a *D-cluster*$\}$ be the set of all the *D-clusters* in $\mathbf{r}$. For example, in Figure 6 we have three clusters $\mathbf{r}^c_1 = \{t_1, \ldots, t_5\}$, $\mathbf{r}^c_2 = \{t_6, t_7, t_8\}$, and $\mathbf{r}^c_3 = \{t_9, \ldots, t_{12}\}$, respectively. We would like to point out the advantage of the compass representation (Figure 6 below) with respect to the interval one (Figure 6 above). In fact, it turns out difficult to see immediately *D-clusters* drawing them in the standard way, because it may happen that many intervals belonging to different *D-clusters* overlap. If we look

at the compass representation, things appear clearer. In compass structures, *D-clusters* are represented either by boxes or by isolated points, which are pairwise disjoint, as stated more formally by the following result.

Lemma 3 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$ and any instance $\mathbf{r}$ of it, the set $\mathcal{C}_\mathbf{r}$ is a partition of $\mathbf{r}$.*

Proof Directly follows from the fact that $\overset{D*}{\rightarrow}$ is an equivalence relation over $\mathbf{r}$ and any *D-cluster* represents an equivalence class of $\overset{D*}{\rightarrow}$.

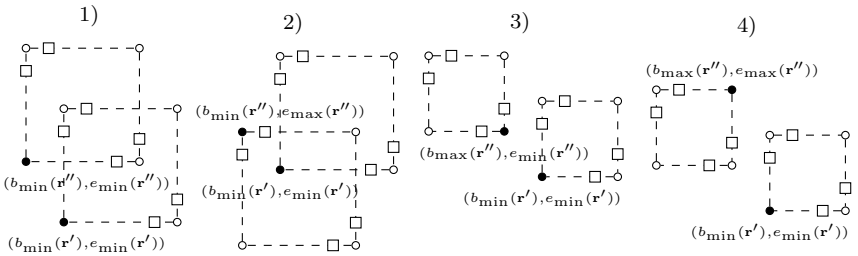
Given a non empty *D-closed* set $\mathbf{r}'$ of $\mathbf{r}$, we define the following four elements drawn from $\pi_B(\mathbf{r}) \cup \pi_E(\mathbf{r})$: $b_{\min}(\mathbf{r}^c) = \min_{t \in \mathbf{r}^c} t[B]$, $e_{\min}(\mathbf{r}^c) = \min_{t \in \mathbf{r}^c} t[E]$, $b_{\max}(\mathbf{r}^c) = \max_{t \in \mathbf{r}^c} t[B]$, and $e_{\max}(\mathbf{r}^c) = \max_{t \in \mathbf{r}^c} t[E]$.
Since *D-clusters* are specializations of *D-closed* sets, the definitions of $b_{\min}(\mathbf{r}^c)$, $e_{\min}(\mathbf{r}^c)$, $b_{\max}(\mathbf{r}^c)$ and $e_{\max}(\mathbf{r}^c)$ hold for them as well. Once more, it is useful to look at the compass representation of intervals to have a better intuition of these four points. Basically, points $(b_{\min}(\mathbf{r}^c),\ e_{\min}(\mathbf{r}^c))$ and $(b_{\max}(\mathbf{r}^c),\ e_{\max}(\mathbf{r}^c))$ are the low left corner and the top right corner of the box representing *D-cluster* $\mathbf{r}^c$, respectively. In Figure 6 we have $(b_{\min}(\mathbf{r}_1^c), e_{\min}(\mathbf{r}_1^c)) = (1, 6)$, $(b_{\max}(\mathbf{r}_1^c), e_{\max}(\mathbf{r}_1^c)) = (4, 9)$, $(b_{\min}(\mathbf{r}_2^c),\ e_{\min}(\mathbf{r}_2^c)) = (b_{\max}(\mathbf{r}_2^c),\ e_{\max}(\mathbf{r}_2^c)) = (5, 11)$ (clearly for isolated points the two corners coincide), $(b_{\min}(\mathbf{r}_3^c),\ e_{\min}(\mathbf{r}_3^c)) = (7, 12)$, and $(b_{\max}(\mathbf{r}_3^c), e_{\max}(\mathbf{r}_3^c)) = (9, 15)$. The following result provides a sufficient and necessary property for determining whether two non-empty *D-closed* sets belong or not to the same *D-cluster*.

Lemma 4 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$ and any instance $\mathbf{r}$ of it, given two non-empty D-closed sets $\mathbf{r}'$ and $\mathbf{r}''$, if there exist two intervals $[b', e']$ and $[b'', e'']$ with $b' \in \{b_{\min}(\mathbf{r}'), b_{\max}(\mathbf{r}')\}$, $e' \in \{e_{\min}(\mathbf{r}'), e_{\max}(\mathbf{r}')\}$, $b'' \in \{b_{\min}(\mathbf{r}''), b_{\max}(\mathbf{r}'')\}$ and $e'' \in \{e_{\min}(\mathbf{r}''), e_{\max}(\mathbf{r}'')\}$, such that $[b', e']\ D\ [b'', e'']$ or $[b'', e'']\ D\ [b', e']$, then there exists a cluster $\mathbf{r}^c$ of $\mathbf{r}$ with $(\mathbf{r}' \cup \mathbf{r}'') \subseteq \mathbf{r}^c$.*

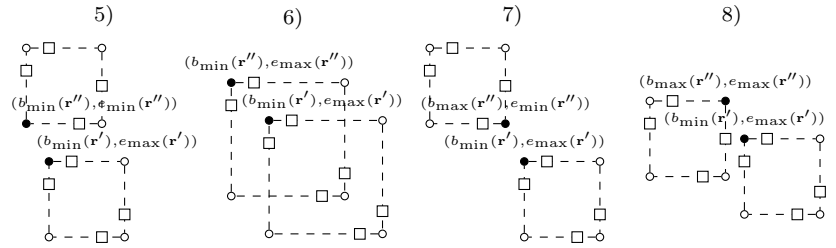
Proof We prove one among the 16 possible cases (Figure 8). The remaining 15 cases may be proved in an analogous way. In order to prove the claim, it suffices to find two tuples $t \in \mathbf{r}'$ and $t' \in \mathbf{r}''$ such that $t \overset{D*}{\rightarrow} t'$. Then, the $\overset{D*}{\rightarrow}$-maximality property of *D-cluster* will do the rest. Let us consider the case where $[b'', e''] = [b_{\max}(\mathbf{r}''), e_{\min}(\mathbf{r}'')]$ and $[b', e'] = [b_{\min}(\mathbf{r}'), e_{\min}(\mathbf{r}')]$ (case 3 in Figure 8). The other cases may be dealt with similarly. By hypothesis, we have either $[b', e']\ D\ [b'', e'']$ or $[b'', e'']\ D\ [b', e']$. Suppose that $[b', e']\ D\ [b'', e'']$ (the other case is completely symmetric). From the definitions of $b_{\max}(\mathbf{r})$ and $b_{\min}(\mathbf{r})$ we have that there exist two tuples $t \in \mathbf{r}''$ and $t' \in \mathbf{r}'$ for which $I(t) = [b_t, e_{min}(\mathbf{r}'')]$ for some $b_t \leqslant b_{\max}(\mathbf{r}'')$, $I(t') = [b_{t'}, e_{\min}(\mathbf{r}')]$ for some $b_{t'} \geqslant b_{\min}(\mathbf{r}')$. From $[b', e']\ D\ [b'', e'']$ we have $b_t \leqslant b_{\max}(\mathbf{r}'') < b_{\min}(\mathbf{r}') \leqslant b_{t'} \leqslant e_{\min}(\mathbf{r}') < e_{\min}(\mathbf{r}'')$ and thus $I(t')\ D\ I(t)$.

A *D-cluster* $\mathbf{r}^c$ of $\mathbf{r}$ is *interval-uniform* iff $(b_{\max}(\mathbf{r}^c), e_{\max}(\ \mathbf{r}^c)) = (b_{\min}(\mathbf{r}^c), e_{\min}(\ \mathbf{r}^c))$. It is easy to see that in the compass representation interval-uniform *D-clusters* are represented only by isolated points. In the example of Figure 6, we have that cluster $\mathbf{r}_2^c = \{t_6, t_7, t_8\}$ is interval-uniform. By means of the four elements $b_{\min}$, $e_{\min}$, $b_{\max}$, and $e_{\max}$ of $\mathbb{O}$, we give a relation $\leqslant_c$ over $\mathcal{C}_\mathbf{r}$. For every pair $\mathbf{r}_1^c, \mathbf{r}_2^c \in \mathcal{C}_\mathbf{r}$, we have that $\mathbf{r}_1^c <_c \mathbf{r}_2^c$ if and only if $\mathbf{r}_1^c \neq \mathbf{r}_2^c$, $b_{\max}(\mathbf{r}_1^c) \leqslant b_{\min}(\mathbf{r}_2^c)$ and $e_{\max}(\mathbf{r}_1^c) \leqslant e_{\min}(\mathbf{r}_2^c)$.

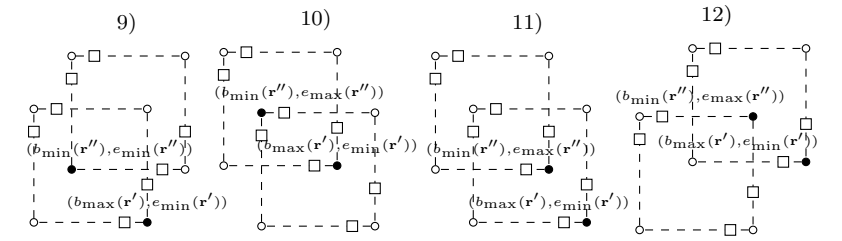
1	$[b_{\min}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$	3	$[b_{\min}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$
2	$[b_{\min}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$	4	$[b_{\min}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$



5	$[b_{\min}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$	7	$[b_{\min}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$
6	$[b_{\min}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$	8	$[b_{\min}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$



9	$[b_{\max}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$	11	$[b_{\max}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$
10	$[b_{\max}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$	12	$[b_{\max}(\mathbf{r}'),e_{\min}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$



13	$[b_{\max}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$	15	$[b_{\max}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\min}(\mathbf{r}'')]$
14	$[b_{\max}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\min}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$	16	$[b_{\max}(\mathbf{r}'),e_{\max}(\mathbf{r}')]\ D\ [b_{\max}(\mathbf{r}''),e_{\max}(\mathbf{r}'')]$

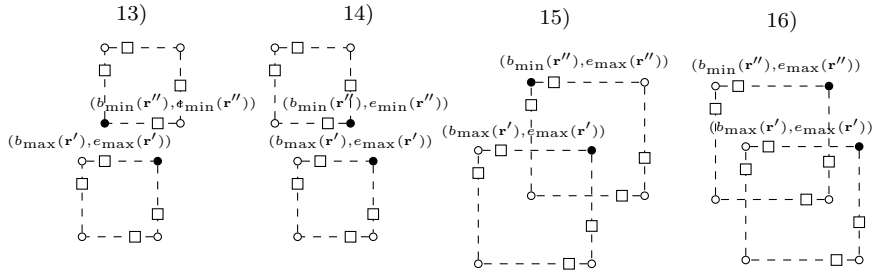


Fig. 8 The sixteen cases of Lemma 3.

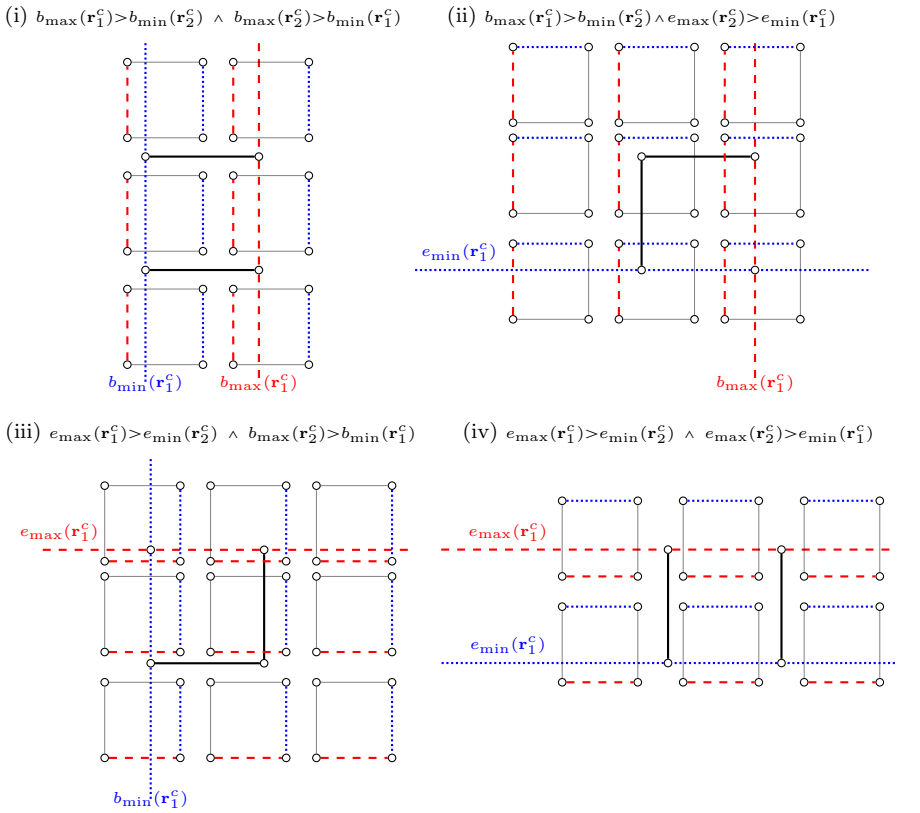


Fig. 9 An example of the four cases of Lemma 5.

Lemma 5 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$ and any instance $\mathbf{r}$ of it, relation $<_c$ is a total order relation over $\mathcal{C}_\mathbf{r}$.*

Proof Suppose by contradiction that there exist two clusters $\mathbf{r}_1^c, \mathbf{r}_2^c \in \mathcal{C}_\mathbf{r}$ with $\mathbf{r}_1^c \neq \mathbf{r}_2^c$, such that neither $\mathbf{r}_1^c <_c \mathbf{r}_2^c$ nor $\mathbf{r}_2^c <_c \mathbf{r}_1^c$ hold. We have four cases to consider: (i) $b_{\max}(\mathbf{r}_1^c) > b_{\min}(\mathbf{r}_2^c)$ and $b_{\max}(\mathbf{r}_2^c) > b_{\min}(\mathbf{r}_1^c)$; (ii) $b_{\max}(\mathbf{r}_1^c) > b_{\min}(\mathbf{r}_2^c)$ and $e_{\max}(\mathbf{r}_2^c) > e_{\min}(\mathbf{r}_1^c)$; (iii) $e_{\max}(\mathbf{r}_1^c) > e_{\min}(\mathbf{r}_2^c)$ and $b_{\max}(\mathbf{r}_2^c) > b_{\min}(\mathbf{r}_1^c)$; (iv) $e_{\max}(\mathbf{r}_1^c) > e_{\min}(\mathbf{r}_2^c)$ and $e_{\max}(\mathbf{r}_2^c) > e_{\min}(\mathbf{r}_1^c)$. Each case is depicted in Figure 9: cluster $\mathbf{r}_1^c$ is denoted by thick black lines and has fixed position. Cluster $\mathbf{r}_2^c$ is drawn in dashed, dotted and thinner lines and may assume any size or position under the constraint that both their dashed/dotted lines maintain the same relative position with respect to the long dashed/dotted line departing from $\mathbf{r}_1^c$. As an example, consider case (i) in Figure 9. We can move and resize cluster $\mathbf{r}_2^c$ as we want, as long as the dashed line stays on the left of the long dashed line labelled with $b_{\max}(\mathbf{r}_1^c)$ line, and the dotted line stays on the right of the long dotted line labelled with $b_{\min}(\mathbf{r}_1^c)$.

We shall show only the contradiction for case (i), as for the remaining cases contradiction may be achieved in a similar way. If $b_{\max}(\mathbf{r}_1^c) > b_{\min}(\mathbf{r}_2^c)$ and $b_{\max}(\mathbf{r}_2^c) > b_{\min}(\mathbf{r}_1^c)$ hold, then we may have two sub-cases.

i.a) $b_{\min}(\mathbf{r}^c_1) < b_{\max}(\mathbf{r}^c_2) < b_{\max}(\mathbf{r}^c_1)$. If $e_{\min}(\mathbf{r}^c_2) < e_{\max}(\mathbf{r}^c_1)$, we have $[b_{\max}(\mathbf{r}^c_2), e_{\min}(\mathbf{r}^c_2)]$ $D[b_{\max}(\mathbf{r}^c_1), e_{\max}(\ \mathbf{r}^c_1)]$; otherwise, we have $[b_{\max}(\mathbf{r}^c_1), e_{\min}(\ \mathbf{r}^c_1)]$ D $[b_{\max}(\mathbf{r}^c_2), e_{\min}(\ \mathbf{r}^c_2)]$. In both cases Lemma 4 is applicable and we have that $\mathbf{r}^c_1 = \mathbf{r}^c_2$ (contradiction);

i.b) $b_{\min}(\mathbf{r}^c_1) < b_{\max}(\mathbf{r}^c_1) \leqslant b_{\max}(\mathbf{r}^c_2)$. Since $b_{\max}(\mathbf{r}^c_1) > b_{\min}(\mathbf{r}^c_2)$, we may have $b_{\min}(\mathbf{r}^c_1) \leqslant b_{\min}(\mathbf{r}^c_2)$ and thus either $[b_{\max}(\mathbf{r}_2), e_{\min}(\mathbf{r}_2)]$ D $[b_{\min}(\mathbf{r}_1), e_{\max}(\mathbf{r}_1)]$ or $[b_{\max}(\mathbf{r}_1), e_{\min}(\mathbf{r}_1)]$ D $[b_{\min}(\ \mathbf{r}_2), e_{\max}(\mathbf{r}_2)]$ holds. On the other hand, we may have $b_{\min}(\mathbf{r}^c_2) < b_{\min}(\mathbf{r}^c_1)$ and thus either $[b_{\min}(\mathbf{r}_1), e_{\min}(\mathbf{r}_1)]$ D $[b_{\min}(\ \mathbf{r}_2), e_{\max}(\mathbf{r}_2)]$ or $[b_{\max}(\mathbf{r}_2), e_{\min}(\mathbf{r}_2)]$ D $[b_{\min}(\ \mathbf{r}_1), e_{\max}(\mathbf{r}_1)]$ hold. In both cases Lemma 4 is applicable and we have that $\mathbf{r}^c_1 = \mathbf{r}^c_2$ (contradiction).

In Figure 6 we have that $\mathbf{r}^c_1 <_c \mathbf{r}^c_2 <_c \mathbf{r}^c_3$. The following result connects *D-consistent* instances $\mathbf{r}$ with a property of clusters in $\mathcal{C}_\mathbf{r}$.

Theorem 6 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$, any attribute $Y \in U$ and any instance $\mathbf{r}$ of $\mathcal{R}$, $\mathbf{r}$ is D-consistent w.r.t. Y if and only if for each $\mathbf{r}^c \in \mathcal{C}_\mathbf{r}$, either $\mathbf{r}^c$ is interval-uniform or for all pairs of tuples $t, t' \in \mathbf{r}^c$ it holds $t[Y] = t'[Y]$.*

Proof We prove the left-to-right direction by contradiction. We have that there exists a cluster $\mathbf{r}^c$ which is not interval uniform and there exists two tuples $t, t' \in \mathbf{r}^c$, for which $t[Y] \neq t'[Y]$. Since both t and t' belong to the same cluster, we have $t \xrightarrow{D*} t'$ and we prove by induction on the length of the chain $t \xrightarrow{D} t_1 \xrightarrow{D} \ldots \xrightarrow{D} t_n \xrightarrow{D} t'$ that a contradiction always arises. If $n = 0$ we have that $I(t) = I(t')$. Since $\mathbf{r}^c$ is not interval-uniform, there exists a tuple t'', for which either $I(t'')$ D $I(t)$ or $I(t)$ D $I(t'')$ hold. Since $\mathbf{r}$ is *D-consistent*, we have $t''[Y] = t[Y]$. Since either $I(t'')$ D $I(t')$ or $I(t')$ D $I(t'')$ hold and $t''[Y] \neq t'[Y]$, it contradicts the *D-consistency* hypothesis. If $n > 0$, let us consider $t \xrightarrow{D} t_1$. We may have either $I(t_1)$ D $I(t)$ or $I(t)$ D $I(t_1)$. In both cases the *D-consistency* condition requires $t[Y] = t_1[Y]$. Thus, we have a chain $t_1 \xrightarrow{D} \ldots \xrightarrow{D} t_n \xrightarrow{D} t'$ with $t_1[Y] \neq t'[Y]$ and, by inductive hypothesis, a contradiction arises for chain of length less than n. The right-to-left direction is straightforward. Let us take any pair of tuples $t, t' \in \mathbf{r}$ such that either $I(t)$ D $I(t')$ or $I(t')$ D $I(t)$. We have that there exists a *D-cluster* $\mathbf{r}^c$, such that $t, t' \in \mathbf{r}^c$ and $\mathbf{r}^c$ is not interval-uniform. Thus, by hypothesis we have $t[Y] = t'[Y]$. We have that $\mathbf{r}$ is *D-consistent* with respect to Y.

Theorem 6 allows us to define, only for all non interval-uniform clusters $\mathbf{r}^c$ in a *D-consistent* instance $\mathbf{r}$, the value $v_Y(\mathbf{r}^c)$, such that for every tuple in $\mathbf{r}^c$ we have $t[Y] = v_Y(\mathbf{r}^c)$. Value $v_Y(\mathbf{r}^c)$ is well defined if we assume that instance $\mathbf{r}$ is *D-consistent* and cluster $\mathbf{r}^c$ is not interval-uniform. Let us consider instance $\mathbf{r}$ represented in Figure 6. Theorem 6 says that $\mathbf{r}$ is *D-consistent* w.r.t. Y if and only if $v_1 = \ldots = v_5$ and $v_9 = \ldots = v_{12}$. It is worth to notice that D-consistency does not impose any constraint on cluster $\mathbf{r}^c_2$, since it is interval-uniform. It implies that there exists a unique interval associated with tuples in $\mathbf{r}^c_2$ and we have that their value for attribute Y may differ.

Given a *D-consistent* instance $\mathbf{r}$ of some temporal relation schema $\mathcal{R} = R(U, B, E)$ and a tuple t on attributes U, B, and E, we say that t is *covered* in $\mathbf{r}$ with respect to Y if there exists a cluster $\mathbf{r}^c \subseteq \mathbf{r}$, for which one of the following conditions holds: (i) $\mathbf{r}^c$ is interval-uniform, $b_{min}(\mathbf{r}^c) = t[B]$ and $e_{min}(\mathbf{r}^c) = t[E]$; (ii) $\mathbf{r}^c$ is not interval-uniform, $b_{min}(\mathbf{r}^c) \leqslant t[B] \leqslant b_{max}(\mathbf{r}^c)$, $e_{min}(\mathbf{r}^c) \leqslant t[E] \leqslant e_{max}(\mathbf{r}^c)$, and

$v_Y(\mathbf{r}^c) = t[Y]$. In the example in Figure 6 we have that tuple t_3 is covered w.r.t. Y by every instance containing $\mathbf{r}' = \{t_1, t_2, t_4\}$ assuming $v_1 = \ldots = v_4$.

Lemma 6 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$, any (singleton) subset $Y \subseteq U$, any D-consistent instance $\mathbf{r}$ of $\mathcal{R}$ w.r.t. Y, and any tuple t on attributes U, B, and E, we have that if t is covered in $\mathbf{r}$ w.r.t. Y then $\mathbf{r}' = \mathbf{r} \cup \{t\}$ is D-consistent.*

Proof Since t is covered, we may have two cases. If there exists a *D-cluster* $\mathbf{r}^c$ in $\mathbf{r}$ that is interval-uniform and $b_{min}(\mathbf{r}^c) = t[B]$ and $e_{min}(\mathbf{r}^c) = t[E]$, we have that for each tuple $t' \in \mathbf{r}$ neither $I(t)$ D $I(t')$ nor $I(t')$ D $I(t)$ holds and thus $\mathbf{r} \cup \{t\}$ is still *D-consistent*. If there exists a not interval-uniform *D-cluster* $\mathbf{r}^c$ in $\mathbf{r}$ with $b_{min}(\mathbf{r}^c) \leqslant t[B] \leqslant b_{max}(\mathbf{r}^c)$, $e_{min}(\mathbf{r}^c) \leqslant t[E] \leqslant e_{max}(\mathbf{r}^c)$, and $v_Y(\mathbf{r}^c) = t[Y]$, then from Lemma 4 we have that $\{t\} \cup \mathbf{r}^c$ is a *D-cluster* in $\mathbf{r} \cup \{t\}$. By definition of *D-cluster* we have that for each $t' \in \mathbf{r}$, if either $I(t)$ D $I(t')$ or $I(t')$ D $I(t)$, then $t' \in \mathbf{r}^c$. Since $\mathbf{r}$ is *D-consistent*, we have that $t'[Y] = v_Y(\mathbf{r}^c)$ and, by definition of covered tuple, $t[Y] = v_Y(\mathbf{r}^c)$. Thus, we can conclude that $\mathbf{r} \cup \{t\}$ is *D-Consistent*.

As a consequence of the above lemma we have that *D-consistency* is preserved under the insertion of covered tuples in $\mathbf{r}$.

Corollary 1 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$, any (singleton) subset $Y \subseteq U$, any D-consistent instance $\mathbf{r}$ of $\mathcal{R}$ w.r.t. Y, any set of tuples $\mathbf{r}'$ on attributes U, B and E, which are covered in $\mathbf{r}$ w.r.t. Y, we have that for any tuple t', if it is covered in $\mathbf{r}$ w.r.t. Y, then t' is covered w.r.t. Y in $\mathbf{r}'' = \mathbf{r} \cup \mathbf{r}'$ and $\mathbf{r}''$ is still D-Consistent.*

Basically Lemma 6 says that, given a *D-consistent* instance $\mathbf{r}$, any covered tuple t in $\mathbf{r}$ can be safely added to $\mathbf{r}$, preserving *D-consistency*. In addition, Lemma 1 says that the property of being covered is preserved under the operation of introducing covered tuples in *D-consistent* instances.

Given an instance $\mathbf{r}$ and a *D-closed* set $\overline{\mathbf{r}} \subseteq \mathbf{r}$, we say that $\overline{\mathbf{r}}$ is a *candidate D-cluster* w.r.t. Y for $\mathbf{r}$, if $\overline{\mathbf{r}}$ is *D-consistent* w.r.t. Y and any tuple $t \in \mathbf{r} \backslash \overline{\mathbf{r}}$ is not covered w.r.t. Y in $\overline{\mathbf{r}}$. The following result provides a polynomial bound on the number of *candidate D-clusters*.

Lemma 7 *For any temporal relation schema $\mathcal{R} = R(U, B, E)$, any (singleton) subset $Y \subseteq U$, and any instance $\mathbf{r}$ of $\mathcal{R}$, it holds $|\{\overline{\mathbf{r}} \subseteq \mathbf{r} :$ $\overline{\mathbf{r}}$ is a candidate D-cluster w.r.t. Y $\}| \leqslant |\mathbf{r}|^5)$.*

Proof Suppose by contradiction that there exist two distinct candidate *D-clusters* $\mathbf{r}', \mathbf{r}'' \subseteq \mathbf{r}$, such that $\mathbf{r}' \neq \mathbf{r}''$,$b_{\min}(\mathbf{r}') = b_{\min}(\mathbf{r}'')$, $e_{\min}(\mathbf{r}') = e_{\min}(\mathbf{r}'')$, $b_{\max}(\mathbf{r}') = b_{\max}(\mathbf{r}'')$, $e_{\max}(\mathbf{r}') = e_{\max}(\mathbf{r}'')$ and $v_Y(\mathbf{r}') = v_Y(\mathbf{r}'')$, if both of them are not interval-uniform.
We have that three cases may arise. Suppose that $\mathbf{r}'$ is interval-uniform and $\mathbf{r}''$ is not interval-uniform (the inverse case is symmetric and thus omitted). Then, we have by definition of interval-uniform that $b_{\min}(\mathbf{r}') = b_{\max}(\mathbf{r}')$ and $e_{\min}(\mathbf{r}') = e_{\max}(\mathbf{r}')$ and, since $\mathbf{r}''$ is not interval-uniform, we have $b_{\min}(\mathbf{r}'') \neq b_{\max}(\mathbf{r}'')$ or $e_{\min}(\mathbf{r}'') \neq e_{\max}(\mathbf{r}'')$. This leads immediately to a contradiction. For the second case we have that both $\mathbf{r}'$ and $\mathbf{r}''$ are interval-uniform. Thus, since $\mathbf{r}' \neq \mathbf{r}''$, we can assume that there exists a tuple $t \in \mathbf{r}'' \backslash \mathbf{r}'$ (again the inverse case is symmetric and thus omitted). Since $\mathbf{r}'' \subseteq \mathbf{r}$, we have that $t \in \mathbf{r}$. Thus, t is covered in $\mathbf{r}'$ since $t[B] = b_{\min}(\mathbf{r}') = b_{\max}(\mathbf{r}') = b_{\min}(\mathbf{r}'') = b_{\max}(\mathbf{r}'')$ and $t[E] = e_{\min}(\mathbf{r}') = e_{\max}(\mathbf{r}') =$

$e_{\min}(\mathbf{r}'') = e_{\max}(\mathbf{r}'')$ hold, by definition of interval-uniformity. We have that $\mathbf{r}'$ is not a candidate *D-Cluster* (contradiction).
For the last case, we have that both $\mathbf{r}'$ and $\mathbf{r}''$ are not interval-uniform. Then, since $\mathbf{r}' \neq \mathbf{r}''$, we can assume that there exists a tuple $t \in \mathbf{r}''\backslash\mathbf{r}'$ (again the inverse case is symmetric and thus omitted). Since $t \in \mathbf{r}''$ and by hypothesis we have that $b_{\min}(\mathbf{r}') = b_{\min}(\mathbf{r}'') \leqslant t[B] \leqslant b_{\max}(\mathbf{r}'') = b_{\max}(\mathbf{r}')$, $e_{\min}(\mathbf{r}') = e_{\min}(\mathbf{r}'') \leqslant t[E] \leqslant e_{\max}(\mathbf{r}'') = e_{\max}(\mathbf{r}')$ and $t[Y] = v_Y(\mathbf{r}'') = v_Y(\mathbf{r}')$ implying that t is covered in $\mathbf{r}'$ and thus $\mathbf{r}'$ is not a candidate *D-Cluster* (contradiction). Thus, we have that a candidate *D-Cluster* $\overline{\mathbf{r}}$ may be identified uniquely by the tuple $(b_{\min}(\overline{\mathbf{r}}), e_{\min}(\overline{\mathbf{r}}), b_{\max}(\overline{\mathbf{r}}), e_{\max}(\overline{\mathbf{r}}), v_Y(\overline{\mathbf{r}}))$ if either $b_{\min}(\overline{\mathbf{r}}) \neq b_{\max}(\overline{\mathbf{r}})$ or $e_{\min}(\overline{\mathbf{r}}) \neq e_{\max}(\overline{\mathbf{r}})$ ($\overline{\mathbf{r}}$ is not interval-uniform) and by $(b_{\min}(\overline{\mathbf{r}}), e_{\min}(\overline{\mathbf{r}}), b_{\max}(\overline{\mathbf{r}})$, otherwise ($\overline{\mathbf{r}}$ is interval-uniform). Since every element $\{b_{\min}(\overline{\mathbf{r}}), e_{\min}(\overline{\mathbf{r}}), b_{\max}(\overline{\mathbf{r}}), e_{\max}(\overline{\mathbf{r}}), v_Y(\overline{\mathbf{r}})\}$ may assume at most $|\mathbf{r}|$ different values, we have that the number of candidate *D-Clusters* w.r.t. Y in an instance $\mathbf{r}$ is roughly bounded by $|\mathbf{r}|^5$.

The idea behind the proof is that a candidate *D-cluster* may be identified uniquely by values $b_{\min}(\overline{\mathbf{r}}), e_{\min}(\overline{\mathbf{r}}), b_{\max}(\overline{\mathbf{r}}), e_{\max}(\overline{\mathbf{r}})$ and $v_Y(\overline{\mathbf{r}})$, if $\overline{\mathbf{r}}$ is not interval-uniform. In the other case, $\overline{\mathbf{r}}$ is interval-uniform and thus it suffices to take only the pair $t[B], t[E]$ for any tuple $t \in \overline{\mathbf{r}}$ to identify it (all the tuples in $\overline{\mathbf{r}}$ share the same interval by definition). Such a fixed representation of candidate *D-cluters* leads to the following corollary.

Corollary 2 *For every temporal relation schema $\mathcal{R} = R(U, B, E)$, every subset $Y \subseteq U$, and every instance $\mathbf{r}$ of $\mathcal{R}$, let n be the solution of problem D-MaxConsistent on $\mathbf{r}$ w.r.t. Y: for every D-consistent subset $\mathbf{r}' \subseteq \mathbf{r}$ with $|\mathbf{r}'| = n$, we have that every D-cluster $\mathbf{r}^c$ of $\mathbf{r}'$ is a candidate D-cluster for $\mathbf{r}$.*

Corollary 2 restricts the number of *D-consistent* instances $\mathbf{r}' \subseteq \mathbf{r}$ to be considered in order to solve *D-MaxConsistent*. Let $\mathbf{r}'$ be a *D-consistent* subset of $\mathbf{r}$, for which $|\mathbf{r}'|$ is the solution of problem *D-MaxConsistent* on $\mathbf{r}$ w.r.t. Y. Corollary 2 says that every *D-cluster* of $\mathbf{r}'$ must be a candidate *D-cluster* for $\mathbf{r}$ w.r.t. Y. Moreover, from Lemma 5 we have that *D-clusters* of $\mathbf{r}'$ must be totally ordered with respect to relation $<_c$ and thus two incomparable candidate *D-clusters* cannot both belong to $\mathbf{r}'$. The previous results guarantee the soundness and completeness of Algorithm 7.1. The algorithm builds a weighted DAG, whose nodes are all candidate *D-clusters* w.r.t. Y, plus two nodes, a source node s and a sink node f.

The edges of such a DAG represent relation $<_c$; more formally, for every pair $\overline{\mathbf{r}}, \overline{\mathbf{r}}'$ of candidate *D-clusters*, we have edge $(\overline{\mathbf{r}}, \overline{\mathbf{r}}')$ in the DAG if and only if $\overline{\mathbf{r}} <_c \overline{\mathbf{r}}'$. Moreover, there is an edge from node s (resp. to node f) to (resp. from) each candidate *D-cluster* $\overline{\mathbf{r}}$. For every edge $(\overline{\mathbf{r}}, \overline{\mathbf{r}}')$, its weight is calculated as the number of tuples $t \notin \overline{\mathbf{r}} \cup \overline{\mathbf{r}}'$, for which $I(t)$ is "between" $(b_{\max}(\overline{\mathbf{r}}), e_{max}(\overline{\mathbf{r}}))$ and $(b_{\max}(\overline{\mathbf{r}}'), e_{max}(\overline{\mathbf{r}}'))$, paying particular attention to count every tuple once.

It is easy to see that every path from s to f represents a *D-consistent* subset of $\mathbf{r}$ made by candidate *D-clusters* and, on the other side, any *D-consistent* subset of $\mathbf{r}$ made by candidate *D-clusters* may be represented as a path from s to f in the DAG. Moreover, the weight of the overall path is the number of tuples to delete from $\mathbf{r}$, in order to obtain $\mathbf{r}'$ (i.e $|\mathbf{r}\backslash\mathbf{r}'|$). In conclusion, finding the value w of the minimum weighted path from s to f using any well known algorithm on weighted DAGs leads to the solution of *D-MaxConsistency* w.r.t. Y on $\mathbf{r}$, which is $|\mathbf{r}| - w$. An example of such a weighted DAG for a small instance $\mathbf{r}$ is shown in Figure 10: for

Algorithm 7.1: D-MaxConsistency$(Y, \mathbf{r})$

$$
\begin{array}{l}
Endpoints \leftarrow \pi_B(\mathbf{r}) \cup \pi_E(\mathbf{r}) \\
Bounds \leftarrow \left\{ \begin{array}{l} (b_{\min}, e_{\min}, b_{\max}, e_{\max}) \ : \\ b_{\min}, e_{\min}, b_{\max}, e_{\max} \in Endpoints \wedge \\ b_{\min} \leqslant e_{\min} \wedge b_{\max} \leqslant e_{\max} \wedge \\ ((b_{\min}, e_{\min}) = (b_{\max}, e_{\max}) \vee \\ (b_{\min}, e_{\min}) < (b_{\max}, e_{\max})) \end{array} \right\} \\
Clusters \leftarrow \varnothing \\
\textbf{for each } (b_{\min}, e_{\min}, b_{\max}, e_{\max}) \in Bounds \\
\quad \textbf{do} \\
\quad \left\{ \begin{array}{l}
\textbf{if } \left(\begin{array}{l} (b_{\min}, e_{\min}) = (b_{\max}, e_{\max}) \wedge \\ \exists t \in \mathbf{r} \text{ s.t. } t[B] = b_{\min} \wedge t[E] = e_{\min} \end{array} \right) \\
\quad \textbf{then} \\
\quad \left\{ \begin{array}{l} \mathbf{r}^c \leftarrow \{t \in \mathbf{r} \ : \ t[B] = b_{\min} \wedge t[E] = e_{\min}\} \\ Clusters \leftarrow Clusters \cup \{\mathbf{r}^c\} \end{array} \right. \\
\quad \textbf{else} \\
\quad \left\{ \begin{array}{l} \textbf{for each } v \in \pi_Y(\mathbf{r}) \\ \quad \textbf{do} \\ \quad \left\{ \begin{array}{l} \textbf{if } \left(\begin{array}{l} \exists t_1, t_2, t_3, t_4 \in \mathbf{r} \text{ s.t. } \wedge \\ t_1[B] = b_{\min} \wedge t_1[E] \leqslant e_{\max} \wedge \\ t_2[E] = e_{\min} \wedge t_2[B] \geqslant b_{\min} \wedge \\ t_3[B] = b_{\max} \wedge t_3[E] \leqslant e_{\max} \wedge \\ t_4[E] = e_{\max} \wedge t_4[B] \geqslant b_{\min} \wedge \\ t_1[Y] = t_2[Y] = t_3[Y] = t_4[Y] = v \end{array} \right) \\ \quad \textbf{then} \\ \quad \left\{ \begin{array}{l} \mathbf{r}^c \leftarrow \left\{ t \in \mathbf{r} \ : \ \begin{array}{c} b_{\min} \leqslant t[B] \leqslant b_{\max} \wedge \\ e_{\max} \leqslant t[E] \leqslant e_{\max} \\ \wedge t[Y] = v \end{array} \right\} \\ Clusters \leftarrow Clusters \cup \{\mathbf{r}^c\} \end{array} \right. \end{array} \right. \end{array} \right. \\
V \leftarrow Clusters \cup \{s, f\} \\
E_s \leftarrow \{(s, \mathbf{r}^c) : \ \mathbf{r}^c \in Clusters\} \\
E_t \leftarrow \{(\mathbf{r}^c, f) : \ \mathbf{r}^c \in Clusters\} \\
E_c \leftarrow \{(\mathbf{r}_1^c, \mathbf{r}_2^c) : \ \mathbf{r}_1^c, \mathbf{r}_2^c \in Clusters \wedge \mathbf{r}_1^c < \mathbf{r}_2^c\} \\
\textbf{for each } \mathbf{r}^c \in Clusters \\
\quad \textbf{do} \\
\quad \left\{ \begin{array}{l} W(s, \mathbf{r}^c) \leftarrow \left| \left\{ t \ : \ \begin{array}{l} t \in \mathbf{r} \backslash \mathbf{r}^c \wedge t[B] < b_{\max}(\mathbf{r}^c) \vee \\ t[E] < e_{\max}(\mathbf{r}^c) \end{array} \right\} \right| \\ W(\mathbf{r}^c, t) \leftarrow \left| \left\{ t \ : \ \begin{array}{l} t \in \mathbf{r} \backslash \mathbf{r}^c \wedge t[B] \geqslant b_{\max}(\mathbf{r}^c) \wedge \\ t[E] \geqslant e_{\max}(\mathbf{r}^c) \end{array} \right\} \right| \end{array} \right. \\
\textbf{for each } \mathbf{r}_1^c <_c \mathbf{r}_2^c \in Clusters \\
\quad \textbf{do} \\
\quad \left\{ W(\mathbf{r}_1^c, \mathbf{r}_2^c) \leftarrow \left| \left\{ t : \left(\begin{array}{c} t \in \mathbf{r} \backslash (\mathbf{r}_1 \cup \mathbf{r}_2) \\ \wedge \\ \left(\begin{array}{l} b_{\max}(\mathbf{r}_1) \leqslant t[B] \wedge \\ t[B] < b_{\max}(\mathbf{r}_2) \wedge \\ t[E] \geqslant e_{\max}(\mathbf{r}_1) \end{array} \right) \\ \vee \\ \left(\begin{array}{l} e_{\max}(\mathbf{r}_1) \leqslant t[E] \wedge \\ t[E] < e_{\max}(\mathbf{r}_2) \wedge \\ t[B] \geqslant b_{\max}(\mathbf{r}_1) \end{array} \right) \end{array} \right) \right\} \right| \right. \\
Min \leftarrow SingleSourceSP(V, E_s \cup E_f \cup E_c, W, s) \\
\textbf{return } (|\mathbf{r}| - Min)
\end{array}
$$

the sake of readability we omitted transitive closure edges (edges $(\overline{\mathbf{r}}, \overline{\mathbf{r}}')$ for which exists $\overline{\mathbf{r}}''$ such that edges $(\overline{\mathbf{r}}, \overline{\mathbf{r}}'')$ and $(\overline{\mathbf{r}}'', \overline{\mathbf{r}}')$ belong to the DAG). Moreover in Figure 11 we provide the DAGs for the motivational scenario proposed in Section 3. There are several improvements that can be done to reduce the asymptotic complexity of Algorithm 7.1, but here we are interested in addressing the complexity class of *D-MaxConsistent* which is *P*-time. For instance, it can be proved that the way in which weights are assigned in the DAG forbids any transitive closure edge to be part of a minimum weighted path from s to f and thus such edges can be removed from the DAG. On the basis of the previous result we can claim the soundness and completeness of Algorithm 7.1 and give the following result.

Theorem 7 *For every temporal relation schema $\mathcal{R} = R(U, B, E)$, every (singleton) subset $Y \subseteq U$, and every instance $\mathbf{r}$ of $\mathcal{R}$, the complexity of problem D-$MaxConsistent(Y, \mathbf{r})$ is $\mathcal{O}(|\mathbf{r}|^{10})$.*

The proof is simple, it suffices to observe how Algorithm 7.1 works. Both its soundness and completeness rely on the results given in this section. In particular, according to Lemma 5, we have that the clusters must be totally ordered in the final solution. Moreover we have from Lemma 7 that the number of candidate clusters is bounded by $\mathcal{O}(|\mathbf{r}|^5)$. The idea behind Algorithm 7.1 is to retrieve the solution of D-$MaxConsistent(Y, \mathbf{r})$ as a shortest path between two special nodes, a source s and a sink f respectively, in a weighted DAG. The other nodes of such DAG are all the possible $\mathcal{O}(|\mathbf{r}|^5)$ candidate clusters (Lemma 5). There exists an edge from $\overline{\mathbf{r}}$ to $\overline{\mathbf{r}}'$ if and only if we have $\overline{\mathbf{r}} < \overline{\mathbf{r}}'$. The weight of such edge is exactly the number of tuples that need to be deleted in order to make $\overline{\mathbf{r}}$ and $\overline{\mathbf{r}}'$ two consecutive clusters. There exists an edge from the source node s to each cluster $\overline{\mathbf{r}}$ and its associated weight is the number of tuples that need to be deleted in order to make $\overline{\mathbf{r}}$ the first cluster w.r.t. the order $<$. In a completely symmetric way, there exists an edge from each cluster $\overline{\mathbf{r}}$ to the sink node f and its associated weight is the number of tuples that need to be deleted in order to make $\overline{\mathbf{r}}$ the last cluster w.r.t. the order $<$. It is easy to see that each path from s to f represents a *D-consistent* subset $\mathbf{r}'$ of $\mathbf{r}$. The weight of such path represents the number of tuples that has to be deleted from $\mathbf{r}$ in order to obtain $\mathbf{r}'$ (i.e., $|\mathbf{r} \backslash \mathbf{r}'|$). We can conclude that the result can be obtained via any shortest path procedure for DAGs, some of them have complexity $\mathcal{O}(V + E)$ where V is the number of nodes and E is the number of edges in the input DAG. Finally the complexity is $\mathcal{O}(|\mathbf{r}|^{10})$.

8 Maximal Consistency for M and O cases

Complexity analysis for M and O cases is done via a reduction from the classical *NP-Complete* problem *Max2Sat*. Let us now introduce the related basic concepts and notations. A literal l is a propositional variable p or its negation $\neg p$, and a clause Cl is a set of literals. Given a set of clauses $\mathcal{CL}$, the set of all propositional variables in $\mathcal{CL}$ is denoted by $prop(\mathcal{CL}) = \{p \ : \ \exists Cl \in \mathcal{CL}(p \in Cl \vee \neg p \in Cl)\}$. An assignment $\mathcal{V}$ is a set of literals such that for every propositional variable p we have $\{\neg p, p\} \nsubseteq \mathcal{V}$. An assignment $\mathcal{V}$ satisfies a set of clauses $\mathcal{CL}$ if and only if for each $Cl \in \mathcal{CL}$ we have that $\mathcal{V} \cap Cl \neq \varnothing$.

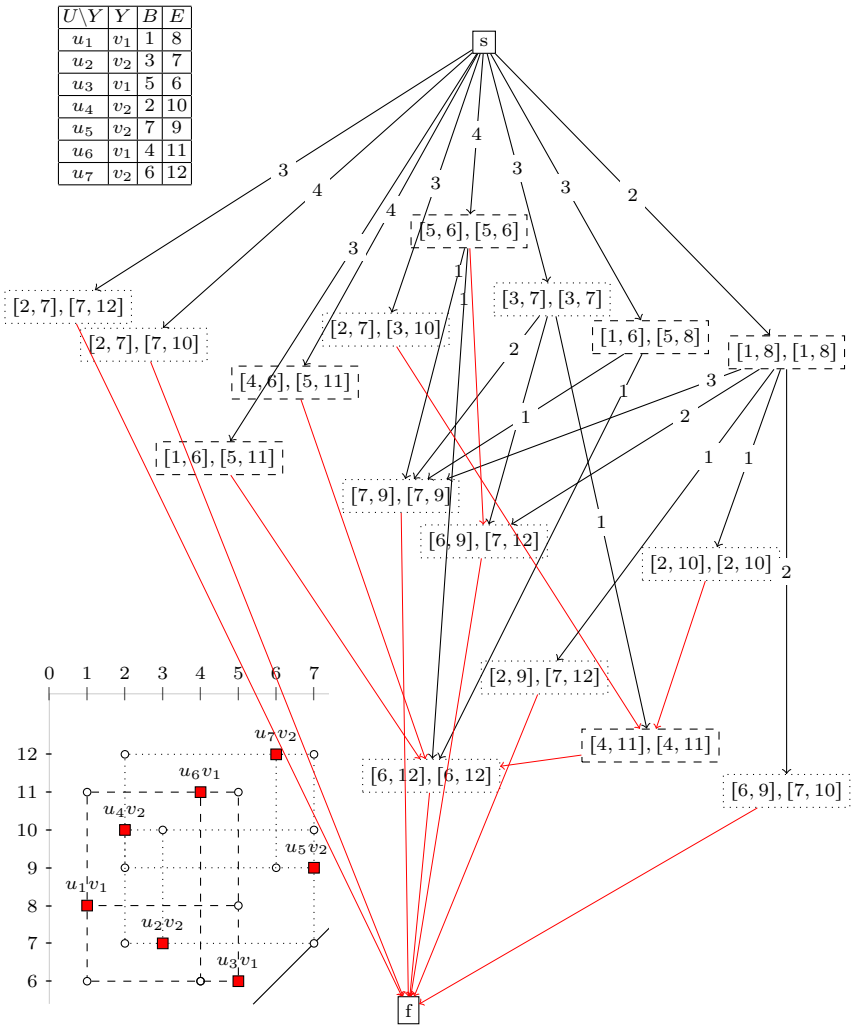


Fig. 10 An example of the weighted DAG for solving *D-MaxConsistency* on an instance **r** (edges with weight equal to zero are brighter and without label).

Problem 5 Given a set of clauses $\mathcal{CL} = \{Cl_0, \ldots, Cl_{n-1}\}$, each of them containing exactly two literals, problem $Max2Sat(\mathcal{CL})$ consists of determining the maximum natural number $k \leqslant n$, for which there exists a subset $\mathcal{CL}' \subseteq \mathcal{CL}$ with $|\mathcal{CL}'| = k$ and there exists an assignment $\mathcal{V}$ that satisfies $\mathcal{CL}'$.

This formulation of the problem is slightly different from the original one [17], but it is not difficult to show that it is equivalent from the complexity point of view[2].

[2] This is the formulation of the problem in its *evaluation* version and the same assumption made in Section 2.2 for $\sim$-*MaxConsistent* holds for $Max2Sat$ too.

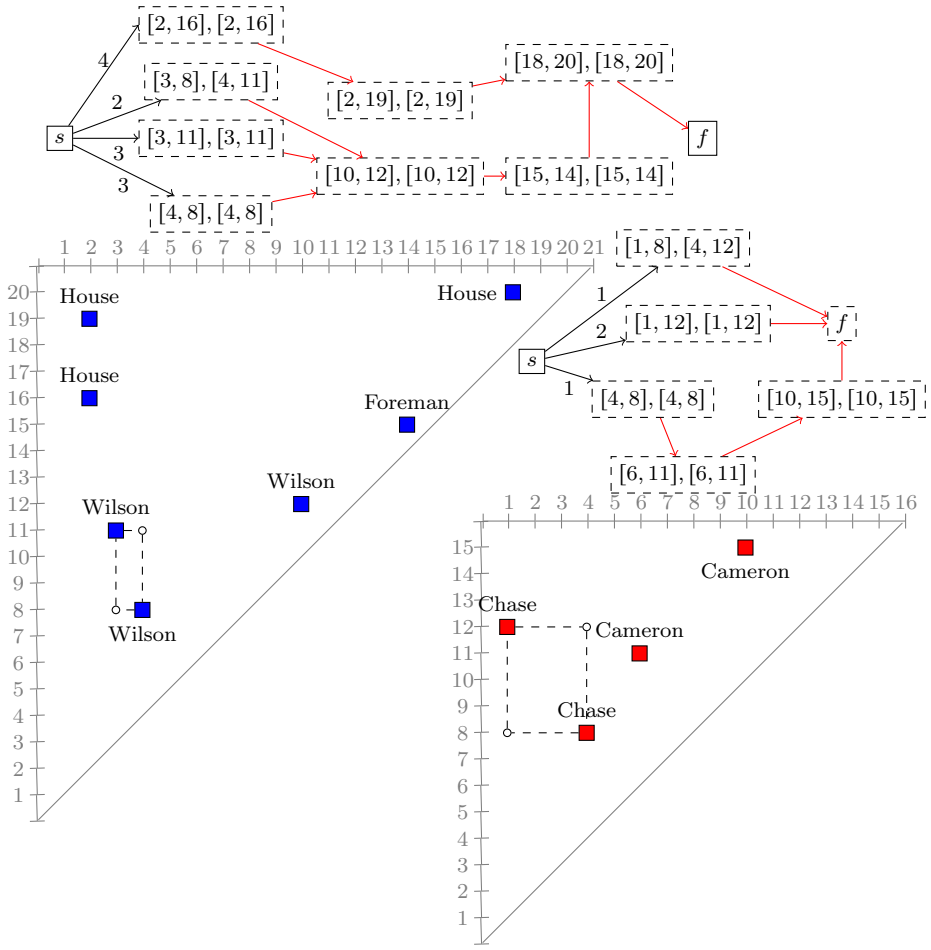


Fig. 11 The two DAGs built on the relation *PatTherapies* of Figure 2 for the ITFD $PatId \rightarrow_D Phys$. The DAG on the left is built on the tuples with $PatId = 1$ while the one on the right on the tuple with $PatId = 2$.

Theorem 8 *Max2Sat is NP-Complete*

In the following we provide two closely related polynomial time reductions from the problem *Max2Sat* to the problems *M-MaxConsistent* and *O-MaxConsistent* respectively. By means of such reductions and Theorem 8 we can conclude that *M-MaxConsistent* and *O-MaxConsistent* are *NP-Complete* problems as well.

Let $\mathcal{CL} = \{Cl_0, \ldots, Cl_{n-1}\}$ be a set of clauses. First we fix an arbitrary total order $<_p$ over $Prop(\mathcal{CL})$. We assume without loss of generality that for every propositional variable p we have that the clause containing both p and $\neg p$ does not belong to $\mathcal{CL}$, otherwise it suffices to remove such a clause from $\mathcal{CL}$. Under the previous assumption together with the fact that all clauses in $\mathcal{CL}$ feature exactly two distinct literals, for every clause $Cl_j \in \mathcal{CL}$ with $1 \leqslant j \leqslant n$ we can identify $p_i <_p p_{\bar{i}}$ as two propositional variables which appear in Cl_j. We define set $Sat(Cl_j)$

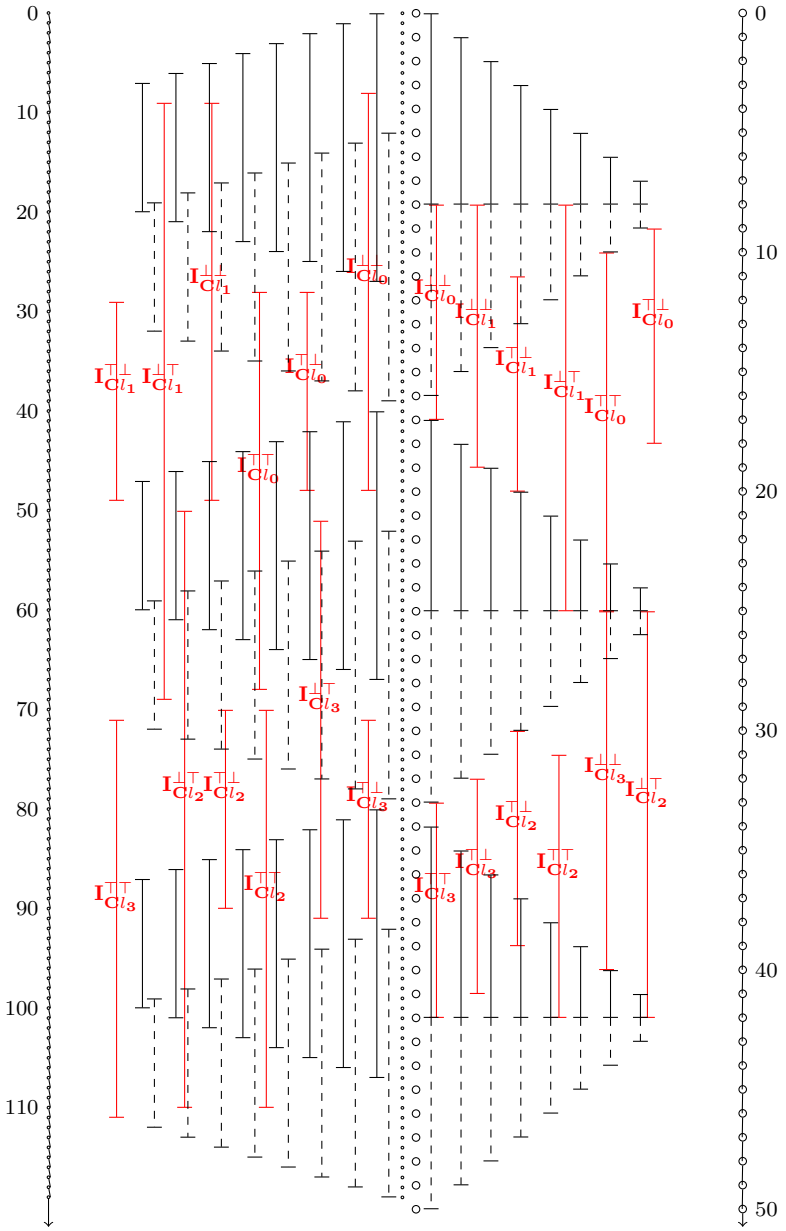


Fig. 12 Encoding in *M-MaxConsistent* (right) and *O-MaxConsistent* (left) for the Max2Sat problem with input formula $(p \vee \neg q) \wedge (\neg p \vee \neg q) \wedge (q \vee r) \wedge (q \vee \neg r)$ (the different time flows are reported on their relative sides in order to improve readability).

as the set of all assignments that may contain only literals $p_i, \neg p_i, p_{\bar{i}}$ and $\neg p_{\bar{i}}$, which satisfy clause Cl_j: $Sat(Cl_j) = \{(*_1, *_2) \mid *_1 \in \{p_i, \neg p_i\}, *_2 \in \{p_{\bar{i}}, \neg p_{\bar{i}}\}$ s.t. $\{*_1, *_2\} \cap Cl_j \neq \varnothing\}$. It is easy to see that for every j we have $|Sat(Cl_j)| = 3$. Given a

clause Cl_j, we denote $Cl_j|_1$ the first propositional letter of Cl_j and $Cl_j|_2$ the second one, regardless if they appear positive or negated in Cl_j (e.g., if $Cl_j = \{\neg p_1, \neg p_4\}$ we have $Cl_j|_1 = p_1$ and $Cl_j|_2 = p_4$). We build an instance $\mathbf{r}_{\mathcal{CL}}$ of temporal relational schema $R = (\{Y\}, B, E)$ where the domain of Y consists of three elements $Dom(Y) = \{\top, \bot, c\}$. The value $\top$ labels tuples associated to assignments to true for each variable in $Prop(\mathcal{CL})$. The value $\bot$ labels tuples associated to assignments to false for each variable in $Prop(\mathcal{CL})$. Moreover the value c means *clause* and labels tuples corresponding to truth assignments for the clauses in $\mathcal{CL}$. We associate $4 \cdot n$ intervals $I^{\top}_{i_0}, \ldots, I^{\top}_{i_{2n-1}}$ and $I^{\bot}_{i_0}, \ldots, I^{\bot}_{i_{2n-1}}$ with each propositional letter $p_i \in Prop(\mathcal{CL})$, with $i = 0, \ldots, m-1$. For each $j = 0, \ldots, 2n-1$ such intervals are defined as follows:

(M-case $I^{\top}$) $I^{\top}_{i_j} = [4in + i + j \ , \ 4in + i + 2n]$;

(M-case $I^{\bot}$) $I^{\bot}_{i_j} = [4in + i + 2n \ , \ 4in + i + 2n + j + 1]$;

(O-case $I^{\top}$) $I^{\top}_{i_j} = [10in + j \ , \ 10in + 7n - j - 1]$;

(O-case $I^{\bot}$) $I^{\bot}_{i_j} = [10in + j + 3n \ , \ 10in + 10n - j - 1]$.

We associate 4 intervals defined as follows with each clause Cl_j with $j = 0, \ldots, n-1$. Let $p_i, p_{\bar{i}}$, with $p_i <_p p_{\bar{i}}$, be the two propositional letters that appear in Cl_j. We define 4 intervals for each interval relation (i.e., 4 for the M relation and 4 for the O relation) related to the possible four assignment of p_i and $p_{\bar{i}}$:

(M-case $I^{\bot\bot}$) for $p_i = \bot$ and $p_{\bar{i}} = \bot$ we define $I^{\bot\bot}_{Cl_j} = [4in + i + 2n \ , \ 4\bar{i}n + \bar{i} + 2j]$;

(M-case $I^{\bot\top}$) for $p_i = \bot$ and $p_{\bar{i}} = \top$ we define $I^{\bot\top}_{Cl_j} = [4in + i + 2n + 1 \ , \ 4\bar{i}n + \bar{i} + 2n + 1]$;

(M-case $I^{\top\bot}$) for $p_i = \top$ and $p_{\bar{i}} = \bot$ we define $I^{\top\bot}_{Cl_j} = [4in + i + 2j + 2n + 1 \ , \ 4\bar{i}n + \bar{i} + 2j + 1]$;

(M-case $I^{\top\top}$) for $p_i = \top$ and $p_{\bar{i}} = \top$ we define $I^{\top\top}_{Cl_j} = [4in + i + 2j + 2n + 2 \ , \ 4\bar{i}n + \bar{i} + 2n + 1]$;

(O-case $I^{\bot\bot}$) for $p_i = \bot$ and $p_{\bar{i}} = \bot$ we define $I^{\bot\bot}_{Cl_j} = [10in + j + 2n \ , \ 10\bar{i}n + j + 2n]$;

(O-case $I^{\bot\top}$) for $p_i = \bot$ and $p_{\bar{i}} = \top$ we define $I^{\bot\top}_{Cl_j} = [10in + j + 2n \ , \ 10\bar{i}n + j + 7n]$;

(O-case $I^{\top\bot}$) for $p_i = \top$ and $p_{\bar{i}} = \bot$ we define $I^{\top\bot}_{Cl_j} = [10in + j + 7n \ , \ 10\bar{i}n + j + 2n]$;

(O-case $I^{\top\top}$) for $p_i = \top$ and $p_{\bar{i}} = \top$ we define $I^{\top\top}_{Cl_j} = [10in + j + 7n \ , \ 10\bar{i}n + j + 7n]$.

Now we are ready to define the instance $\mathbf{r}_{\mathcal{CL}}$ of R as the set of tuples:

$$\bigcup_{i=0}^{n-1} \bigcup_{j=0}^{m-1} \{t \ : \ t[Y] = \top \wedge [t[B], t[E]] = I^{\top}_{i_j}\} \cup$$

$$\bigcup_{i=0}^{n-1} \bigcup_{j=0}^{m-1} \{t \mid t[Y] = \bot \wedge [t[B], t[E]] = I^{\bot}_{i_j}\} \cup$$

$$\bigcup_{j=0}^{j-1} \bigcup_{(*_1,*_2) \in Sat(Cl_j)} \{t \mid t[Y] = c \wedge [t[B], t[E]] = I^{*_1*_2}_{Cl_j}\}.$$

It is worth to notice that intervals of kind $I^{*_1*_2}_j$ belong to instance $\mathbf{r}$ if and only if $(*_1, *_2)$ represents an assignment that satisfies clause Cl_j, for each clause we have exactly three tuples in $\mathbf{r}_{\mathcal{CL}}$ with value c for attribute Y. An example of the reduction for both the M and O cases on a very small set of clauses is depicted in Figure 12. In Figure 12 we represent both the *M-MaxConsistent* (right) and the *O-MaxConsistent* (left) instances created for the $Max2Sat$ instance $\psi = (p \vee \neg q) \wedge (\neg p \vee \neg q) \wedge (q \vee r) \wedge (q \vee \neg r)$. The flow of time goes from the top to the bottom of the picture and we may notice how the *O-MaxConsistent* instance on the left requires a larger amount of time points than the *M-MaxConsistent*

instance on the right.

The intervals are represented according to the value of attribute Y as follows: i) full black intervals are associated with tuples t having $t[Y] = \top$; ii) dashed black intervals are associated with tuples t having $t[Y] = \bot$; iii) lighter intervals are associated with tuples t having $t[Y] = c$. Let us consider now the case of the M relation (right). We have that the black intervals are grouped into 3 sets of adjacent intervals one for each variable in ψ. The set on the top is associated with variable p, the one in the middle with variable q, and the one on the bottom is associated with variable r. It is easy to observe that each one of such sets graphically resembles the shape of a triangle. Such a triangle is the result of two components, the black full intervals on the top, which represent the positive assignment of the relative variable, and the black dashed intervals on the bottom which represent the negative assignment of the same variable. For each clause Cl in ψ and for each truth assignment of it, we have exactly one lighter interval. Since each clause has exactly three truth assignments, we have three red intervals for each clause. Each red interval meets or is met by all and only the intervals that are inconsistent with its truth assignment. For instance, interval $I^{\bot\top}_{Cl_2}$ on the right side of Figure 12 represents the truth assignment $q = \bot$ and $r = \top$ for clause $(q \vee r)$ of ψ. We may observe how $I^{\bot\top}_{Cl_2}$ is met by all the full black intervals belonging to the triangle in the middle representing the assignment $q = \top$, which is clearly inconsistent with its assignment.

Moreover $I^{\bot\top}_{Cl_2}$ meets all the dashed intervals belonging to the triangle on the bottom and representing the assignment $r = \bot$, which is again inconsistent with its assignment. Notice that these intervals are the only intervals that are in relation either M or $\overline{M}$ with $I^{\bot\top}_{Cl_2}$. Clearly, the instance violates M-*Consistency* since $\top, \bot$ and c are pairwise distinct values for Y. It is easy to observe that by construction we have that, if we want to achieve consistency by removing tuples, we are forced to remove completely the top part or the bottom part of each triangle. Otherwise, we would have at least one full black interval that meets a dashed one. Such a removal operation can be directly translated into an assignment. For instance, let us consider Figure 13 which shows both the solutions for instances in Figure 12. Looking again on the right side (M-

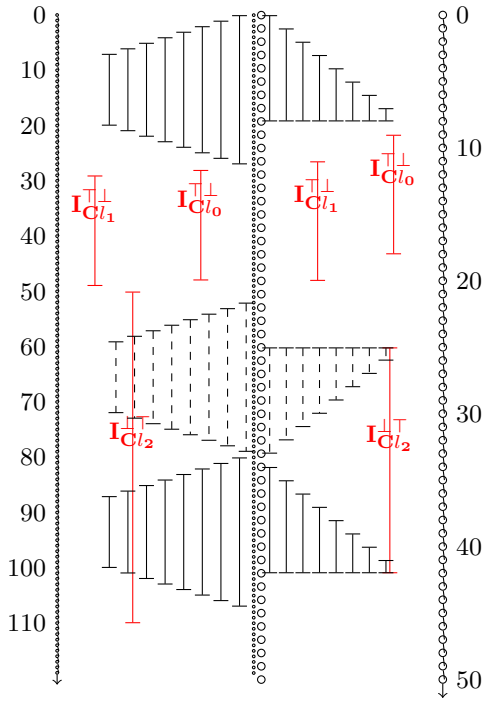


Fig. 13 A solution for the encoding of the $Max2Sat$ instance $(p \vee \neg q) \wedge (\neg p \vee \neg q) \wedge (q \vee r) \wedge (q \vee \neg r)$ depicted in Figure 12. The assignment represented is $p = \top, q = \bot, r = \top$.

relation) we have that the triangle on the top has lost its dashed portion and thus it means that p has been assigned to $\top$. On the other side the triangle in the middle has lost its black full portion and thus it means that q has been assigned to $\bot$. Finally the triangle on the bottom has lost its dashed portion and thus it means that r has been assigned to $\top$. The resulting assignment is $p = \top, q = \bot$, and $r = \top$. It is easy to see by comparison with Figure 12 that the only three red intervals that may be kept in such an assignment for the black ones are the intervals $I^{\top\bot}_{Cl_0}, I^{\top\bot}_{Cl_1}$ and $I^{\bot\top}_{Cl_2}$. Notice that the only three clauses in ψ that are satisfied by the assignment $p = \top, q = \bot$, and $r = \top$ are Cl_0, Cl_1 and Cl_2. On the other side $Cl_3 = (q \vee \neg r)$ is not satisfied since $q = \bot$ and $r = \top$. As a matter of fact 3 is the maximum number of clauses that can be satisfied by an assignment in ψ. The very same arguments hold for the instance of *O-MaxConsistent* on the left of Figures 12 and 13. In the following we shall prove formally that, among all the solutions for instances built using one of our reductions, we have a solution that resembles the one proposed in Figure 13. The distinctive feature of such solutions is that we take all the black intervals either on the top or on the bottom of each triangle. Taking advantage of the maximality requirement for the problem *M-MaxConsistent* (resp. *O-MaxConsistent*), we shall prove that the number of red intervals that "survive" in such a solution is exactly the maximum number of clauses that can be satisfied in the original $Max2Sat$ instance.

Such a construction requires quadratic time w.r.t. the size of $\mathcal{CL}$. The definition of $\mathbf{r}_{\mathcal{CL}}$ requires a linear parsing of the input in order to determine $prop(\mathcal{CL})$ and values m and n. Moreover, we need a quadratic iteration of $\mathcal{O}(n \cdot m)$ operations for populating $\mathbf{r}_{\mathcal{CL}}$ with tuples relative to intervals $I^{\top}_{i_j}$ and $I^{\bot}_{i_j}$, plus a linear parsing for populating $\mathbf{r}_{\mathcal{CL}}$ with tuples relative to intervals $I^{*_1 *_2}_{C_j}$. Let $\mathbf{f}_M$ and $\mathbf{f}_O$ be the functions that perform the reduction from $\mathcal{CL}$ to $\mathbf{r}_{\mathcal{CL}}$ in M case and in O case, respectively. For every set of clauses $\mathcal{CL}$ we define an *assignment set* as a subset of tuples $\mathbf{r}^a_{\mathcal{CL}} \subseteq \mathbf{f}_*(\mathcal{CL})$, with $* \in \{M, O\}$, such that, for every $p_j \in Prop(\mathcal{CL})$, either for each $0 \leqslant i \leqslant n-1$ we have $I^{\top}_{i_j} \in \mathbf{r}^a_{\mathcal{CL}}$ or for each $0 \leqslant i \leqslant n-1$ we have $I^{\bot}_{i_j} \in \mathbf{r}^a_{\mathcal{CL}}$. Informally, every assignment set represents (and can be trivially translated into) an admissible assignment set for the original clause set $\mathcal{CL}$. In Figure 13 we represent both an *M-Consistent* and an *O-Consistent* solution for the instance presented in Figure 12. As we may notice, among the red intervals only the ones consistent with the truth assignment "survive" and thus the number of lighter intervals in the solution is exactly the solution for the $Max2Sat$ instance encoded. Such a behaviour is formally proved in the following result.

Lemma 8 *For every set of clauses $\mathcal{CL}$, where each clause in $\mathcal{CL}$ contains exactly two literals, let k be the solution of problem M-MaxConsistent$(Y, \mathbf{f}_M(\mathcal{CL}))$ (resp. O-MaxConsistent$(Y, \mathbf{f}_O(\mathcal{CL}))$). There exists an $M-$ consistent (resp. $O-$ consistent) set $\mathbf{r} \subseteq \mathbf{f}_M(\mathcal{CL})$ (resp. $\mathbf{r} \subseteq \mathbf{f}_O(\mathcal{CL})$) w.r.t. to Y with $|\mathbf{r}| = k$ and an assignment set $\mathbf{r}^a_{\mathcal{CL}}$ with $\mathbf{r}^a_{\mathcal{CL}} \subseteq \mathbf{r}$.*

Proof We prove the claim for the *meets* case, as the *overlaps* one is analogous. Suppose that *M-MaxConsistent*$(Y, \mathbf{f}_M(\mathcal{CL})) = k$ and thus there exists an M-consistent set $\mathbf{r} \subseteq \mathbf{f}_M(\mathcal{CL})$ with $|\mathbf{r}| = k$. Now we have to prove that there exists an M-consistent

set $\mathbf{r}' \subseteq \mathbf{f}_M(\mathcal{CL})$ with $|\mathbf{r}'| = k$ and an assignment set $\mathbf{r}^a_{\mathcal{CL}} \subseteq \mathbf{r}'$. We obtain $\mathbf{r}'$ by "repairing" $\mathbf{r}$ with successive iterations. Before starting with the procedure, we observe that there are only three ways in which $\mathbf{r}$ may not contain an assignment set:

(i) there exists $p_i \in Prop(\mathcal{CL})$, two indexes $0 \leqslant j \neq j' \leqslant 2n-1$, and two tuples t, t' such that $t[Y] = t'[Y] = \top$, $[t[B], t[E]] = I^{\top}_{i_j}$, $[t'[B], t'[E]] = I^{\top}_{i_{j'}}$, $t \in \mathbf{r}$ and $t' \notin \mathbf{r}$;
(ii) there exists $p_i \in Prop(\mathcal{CL})$, two indexes $0 \leqslant j \neq j' \leqslant 2n-1$, and two tuples t, t' such that $t[Y] = t'[Y] = \bot$, $[t[B], t[E]] = I^{\bot}_{i_j}$, $[t'[B], t'[E]] = I^{\bot}_{i_{j'}}$, $t \in \mathbf{r}$ and $t' \notin \mathbf{r}$;
(iii) there exists $p_i \in Prop(\mathcal{CL})$ such that for every $0 \leqslant j \leqslant 2n-1$, and for every tuple t with $[t[B], t[E]] = I^{\bot}_{i_j}$ and either $t[Y] = \top$ or $t[Y] = \bot$, we have $t \notin \mathbf{r}$. It is worth to notice that it cannot be the case in which there exists $p_i \in Prop(\mathcal{CL})$, two indexes $0 \leqslant j \neq j' \leqslant 2n-1$, and two tuples t, t' such that $t[Y] = \bot$ and $t'[Y] = \top$ (resp. $t[Y] = \top$ and $t'[Y] = \bot$), $[t[B], t[E]] = I^{\bot}_{i_j}$, $[t'[B], t'[E]] = I^{\top}_{i_{j'}}$ (resp. $[t[B], t[E]] = I^{\top}_{i_j}$, $[t'[B], t'[E]] = I^{\bot}_{i_{j'}}$), $t, t' \in \mathbf{r}$.

Indeed, such a scenario violates M-consistency since $I^{\top}_{i_{j'}}\ M\ I^{\bot}_{i_j}$ (resp. $I^{\top}_{i_j}\ M\ I^{\bot}_{i_{j'}}$) and $t[Y] \neq t'[Y]$. It also to be pointed out that, if $\mathbf{r}$ is M-Consistent and it does not satisfy each one of the three conditions above, then $\mathbf{r}$ contains an assignment set $\mathbf{r}^a_{\mathcal{CL}}$. We begin our iterative procedure that builds $\mathbf{r}'$ by setting $\mathbf{r}_0 = \mathbf{r}$. As invariant conditions we guarantee that at each step $h \geqslant 0$ instance $\mathbf{r}_h$ satisfies $|\mathbf{r}_h| \geqslant k$ and it is *M-Consistent*. Our invariant condition takes into account the fact that k may be increased. We shall not pay too much attention to this case, because, since our instance is maximal by hypothesis, a contradiction will arise. At each step we shall focus only on the fact that we introduce in the new instance $\mathbf{r}_{h+1}$ at least the same number of tuples that we remove from $\mathbf{r}_h$. At each step i we choose one among the following cases, according to the fact that its precondition holds.

- case (i) holds for $\mathbf{r}_h$. First, we observe that for each $t'' \in \mathbf{f}_M(\mathcal{CL})$ with $t''[Y] = c$ and $[t'[B], t'[E]]\ M\ [t''[B], t''[E]]$ we have $t'' \notin \mathbf{r}_h$. This is a consequence of the fact that $t \in \mathbf{r}_h$ and thus $[t[B], t[E]]\ M\ [t''[B], t''[E]]$ and $t[Y] = \top \neq c = t''[Y]$. Otherwise, *M-Consistency* would be violated but it is guaranteed by the invariant conditions. By construction we have that there exists at most one tuple t'' such that $t''[Y] = c$ and $[t''[B], t''[E]]\ M\ [t'[B], t'[E]]$. Thus, we can define $\mathbf{r}_{h+1} = (\mathbf{r}_i \backslash \{t''\}) \cup \{t'\}$ and both the invariants turn out to be preserved;
- case (ii) holds for $\mathbf{r}_h$. This case is completely symmetric to case (i).
- case(iii) holds for $\mathbf{r}_h$. In this case we can choose which assignment for the letter p_i may be introduced as a set of tuples. Such a choice makes no difference with respect to the invariant conditions. Thus, we choose the subset $\mathbf{r}_i \subseteq \mathbf{f}_M(\mathcal{CL})$ such that $\mathbf{r}_i = \{t : t[Y] = \top \wedge [t[B], t[E]] = I^{\top}_{i_j}$ with $0 \leqslant j \leqslant 2n-1\}$. Clearly $|\mathbf{r}_i| = 2n$. Consider now the set $\bar{\mathbf{r}}_i \subseteq \mathbf{f}_M(\mathcal{CL})$ such that $\bar{\mathbf{r}}_i = \{t \in \mathbf{f}_M(\mathcal{CL}) : t[Y] = c \wedge [t[B], t[E]] = I^{\bot *_2}_{C_j}$ with $0 \leqslant j \leqslant 2n-1$ and $p_i = C_j|_1\} \cup \{t : t[Y] = c \wedge [t[B], t[E]] = I^{*_1 \bot}_{C_j}$ with $0 \leqslant j \leqslant 2n-1$ and $p_i = Cl_j|_2\}$. By construction we have that $|\bar{\mathbf{r}}_i| \leqslant 2n$. From condition (iii) and the invariant ones we have that if a tuple $t \in \mathbf{r}$ satisfies either $[t[B], t[E]]\ M\ [t'[B], t'[E]]$ or $[t'[B], t'[E]]\ M\ [t[B], t[E]]$ with $t' \in \mathbf{r}_i$, then it satisfies $t \in \bar{\mathbf{r}}_i$. We can conclude

that the assignment $\mathbf{r}_{h+1} = (\mathbf{r}_h \backslash \overline{\mathbf{r}}_i) \cup \mathbf{r}_i$ satisfies both the invariant conditions and this is the one we choose;
- No one among conditions (i), (ii), and (iii) is satisfied. Thus $\mathbf{r}_h$ contains an assignment set $\mathbf{r}^a_{\mathcal{CL}}$. We put $\mathbf{r}' = \mathbf{r}_h$ and terminate the procedure.

Since at each of the first three cases we introduce at least one new tuple $t \in \mathbf{r}^a_{\mathcal{CL}}$ in $\mathbf{r}_h$ for some assignment $\mathbf{r}^a_{\mathcal{CL}}$ and such set is finite, we have that the procedure above reaches the fourth case in at most $|\mathbf{r}^a_{\mathcal{CL}}|$ steps and thus its termination is proved.

This result directly connects the solution of $M\text{-}MaxConsistent(Y, \mathbf{f}_M(\mathcal{CL}))$ (resp. $O\text{-}MaxConsistent(Y, \mathbf{f}_O(\mathcal{CL}))$) to the solution of $Max2Sat(\mathcal{CL})$.

Lemma 9 *For every set of clauses $\mathcal{CL} = \{Cl_1, \ldots, Cl_n\}$, each containing exactly two literals, let $prop(\mathcal{CL}) = \{p_1, \ldots, p_m\}$ be the set of propositional variables in $\mathcal{CL}$. We have that $Max2Sat(\mathcal{CL}) = M\text{-}MaxConsistent\ (Y, \mathbf{f}_M(\mathcal{CL})) - m \cdot n = O\text{-}MaxConsistent(Y, \mathbf{f}_O(\mathcal{CL})) - m \cdot n$.*

Proof We consider the *meets* case, as the *overlaps* one is analogous. Let $\mathbf{r}$ be the *M-Consistent* subset $\mathbf{r} \subseteq \mathbf{f}_M(\mathcal{CL}))$ such that $|\mathbf{r}| = M\text{-}MaxConsistent(Y, \mathbf{f}_M(\ \mathcal{CL}))$. From Lemma 8 we may assume that $\mathbf{r}$ contains an assignment set $\mathbf{r}^a_{\mathcal{CL}} \subseteq \mathbf{r}$. Using $\mathbf{r}^a_{\mathcal{CL}}$, we define the assignment $\sigma_{\mathbf{r}} : prop(\mathcal{CL}) \to \{\top, \bot\}$ as $\sigma_{\mathbf{r}}(p_i) = \top$ if and only if for every $0 \leqslant j \leqslant 2n-1$ there exists $t \in \mathbf{r}$ with $I(t) = I^{\top}_{i_j}$. By construction, we have that for each $1 \leqslant j \leqslant m$ there exists a tuple $t \in \mathbf{r}$ such that $I(t) = I^{*_1,*_2}_{Cl_j}$ if and only if $p_i = *_1$ and $p_{\overline{i}} = *_2$ is a truth assignment for clause Cl_j and $\sigma_{\mathbf{r}}(p_i) = *_1$ and $\sigma_{\mathbf{r}}(p_{\overline{i}}) = *_2$. It means that the set of clauses $\mathcal{CL}' \subseteq \mathcal{CL}$ that are satisfied by the assignment is $\mathcal{CL}' = \{Cl_j \in \mathcal{CL} : \exists t \in \mathbf{r}, \exists *_1, *_2 \in \{\top, \bot\} I^{*_1*_2}_{Cl_j} = I(t)\}$ and thus $|\mathcal{CL}'| = |\{t \in \mathbf{r} : t[Y] = c\}| = M\text{-}MaxConsistent\ (Y, \mathbf{f}_M(\mathcal{CL})) - m \cdot n$. Suppose now by contradiction that there exists an assignment σ' such that $\mathcal{CL}'' = \{Cl_j : \sigma' \models Cl_j\}$ and $|\mathcal{CL}''| > |\mathcal{CL}'|$. Thus, we could build the following assignment set $\mathbf{r}^{a'}_{\mathcal{CL}} = \{t \in \mathbf{f}_M(\mathcal{CL}) : \exists i\ I(t) = I^{\sigma'(p_i)}_{i_j},\ 0 \leqslant j \leqslant 2n-1\}$ and a set $\mathbf{r}_c = \{t \in \mathbf{f}_M(\mathcal{CL}) : \exists *_1, *_2 \in \{\top, \bot\}, I^{*_1*_2}_{Cl_j} = I(t), \sigma'(Cl_j|_1) = *_1 \wedge \sigma'(Cl_j|_2) = *_2\}$. Clearly $\mathbf{r}^{a'}_{\mathcal{CL}} \cup \mathbf{r}_c$ for construction is an *M-Consistent* subset of $\mathbf{f}_M(\mathcal{CL})$. Moreover $|\mathbf{r}^a_{\mathcal{CL}}| = |\mathbf{r}^{a'}_{\mathcal{CL}}|$, $|\mathbf{r}_c| = |\mathcal{CL}''| > |\mathcal{CL}'|$, which would mean $|\mathbf{r}^{a'}_{\mathcal{CL}} \cup \mathbf{r}_c| > M\text{-}MaxConsistent\ (Y, \mathbf{f}_M(\mathcal{CL}))$ (contradiction).

Since both reduction functions $\mathbf{f}_M$ and $\mathbf{f}_O$ operate in polynomial time (i.e. logarithmic space), we can conclude this section with the following result.

Theorem 9 *Problems M-MaxConsistent and O-MaxConsistent are NP-Complete.*

9 Approximate ITFDs and database repair

Originally, we came to the problem of determining the maximum size for a maximal consistent subsets of a relation $\mathbf{r}$ w.r.t to a given ITFD $X \to_{\sim} Y$ as a way to determine if it holds under a given approximation ϵ. Our approach is oriented towards the discovery of dependencies among data in a given instance $\mathbf{r}$.

There is a completely symmetric representation of our results in the field of *database repairing* [16]. Given a schema $\mathcal{R}$, a relation $\mathbf{r}$ and a finite set Σ of first-order defined constrains over $\mathcal{R}$, we say that a relation $\mathbf{r}'$ is a *repair* or $\mathbf{r}$ with respect

to Σ if it satisfies all constraints in Σ. For each $\mathbf{r}''$ that satisfies all constraints in Σ, we have $|\mathbf{r} \bigoplus \mathbf{r}'| \leqslant |\mathbf{r} \bigoplus \mathbf{r}''|$, where $\bigoplus$ represents the symmetric union (i.e., the union of the disjoint parts of the two sets). It is worth to notice that there are several notions of *repair* in literature (see [15] for example). In particular, we have that our problem, due to the particular constraints expressed through an ITFD, turns out to be related to a class of repairs called *subset repairing*, where the set $\mathbf{r}'$ is required to be a subset of $\mathbf{r}$. According to definitions given in [1], three classes among the possible constraints in Σ are of particular interest[3]:

- an equality-generating dependency (*egd*) is a first order formula of the form $\forall \mathbf{x}(\phi(\ \mathbf{x}) \rightarrow x = x')$, where $\phi(\mathbf{x})$ is a conjunction of atomic formulas over $\mathcal{R}$, each variable $x \in \mathbf{x}$ occurs in $\phi(\mathbf{x})$, and $x, x' \in \mathbf{x}$;
- a denial constraint (*dc*) is a first order formula of the form $\forall \mathbf{x} \neg(\alpha(\mathbf{x}) \wedge \beta(\mathbf{x}))$, where $\alpha(\mathbf{x})$ is a conjunction of atomic formulas over $\mathcal{R}$ and β is a conjunction of comparison atoms $x = x'$, $x \leqslant x'$, $x \neq x'$ and $x < x'$, where both x and x' occur in $\alpha(\mathbf{x})$;
- a tuple-generating dependency (*tgd*) is a first order formula of the form $\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \exists \mathbf{y} \psi(\mathbf{x}, \mathbf{y}))$, where $\phi(\mathbf{x})$ is a conjunction of atomic formulas over $\mathcal{R}$, each variable $x \in \mathbf{x}$ occurs in $\phi(\mathbf{x})$, and $\psi(\mathbf{x}, \mathbf{y})$ is a conjunction of atomic formulas with variables in $\mathbf{x}$ and $\mathbf{y}$.

It is easy to see that an (*egd*) is logically equivalent to a (*dc*) but not viceversa. ITFDs are a class of dependencies different from the ones shown above. A problem related to our approach is the *repair checking* problem: given two relations $\mathbf{r}$ and $\mathbf{r}'$ on the same schema $\mathcal{R}$ and a set of constraints Σ over $\mathcal{R}$, let us try to answer the question "is $\mathbf{r}'$ a repair of $\mathbf{r}$ with respect to Σ?". The complexity of such a problem has been addressed for various notions of repair and various classes of dependencies including the ones shown above [1,6,23]. An ITFD $X \rightarrow_{\sim} Y$ can be translated as $\forall \mathbf{x} \forall y \forall y' \forall x_1 \forall x_1'\ \forall x_2 \forall x_2'(A(\mathbf{x}, y, x_1, x_2)\ \wedge A(\mathbf{x}, y', x_1', x_2') \wedge \beta(x_1, x_2, x_1', x_2') \rightarrow y = y')$. We may notice that we may consider only an atom A of $\mathcal{R}$ at a time. Moreover the syntax on atoms is very restricted. Indeed only a conjunction of the same atom which shares the $\mathbf{x}$ attributes is allowed. Let us notice that $\mathbf{x}$ is the counterpart of the atemporal attributes X. Formula $\beta(x_1, x_2, x_1',\ x_2')$ is a conjunction of comparison atoms and it is the translation of the interval relation $\sim$. Here we have that x_1, x_1' are the counterparts of attribute B and x_2, x_2' are the counterparts of attribute E.

In the present work, we do not consider sets of ITFDs. We focus on a single ITFD and our problem consists of finding the value corresponding to the cardinality of the possible repairs. It is straightforward to adapt our algorithm to obtain a repair without affecting the complexity of the problem. Thus, as a by-product of our results we have the complexity classification for the problem of ITFD repairing when the set of dependencies consists of a single one. Despite the fact that the syntax of our constraints seems more restricted than the syntax of the three classes above and that we consider sets with one dependency at the time, we have shown that, even for a simple single constraint, the boundary between tractable and intractable cases depends on the constraints that we put in β, which is the counterpart of the chosen interval relation. A possible promising development of

[3] Notice that in the terminology of database repairing we have that boldfaced lower-case letters (e.g. $\mathbf{x}, \mathbf{y}, \ldots$) denote sets of attributes while lower case letters (e.g. $x, y, \ldots$) denote a single attribute.

our work is the classification of the repair checking problem with respect to sets of ITFDs together with possible generalizations of them.

10 Conclusions

In this paper we discussed the complexity of deriving approximate interval-based functional dependencies with respect to the size of the given database instance. Complexities for such an operation are different according to the Allen's interval relation considered for the dependency. Compass structures have been used to analyze such complexities. Different *ITFD-Approx* problems are in different complexity classes, according to the considered Allen's relation, and range from P to NP-complete. Such results make approximate ITFDs quite different from both approximate FDs and point-based TFDs, as their related data complexity remains polynomial. As a future research, we plan to apply and refine our (tractable) algorithms to the analysis of clinical data, as ITFD-related knowledge needs to be deeply assessed according to the specific domain. Heuristics for the NP-complete problems will be studied, even considering the specific needs of the medical domain. From a more theoretical point of view we intend to study extensions of ITFDs (and TFDs in general) considering non-equivalence relations between atemporal attributes. As an example, consider the dependency "the same therapy type for the same patient and the same drug is increasing the dose over time".

Compliance with Ethical Standards

Pietro Sala is founded by the Department of Computer Science and the Department of Public Health and Community Medicine, Pharmacology section both of the University of Verona, in the context of the project "An interval-based approach for data analysis and workflow modelling in medical domains". Conflict of interest: The authors declare that they have no conflict of interest.

References

1. Foto N. Afrati and Phokion G. Kolaitis. Repair checking in inconsistent databases: algorithms and complexity. In Fagin [15], pages 31–41.
2. James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983.
3. Claudio Bettini, Sushil G. Jajodia, and Sean X. Wang. *Time Granularities in Databases, Data Mining and Temporal Reasoning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000.
4. Davide Bresolin, Valentin Goranko, Angelo Montanari, and Pietro Sala. Complete and terminating tableau for the logic of proper subinterval structures over dense orderings. *Electr. Notes Theor. Comput. Sci.*, 231:131–151, 2009.
5. Davide Bresolin, Valentin Goranko, Angelo Montanari, and Pietro Sala. Tableaux for logics of subinterval structures over dense orderings. *J. Log. Comput.*, 20(1):133–166, 2010.
6. Jan Chomicki and Jerzy Marcinkowski. Minimal-change integrity maintenance using tuple deletions. *Inf. Comput.*, 197(1-2):90–121, 2005.
7. Edgar Frank Codd. Normalized data structure: A brief tutorial. In E. F. Codd and A. L. Dean, editors, *SIGFIDET Workshop*, pages 1–17. ACM, 1971.
8. Carlo Combi, Elpida Keravnou-Papailiou, and Yuval Shahar. *Temporal Information Systems in Medicine*. Springer-Verlag New York, Inc., New York, NY, USA, 2010.

9. Carlo Combi, Matteo Mantovani, Alberto Sabaini, Pietro Sala, Francesco Amaddeo, Ugo Moretti, and Giuseppe Pozzi. Mining approximate temporal functional dependencies with pure temporal grouping in clinical databases. *Computers in Biology and Medicine*, 62:306 – 324, 2015.
10. Carlo Combi, Angelo Montanari, and Giuseppe Pozzi. The T4SQL Temporal Query Language. In Mário J. Silva, Alberto H. F. Laender, Ricardo A. Baeza-Yates, Deborah L. McGuinness, Bjørn Olstad, Øystein Haug Olsen, and André O. Falcão, editors, *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, (CIKM), Lisbon, Portugal, November 6-10, 2007*, pages 193–202. ACM, 2007.
11. Carlo Combi, Angelo Montanari, and Pietro Sala. A uniform framework for temporal functional dependencies with multiple granularities. In Dieter Pfoser, Yufei Tao, Kyriakos Mouratidis, Mario A. Nascimento, Mohamed F. Mokbel, Shashi Shekhar, and Yan Huang, editors, *Advances in Spatial and Temporal Databases - 12th International Symposium, (SSTD) Minneapolis, MN, USA, August 24-26, 2011, Proceedings*, volume 6849 of *Lecture Notes in Computer Science*, pages 404–421. Springer, 2011.
12. Carlo Combi, Paolo Parise, Pietro Sala, and Giuseppe Pozzi. Mining approximate temporal functional dependencies based on pure temporal grouping. In Wei Ding, Takashi Washio, Hui Xiong, George Karypis, Bhavani M. Thuraisingham, Diane J. Cook, and Xindong Wu, editors, *13th IEEE International Conference on Data Mining Workshops, ICDM Workshops, TX, USA, December 7-10, 2013*, pages 258–265. IEEE Computer Society, 2013.
13. Carlo Combi and Pietro Sala. Temporal functional dependencies based on interval relations. In Carlo Combi, Martin Leucker, and Frank Wolter, editors, *Eighteenth International Symposium on Temporal Representation and Reasoning, (TIME), Lübeck , Germany, September 12-14*, pages 23–30. IEEE, 2011.
14. Carlo Combi and Pietro Sala. Interval-based temporal functional dependencies: specification and verification. *Annals of Mathematics and Artificial Intelligence*, 71(1-3):85–130, 2014.
15. Ronald Fagin, editor. *Database Theory - ICDT 2009, 12th International Conference, St. Petersburg, Russia, March 23-25, 2009, Proceedings*, volume 361 of *ACM International Conference Proceeding Series*. ACM, 2009.
16. Gaëlle Fontaine. Why is it hard to obtain a dichotomy for consistent query answering? In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, (LICS), New Orleans, LA, USA, June 25-28, 2013*, pages 550–559. IEEE Computer Society, 2013.
17. M. R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some simplified np-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976.
18. Ykä Huhtala, Juha Kärkkäinen, Pasi Porkka, and Hannu Toivonen. Efficient discovery of functional and approximate dependencies using partitions. In Susan Darling Urban and Elisa Bertino, editors, *Proceedings of the Fourteenth International Conference on Data Engineering (ICDE), Orlando, Florida, USA, February 23-27, 1998*, pages 392–401. IEEE Computer Society, 1998.
19. Ykä Huhtala, Juha Kärkkäinen, Pasi Porkka, and Hannu Toivonen. Tane: An efficient algorithm for discovering functional and approximate dependencies. *Comput. J.*, 42(2):100–111, 1999.
20. Christian S. Jensen and Richard T. Snodgrass. Temporal database. In Liu and Özsu [24], pages 2957–2960.
21. Christian S. Jensen, Richard T. Snodgrass, and Michael D. Soo. Extending existing dependency theory to temporal databases. *IEEE Trans. Knowl. Data Eng.*, 8(4):563–582, 1996.
22. Jyrki Kivinen and Heikki Mannila. Approximate inference of functional dependencies from relations. *Theoretical Computer Science*, 149(1):129 – 149, 1995.
23. Solmaz Kolahi and Laks V. S. Lakshmanan. On approximating optimum repairs for functional dependency violations. In Fagin [15], pages 53–62.
24. Ling Liu and M. Tamer Özsu, editors. *Encyclopedia of Database Systems*. Springer US, 2009.
25. Heikki Mannila and Kari-Jouko Räihä. On the complexity of inferring functional dependencies. *Discrete Appl. Math.*, 40(2):237–243, December 1992.
26. Heikki Mannila and Kari-Jouko Rih. Algorithms for inferring functional dependencies from relations. *Data And Knowledge Engineering*, 12(1):83 – 99, 1994.
27. Jerzy Marcinkowski and Jakub Michaliszyn. The undecidability of the logic of subintervals. *Fundam. Inform.*, 131(2):217–240, 2014.

28. Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.
29. Pietro Sala. Approximate interval-based temporal dependencies: The complexity landscape. In Amedeo Cesta, Carlo Combi, and François Laroussinie, editors, *21st International Symposium on Temporal Representation and Reasoning, TIME 2014, Verona, Italy, September 8-10, 2014*, pages 69–78. IEEE Computer Society, 2014.
30. Yde Venema. A modal logic for chopping intervals. *J. Log. Comput.*, 1(4):453–476, 1991.
31. Victor Vianu. Dynamic functional dependencies and database aging. *J. ACM*, 34(1):28–59, 1987.
32. Xiaoyang Sean Wang, Claudio Bettini, Alexander Brodsky, and Sushil Jajodia. Logical design for temporal databases with multiple granularities. *ACM Trans. Database Syst.*, 22(2):115–170, 1997.
33. Jef Wijsen. Temporal fds on complex objects. *ACM Trans. Database Syst.*, 24(1):127–176, 1999.
34. Jef Wijsen. Temporal dependencies. In Liu and Özsu [24], pages 2960–2966.
35. Jef Wijsen. Temporal integrity constraints. In Liu and Özsu [24], pages 2976–2982.