Representing and reasoning about temporal granularities

Carlo Combi (corresponding author)

Department of Computer Science, University of Verona,

Ca' Vignal 2 strada le Grazie 15

I - 37134 Verona - VR - Italy

phone: ++39 045 802 7985; fax: ++39 045 802 7068

E-mail: `combi@sci.univr.it`

Massimo Franceschet

Department of Sciences, University 'G. d'Annunzio' of Pescara, Italy

E-mail: `francesc@sci.unich.it`

Adriano Peron

Department of Physical Sciences, University 'Federico II' of Napoli, Italy

E-mail: `peron@na.infn.it`

Abstract

In this paper, we propose a new logical approach to represent and to reason about different time granularities. We identify a time granularity as an infinite sequence of time points properly labelled with proposition symbols marking the starting and ending points of the corresponding granules, and we symbolically model sets of granularities by means of linear time logic formulas. Some real-world granularities are provided, from a clinical domain and from the Gregorian Calendar, to motivate and exemplify our approach. Different formulas are introduced, which represent relations between different granularities. The proposed framework permits to algorithmically solve the consistency, the equivalence, and the classification problems in a uniform way, by reducing them to the validity problem for the considered linear time logic.

1. Introduction

1.1 Context and Motivation

Any *time granularity* can be viewed as the partitioning of a temporal domain in groups of elements, where each group is perceived as an indivisible unit (a granule). The description of a fact can use these granules to provide it with a temporal qualification, at the appropriate abstraction level. However, adding the concept of time granularity to a formalism does not merely mean that one can use different temporal units to represent temporal quantities in a unique flat model, but it involves semantic issues related to the problem of assigning a proper meaning to the association of statements with the different temporal domains of a layered model and of switching from one domain to a coarser/finer one. The ability of providing and relating temporal representations at different 'grain levels' of the same reality is an important research theme in computer science. In particular, it is a major requirement for formal specifications, temporal databases, data mining, problem solving, and natural language understanding.

- As for *formal specifications*, there exists a large class of reactive systems whose components have dynamic behavior regulated by very different time constants (granular reactive systems). A good specification language must enable one to specify and verify the behavior of the components of a granular reactive system and their interactions in a simple and intuitively clear way [13, 19, 20, 29, 40, 46, 47, 48, 49].

- With regard to *temporal databases*, the common way to represent temporal information is to timestamp either attributes (*attribute timestamping*) or tuples/objects (*tuple-timestamping*). Timestamping is performed taking time values over some fixed granularity. However, it may happen that differently-grained timestamps have to be associated with data, for example, when information is collected from different sources which are not under the same control. Furthermore, users and applications may also require the flexibility of viewing and querying the temporal information stored in the database in terms of different granularities. To guarantee consistency either the data must be converted into a uniform representation that is independent of time granularity or temporal operations must be generalized to cope with data associated with different temporal domains. In both cases, a precise semantics for time granularity is needed [3, 12, 18, 26, 37, 38, 45, 51, 54, 58, 59].

- With regard to *data mining*, a huge amount of data is collected every day in the form of event-time sequences. These sequences represent valuable sources of information, not only for what is explicitly registered, but also for deriving implicit information and predicting the future behavior of the process that we are monitoring.

The latter activity requires an analysis of the frequency of certain events, the discovery of their regularity, and the identification of sets of events that are linked by particular temporal relationships. Such frequencies, regularity, and relationships are very often expressed in terms of multiple granularities, and thus analysis and discovery tools must be able to deal with these granularities [1, 4, 6, 25, 42].

- With regard to *problem solving*, several problems in scheduling, planning, and diagnosis can be formulated as temporal constraint satisfaction problems, often involving multiple time granularities. In a temporal constraint satisfaction problem, variables are used to represent event occurrences and constraints are used to represent their granular temporal relationships [8, 5, 21, 28, 36, 39, 50, 53, 55].
- Finally, shifts in the temporal perspective occur very often in natural language communication, and thus the ability of supporting and relating a variety of temporal models, at different grain sizes, is a relevant feature for the task of *natural language understanding* [11, 30, 34].

A further distinction we have to introduce is between the *representation and reasoning on time granularities* and the *representation and reasoning on facts/statements associated with times specified at different granularities*. The requirements for reasoning on facts at different levels of granularity are often related and specific to the different research areas mentioned above: e.g., supporting for different time granularities for database query languages [26], supporting the specification of real-time systems with different granularities [20], providing algorithms for pattern discovery on time series with several time units [4]. Nevertheless, the need for formalisms allowing the specification and the reasoning on granularities is common to all the mentioned research areas and originated several different proposals [9, 30, 35, 44, 51, 52, 60].

More specifically, most approaches proposed in the literature for representing and reasoning about time granularity can be classified into algebraic approaches and logical ones. In the *algebraic* (or *operational*) framework, a *bottom granularity* is assumed, and a finite set of *calendar operators* are exploited to create new granularities by suitably manipulating other granularities [9, 30, 51, 52]. In the *logical* (or *descriptive*) framework for time granularity, the different granularities and their interconnections are represented by means of mathematical structures called layered structures, consisting of a possibly infinite set of related differently-grained temporal domains. Suitable operators make it possible to move within a given temporal domain and across temporal domains. Logical formulas allow one to specify properties involving different time granularities in a single formula by mixing such operators [33, 44, 46, 48, 49].

Algebraic and logical frameworks stem from different research areas calling for different focuses. For instance, in the database context, where the algebraic framework is usually adopted, granule conversion plays a major role because it allows the user to view the temporal information contained in the database in terms of different granularities, while in the context of verification, where logical frameworks have been proposed, decision procedures are unavoidable to automatically validate the system (for example, to establish whether two different representations define the same granularity). Abstracting away from the research areas of the two frameworks, it is possible to identify their main limitations and advantages. The main advantage of the algebraic framework is its naturalness: by applying user-friendly operations to existing standard granularities like 'days', 'weeks', and 'months', a quite large class of new granularities, like 'business weeks', 'business months', and 'years since 2000', can be easily generated. The major weakness of the algebraic framework is that reasoning methods basically reduce to granule conversions and semantic translations of statements. Little attention has received the investigation of algorithms to check whether some meaningful relation holds between granularities (e.g., to verify whether the granularity G_1 is finer than granularity G_2 or G_1 is equivalent to G_2). Moreover, only a finite number of time granularities can be represented. On the contrary, reasoning methods have been extensively investigated in the logical framework, where both a finite and an infinite number of time granularities can be dealt with. Theorem provers make it possible to verify whether a granular requirement is consistent (i.e., specifies a well-defined granularity), while model checkers allow one to check whether a granular property is satisfied in a particular structure. To allow such computational properties, however, some limitations have to be introduced for the involved granularities, assuming, for example, some form of regularity of the sizes of the granules.

With respect to this scenario, several efforts are needed to have a more comprehensive approach, which maintains both the naturalness of the algebraic framework and the reasoning methods developed for the logical framework, allowing its usage in different research areas and a deep comparison of the proposals currently existing for the specification of time granularities.

In general, in order to represent and to reason about time granularity, any formalism should satisfy the following requirements:

- *Expressiveness*. The class of granularities represented in the formalism should be large enough to be of practical use.

- *Effectiveness*. The formalism should provide algorithms to reason about different time granularities. In particular, it should provide an effective solution to the well-known problems of *consistency*, *equivalence* and *classification*.

  - The *consistency problem* is the problem of deciding whether a granularity representation is well-defined. The algorithmic solution of the consistency problem is important to avoid the definition of inconsistent granularities that may produce unexpected failures in the system.

  – The *equivalence problem* is the problem of deciding whether two different representations define the same granularity. The decidability of the equivalence problem implies the possibility of effectively testing the semantic equivalence of two different time granularity representations, making it possible to use the smallest and most tractable one.

  – The *classification problem* is the problem of deciding whether a natural number n, representing a time point, belongs to a granule of a given granularity. The classification problem is strictly related to the granule conversion problem which allows one to relate granules of a given granularity to granules of another one.

- *Compactness*. The formalism should exploit regularities exhibited by the considered granularities to make their representations as compact as possible.

1.2 Focus and goals of the paper

The paper deals with a first attempt to propose an approach for the specification of temporal granularities, taking into account both the algebraic framework and the logical one. The basic idea is to assume the standard definition of granularity proposed by Bettini and colleagues (see, for example, [8]) and extensively adopted by algebraic approaches proposed for temporal databases, temporal data mining, and problem solving [3, 4, 6, 7, 16, 18, 52], and to develop on top of it a logical approach based on a linear temporal logic, also considering how (and which of) the main algebraic operators can be expressed as logical formulas.

More precisely, in this paper, we propose an original logical approach to represent and to reason about different time granularities, which overcomes some limitations of logical and algebraic frameworks. We identify a time granularity with a discrete linear time structure properly labelled with proposition symbols marking the starting and ending points of the corresponding granules. We make use of a linear time logic, interpreted over labelled linear time structures, to model possibly infinite sets of time granularities. Any linear time formula is associated with a set of labelled linear time structures satisfying the formula (the set of models of the formula). Since any properly labelled linear time structure identifies a time granularity, we may model possibly infinite sets of time granularities by means of well-defined linear time formulas. Moreover, a single sequence may identify a finite number of different granularities (a calendar) by using a different couple of marking proposition symbols for any granularity. Hence, well-defined linear time formulas may model possibly infinite sets of calendars as well. The proposed approach permits to model a large set of regular granularities and to algorithmically solve the consistency, the equivalence, and the classification problems in a uniform way by reducing them to the validity problem for the considered linear time logic, which is known to be decidable in polynomial space.

1.3 Structure of the paper

The rest of the paper is organized as follows. In Section 2 we describe in some detail the main approaches to the problem of representing and reasoning about time granularity. In Section 3 we present some real-world motivating examples. In Section 4 we propose our logical approach to represent and to reason about time granularity and discuss both expressiveness and computational features of our proposal. In Section 5 we summarize the comparison of our work with related ones and, finally, in Section 6 we sketch some concluding remarks and outline future work.

2 Related work

In this section, we describe in some detail the main proposals for the algebraic framework and the logical one present in the literature, for representing and reasoning about time granularity; then, we introduce some recent approaches which originated the *string-based* framework.

2.1 Algebraic Framework

In the algebraic framework, new granularities are generated from existing ones, assuming a bottom granularity, through a finite set of calendar operators. A granularity is hence identified by an algebraic expression. In the algebraic framework, algorithms are provided to perform granule conversions, that is, to convert the granules in one granularity to those in another granularity, and to perform semantic conversion of statements associated to different granularities. The algebraic approach to time granularity has been mostly applied in the fields of databases, data mining, and temporal reasoning. Algebraic approaches for time granularities have been proposed by Foster, Leban, and McDonald [30], by Niezette and Stevenne [51], by Bettini and De Sibi [9], and by Ning, Jajodia, and Wang [52]. Foster, Leban, and McDonald propose the *temporal interval collection formalism*. A collection is a structured set of intervals, where the order of the collection gives a measure of the structure depth: a collection of order 1 is an ordered list of intervals, and a collection of order n, with $n > 1$, is an ordered list of collections of order $n - 1$. Each interval denotes a set of contiguous moments of time. To manipulate collections, dicing and slicing operators are used. The former allow one to divide each interval of a collection into another collection, while the latter provide means to select intervals from collections. For instance, the application of the dicing operator `Week` : `during` : `January1998` divides the interval corresponding to `January1998` into the intervals corresponding to the weeks that are fully contained in the month. Moreover, the application of the slicing operator $[1, -1]/\mathtt{Week} : \mathtt{during} : \mathtt{January1998}$ selects the first and the last week from those identified by the dicing operator above.

Niezette and Stevenne introduce a similar formalism, called the *slice formalism*. A slice has the form $\sum_{i=1}^{n} O_i.C_i > D$, where the elements of the sum identify the starting points of the intervals, being C_i a symbol denoting a calendar (i.e., a periodic infinite set of consecutive intervals) and O_i either a set of natural numbers or the keyword `all`, and D their duration. For instance, $\mathtt{all}.\mathtt{Year} + \{3, 5\}.\mathtt{Month} + \{2\}.\mathtt{Day} > 5.\mathtt{Day}$ denotes a set of intervals corresponding to the days 2 - 6 of March and May of each year: i.e, any interval lasts 5 days, as specified by the duration $5.\mathtt{Day}$, and may start either on March 2 or on May 2 of each year, as specified by the first part of the expression.

Bettini and De Sibi show that both slice and collection formalisms can capture the set of finite no-gap granularities and that of infinite periodical no-gap granularities. Intuitively, a finite no-gap granularity is composed by a finite number of granules, i.e., intervals on the basic time line; infinite periodical no-gap granularities are composed by an infinite number of granules, i.e., intervals on the basic time line, with a periodical behaviour with respect to their extensions. Moreover, the collection formalism is extended to capture also both gap and quasi-periodical granularities: gap granularities, which are not expressible by the slice and collection formalisms, have granules which are composed by a set of non contiguous time points of the basic time line (e.g., `Business Months`, defined as the set of business days in a month, is a gap granularity on the time line of days); quasi-periodical granularities behave as periodical granularities, except for a finite number of spans of time, where they have an anomalous behaviour. The extended collection formalism allows one to express also infinite bi-periodical granularities, which are represented by two sets of repeating granules, the first repeating from a maximum time point towards $-\infty$ and the second one repeating from a minimum time point towards $+\infty$.

Finally, Ning, Jajodia, and Wang introduce a *calendar algebra* consisting of a finite set of parametric calendar operations that can be classified into grouping-oriented operations and granule-oriented operations. The former operations group certain granules of a granularity together to form the granules of a new granularity. For instance, a typical group-oriented operation is $\mathtt{Group}_n(G)$ that generates a new granularity G' by partitioning the granules of G into groups containing n granules and making each group a granule of the resulting granularity. The granule-oriented operations do not change the granules of a granularity, but rather select which granules should remain in the new granularity. A typical granule-oriented operation is $\mathtt{Subset}_m^n(G)$ that generates a new granularity G' by taking all the granules of G between m and n. By the calendar algebra, all the finite and the infinite periodical granularities can be represented. Some syntactic restrictions are introduced in the usage of the algebraic operations: these restrictions facilitate the calendar algebraic operations, without decreasing the expressiveness of the algebra. Three layers are identified in the calendar algebra, according to the operators used for defining new granularities: layer 1 is composed by the basic granularity and by all the granularities obtained without introducing gaps within granules and without using operators which produce finite granularities (in this layer, only the grouping-oriented basic operations are allowed); layer 2 is mainly composed by granularities obtained by applying subset and selecting operations on granularities of layer 1 (i.e., only granule-oriented operations are allowed); layer 3 contains granularities obtained from granularities of layers 1 and 2, suitably combined by some specific grouping-oriented operations. Moreover, all the three layers can contain granularities obtained by other granularities of the same layer, using suitable algebraic operators. Being the calendar algebra proposed in the context of temporal databases, algorithms are then provided by the authors to support different kinds of granule conversions: to this regard, also the granule conversion problem has simpler solutions with the above syntactic restrictions, being computations of (up and down) granule conversions based on the features of the involved granularities, with respect to the layer they belong to.

2.2 Logical Framework

In the logical framework for time granularity, mathematical structures, i.e., layered structures, represent the different granularities and their interconnections. A *layered structure* consists of a possibly infinite set of related differently-grained temporal domains. Such a structure identifies the relevant temporal domains and defines the relations between time points belonging to different domains. Suitable operators make it possible to move horizontally *within* a given temporal domain (displacement operators), and to move vertically *across* temporal domains (projection operators). These operators recall the slicing and dicing operators of the collection formalism. Both classical and temporal logics can be interpreted over the layered structure. Logical formulas allow one to specify properties involving different time granularities in a single formula by mixing displacement and projection operators. Algorithms are provided to verify whether a given formula is consistent (satisfiability problem) as well as to check whether a given formula is satisfied in a particular structure (model checking problem).

The logical approach to represent time granularity has been mostly applied in the field of formal specification and verification of concurrent systems. A logical approach to represent and reason about time granularity, based on a many-level view of temporal structures, has been proposed by Montanari in [44], and further investigated by Franceschet, Montanari, Peron, and Policriti in [31, 46, 49]. In the proposed approach, the *flat* temporal structure of standard temporal logics is replaced by a *layered* temporal universe consisting of a possibly infinite set of related differently-grained temporal domains. In [44], a metric and layered temporal logic (MLTL) for time granularity has been proposed. It is provided with temporal operators of displacement and projection, which can be arbitrarily combined, and it is interpreted over layered structures. However, only a sound axiomatic system is given, and no decidability result is proved for MLTL.

Layered structures with exactly $n \geq 1$ temporal domains such that each time point can be refined into $k \geq 2$ time points of the immediately finer temporal domain, if any, are called k-refinable n-layered structures (n-LSs for short). They have been investigated in [49], where a classical second-order language, with second-order quantification restricted to monadic predicates, has been interpreted over them. The language includes a total ordering $<$ and k projection functions $\downarrow_0, \ldots, \downarrow_{k-1}$ over the layered temporal universe such that, for every point x, $\downarrow_0(x), \ldots, \downarrow_{k-1}(x)$ are the k elements of the immediately finer temporal domain, if any, into which x is refined. The satisfiability problem for the monadic second-order language over n-LSs has been proved to be decidable by using a reduction to the emptiness problem for Büchi sequence automata. Unfortunately, the decision procedure has a nonelementary complexity. Layered structures with an infinite number of temporal domains, ω-layered structures, have been studied in [46]. In particular, the authors investigated *k-refinable upward unbounded layered structures* (UULSs), that is, ω-layered structures consisting of a finest temporal domain together with an infinite number of coarser and coarser domains, and *k-refinable downward unbounded layered structures* (DULSs), that is, ω-layered structures consisting of a coarsest domain together with an infinite number of finer and finer domains. A classical monadic second-order language, including a total ordering $<$ and k projection functions $\downarrow_0, \ldots, \downarrow_{k-1}$, has been interpreted over both UULSs and DULSs. The decidability of the monadic second-order theories of UULSs and DULSs has been proved by reducing the satisfiability problem to the emptiness problem for systolic and Rabin tree automata, respectively. In both cases the decision procedure has a nonelementary complexity. Expressively complete and elementarily decidable temporal logic and automata counterparts of the second-order theories of n-LSs, DULSs and UULSs have been proposed in [31]. Moreover, the monadic language for granularity has been extended with meaningful predicates like the equi-level, constraining two time points to belong to the same layer of a layered structure, and the equi-column, constraining two time points to belong to the same column of a layered structure, and the decidability problems of the resulting theories over layered structures have been explored [31].

2.3 String-based Framework

A recent original approach to represent and to reason about time granularity has been proposed by Wijsen [60] and later refined by Dal Lago, Montanari, and Puppis [22, 23, 24]. Wijsen models infinite granularities as infinite strings over a suitable finite alphabet. The resulting string-based model is then used to formally state and solve the problems of granularity equivalence and minimality. This formalism does not fulfill the requirement of compactness: the representation of a granularity can be very long whenever the granularity is periodic with a long prefix or period (as in the case of the Gregorian Calendar).

Dal Lago and Montanari [22] give an automata-theoretic counterpart of the string-based model devised by Wijsen. They propose *single-string automata*, a variant of deterministic Büchi automata accepting a single infinite string, to represent time granularities. Furthermore, they show that regularities of modeled granularities can be naturally expressed by extending single-string automata with counters. This extension makes the structure of the automata more compact, and it allows one to efficiently deal with those granularities which have a quasi-periodic structure. In [23], the authors prove that single-string automata provide an efficient solution to the fundamental problems of equivalence and classification. Moreover, they argue how single-string automata can be used not only as a specification formalism for time granularities, but also as a low-level formalism into which high-level time granularity specifications can be mapped. For instance, expressions of Calendar Algebra [52] can be efficiently mapped into equivalent single-string automata.

Finally, Dal Lago, Montanari, and Puppis [24] focus on two kinds of optimization problems for automata-based representations of time granularities, namely, computing the smallest representation and computing the most tractable representation, that is, the one in which granule conversion algorithms run fastest. They show how to efficiently compute complexity optimal automata from smaller ones in a bottom-up fashion.

3. Motivating examples

To motivate the need of managing different granularities, we will focus on two examples of different nature. The first example comes from clinical medicine and deals with the definition of specific granularities related to therapy plans. Intuitively, therapy plans can be considered as the calendar according to which it is possible to properly observe the evolution of the patient's state. We consider here chemotherapies for oncological patients, a topic which has been extensively considered by the clinical research and that is precisely described and recommended in several clinical practice guidelines, the physicians follow during the daily routine. Chemotherapies are usually intensive and potentially have several side effects for the patients: it is, then, useful to observe how a treated patient reacts during the different phases of the therapy.

In general, oncology patients undergo several chemotherapy cycles: each cycle can include the administration of several drugs, which the patient has to assume according to a specific temporal pattern. Within each cycle, the temporal administration of each drug is usually predefined; a cycle can be repeated several times. As an example, consider the following chemotherapy recommendation [41]:

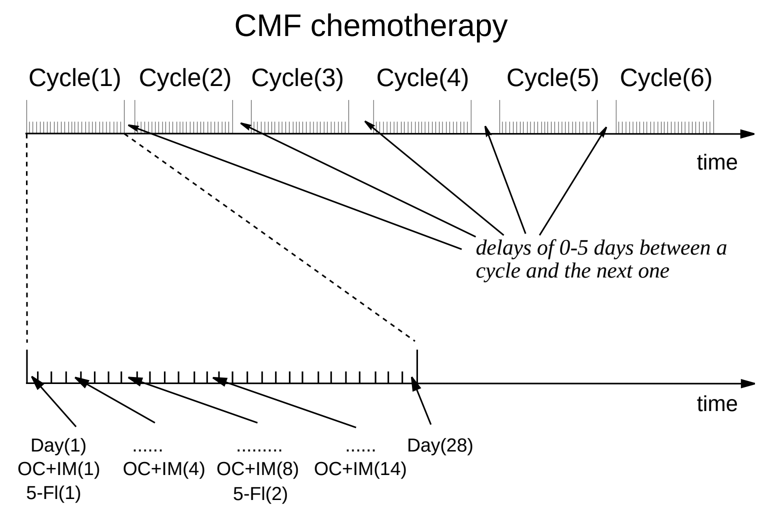

Figure 1. Granularities involved in a chemotherapy treatment.

"*The recommended CMF*[1] *regimen consists of 14 days of oral cyclophosphamide with intravenous methotrexate, and 5-fluorouracil on days 1 and 8. This is repeated every 28 days for 6 cycles*"

Moreover, it may happen that the beginning of a cycle is delayed a few days, due to patient's blood analysis results. Figure 1 provides a graphical representation of the recommended CMF regimen.

According to this scenario, we can easily identify some requirements related to the definition of useful granularity systems:

1. *Definition of therapy-related granularities.* These granularities should be suitably specified for different patients, according to the moment at which they start a given chemotherapy.

2. *Definition of granularities having some degree of uncertainty.* There is, indeed, the need of representing the fact that two consecutive cycles may be separated by some days, due to the patient's conditions.

3. *Verification of consistency between an assigned therapy and the recommended regimen.* Given a therapy assigned to a patient with the specification of days and corresponding drug assumptions, it is important to be able to determine whether the therapy is consistent with the recommended regimen.

4. *Assignment of a therapy according to the recommended regimen and to other granularity-related constraints.* It could be necessary, for organizational reasons, to avoid that some specific drug administrations happen during the weekend: for example, in specifying a CMF therapy, we could avoid that the administration of 5-fluorouracil is on Sundays or on December 25.

[1]CMF stands for the chemotherapy based on the drugs Cyclophosphamide, Methotrexate, and 5-Fluorouracil.

As mentioned in the last point, we cannot disregard, when dealing with time granularities, the representation of the Gregorian Calendar: the second example is, thus, the typical example in the field of time granularity. The Gregorian Calendar has a bottom granularity representing the days. Days group into weeks: a week consists of 7 consecutive days. Days group into months as well. The first month is January and consists of 31 days, the second is February and contains either 28 days or, during leap years, 29 days, the third is March consisting of 31 days and so on. Note that weeks and months may overlap. A year groups twelve months, starting from January. Leap years are those multiple of 4, excluding years multiple of 100, but including years multiple of 400. For instance, 1900 is not leap, but 2000 is leap.

4. A logical approach to represent and reason about calendars

In this section, we propose our logical approach to represent and reason about different time granularities and provide some examples of real-world temporal granularities, represented according to our approach.

4.1. Representing time granularity

We model time granularities according to the following standard definition.

Definition 4.1 *A* granularity *is a mapping $G : \mathbb{N} \rightarrow 2^{\mathbb{N}}$ such that:*

1. *for all $i < j$, for any $n \in G(i)$ and $m \in G(j)$, $n < m$;*
2. *for all $i < j$, if $G(j) \neq \emptyset$, then $G(i) \neq \emptyset$.*

Following the classical definition given in [8], the domain of a granularity G is called *index set* and an element of the codomain of G is called a *granule*. The first condition states that granules in a granularity do not overlap and that their index order is the same as their time domain order. The second condition states that the subset of the index set that maps to nonempty granules forms an initial segment.

The definition of granularity above specializes the definition given in [8], assuming that both the index set and the time domain (i.e., the domain of granules) are the linear discrete domain $(\mathbb{N}, <)$. The choice of linear temporal logics for expressing granularities forces the use of linear discrete domains. For the sake of simplicity, we choose a linear time domain with a least element (instead of integers as in [8]). Since we shall describe granularities by means of Past Propositional Linear Time Logic, our approach could be straightforwardly adapted to deal also with granularities having integers as index set.

A granularity G is said:

- *externally continuous*, if there are no gaps between nonempty granules of G;
- *internally continuous*, if there are no gaps inside the granules of G;
- *continuous*, if it is both internally and externally continuous;
- *total*, if the granules of G cover all the time domain;
- *uniform*, if all the nonempty granules of G have the same cardinality.

With reference to standard calendar granularities, the granularity `Day` is continuous, uniform and total. Granularities `Month` and `Year` are not uniform, `YearsSince2000` is not total, `BusinessWeek` is not externally continuous, and `BusinessMonth` is not internally continuous. Following [52], a number of meaningful relationships can be established between pairs of granularities G_1 and G_2, as detailed in the following.

- $\mathbf{FinerThan}(G_1, G_2)$ holds if each granule of G_1 is contained into a granule of G_2, that is, if, for each index i, there exists an index j such that $G_1(i) \subseteq G_2(j)$. For instance, `Day` is finer than `Month`, but `Week` is not finer than `Month`, since a week can spread over two consecutive months.
- $\mathbf{SubGranularity}(G_1, G_2)$ holds if each granule of G_1 is equal to a granule of G_2, that is, if, for each index i, there exists an index j, such that $G_1(i) = G_2(j)$. For example, the granularity `Sunday` is a subgranularity of `Day`.
- $\mathbf{GranuleRefinement}(G_1, G_2)$ holds if each granule of G_1 is contained into the corresponding granule of G_2, that is, if, for every index i, $G_1(i) \subseteq G_2(i)$. As an instance, we have that `Weekend` is a granule refinement of `Week`.
- $\mathbf{Group}(G_1, G_2)$ holds if each granule of G_2 is obtained by grouping an arbitrary number of consecutive granules of G_1, that is, if for any i there exists a set S (which is either empty or has the form $\{j, j+1, \ldots, j+k\}$, with $j, k \geq 0$) such that $G_2(i) = \bigcup_{l \in S} G_1(l)$.

  More specialized forms of grouping can be defined. Given $p \geq 1$, the relation $\mathbf{Group_p}(G_1, G_2)$ holds if each granule of G_2 is obtained by grouping p consecutive granules of G_1, that is, if, for each index i, $G_2(i) = \bigcup_{j=0}^{p-1} G_1(i \cdot p + j)$. More generally, given $p \geq 1$ and $q \geq 0$, the relation $\mathbf{Group_p Skip_q}(G_1, G_2)$ holds if each granule of G_2 is obtained by grouping p consecutive granules of G_1 and by skipping the following q consecutive granules of G_1, that is, if, for each index i, $G_2(i) = \bigcup_{j=0}^{p-1} G_1(i \cdot (p + q) + j)$. Similarly, the relation $\mathbf{Skip_q Group_p}(G_1, G_2)$ holds if each granule of G_2 is obtained by skipping q consecutive granules of G_1 and by grouping the following p consecutive granules of G_1. Note that $\mathbf{Group_p Skip_0}(G_1, G_2) = \mathbf{Skip_0 Group_p}(G_1, G_2) = \mathbf{Group_p}(G_1, G_2)$. Finally, $\mathbf{EqualGroup}(G_1, G_2)$ holds if each granule of G_2 is obtained by grouping the same (not fixed) number of consecutive granules of G_1. We have that $\mathbf{Group}(\mathtt{Day}, \mathtt{Week})$ holds, and, in particular, $\mathbf{Group_7}(\mathtt{Day}, \mathtt{Week})$ holds. Moreover, $\mathbf{Group_5 Skip_2}(\mathtt{Day}, \mathtt{BusinessWeek})$, and $\mathbf{Skip_5 Group_2}(\mathtt{Day}, \mathtt{Weekend})$ hold.

- Given $p \geq 0$, $\mathbf{Shift_p}(G_1, G_2)$ holds if the granules of G_2 are obtained by backward shifting of p positions the granules of G_1, that is, for each index i, $G_2(i) = G_1(i + p)$. Moreover, $\mathbf{Shift}(G_1, G_2)$ holds if the granules of G_2 are obtained by backward shifting of an arbitrary number of positions the granules of G_1, that is, if there exists $p \geq 0$ such that, for each index i, $G_2(i) = G_1(i + p)$. For example, $\mathbf{Shift_{2000}}(\mathtt{Year}, \mathtt{YearSince2000})$ holds.

Note that, in general, the group operations do not preserve internal continuity. Indeed, if we group two non-adjacent granules of an internally continuous granularity, we get a non-convex granule, i.e. a granule consisting of a set of non-contiguous time points, and the resulting granularity is hence no more internally continuous.

In the following, we propose a logical framework to model *internally continuous* granularities as defined above. We will extend our framework to granularities with gaps inside the granules (i.e., non-internally continuous granularities) in Section 4.2.

Let $\mathcal{G} = \{G_1, \ldots, G_n\}$ be a *finite* set of granularities (we will refer to $\mathcal{G}$ as a *calendar*), and let $\mathcal{P}_{\mathcal{G}} = \{P_{G_i}, Q_{G_i} \mid 1 \leq i \leq n\}$ be a set of proposition symbols associated with the calendar $\mathcal{G}$.

Given an alphabet of proposition symbols $\mathcal{P} \supseteq \mathcal{P}_{\mathcal{G}}$, we shall consider in the following $\mathcal{P}$-labelled (discrete) linear time structures having the form $(\mathbb{N}, <, V)$, where $(\mathbb{N}, <)$ is the set of natural numbers with the usual ordering, and $V : \mathbb{N} \to 2^{\mathcal{P}}$ is a labelling function mapping natural numbers to sets of proposition symbols.

The idea is to identify an internally continuous granularity G with a linear time structure, properly labelled with proposition symbols taken from $\{P_G, Q_G\}$: the starting point of an arbitrary granule of G in the structure is labelled by P_G and the ending point of an arbitrary granule of G in the structure is labelled by Q_G.

Definition 4.2 *A labelled linear time structure* $(\mathbb{N}, <, V)$ *is G-consistent whenever:*

1. *if* $P_G \in V(i)$ *for some* $i \in \mathbb{N}$*, then either* $Q_G \in V(i)$ *or* $Q_G \in V(j)$ *for some* $j > i$ *such that* $P_G \notin V(k)$ *for each* $i < k \leq j$ *and* $Q_G \notin V(k)$ *for each* $i \leq k < j$*;*

2. *if* $Q_G \in V(i)$ *for some* $i \in \mathbb{N}$*, then either* $P_G \in V(i)$ *or* $P_G \in V(j)$ *for some* $j < i$ *such that* $Q_G \notin V(k)$ *for each* $j \leq k < i$ *and* $P_G \notin V(k)$ *for each* $j < k \leq i$*.*

The above conditions say that every point labelled with P_G has to match with a unique point labelled with Q_G, and vice versa. Every G-consistent labelled linear time structure induces an internally continuous granularity G: given a G-consistent labelled linear time structure $\mathcal{M} = (\mathbb{N}, <, V)$, a granule of $\mathcal{M}$ with respect to G is a set $\{n, n+1, \ldots, n+k\}$, for some $n, k \geq 0$, such that $P_G \in V(n)$, $Q_G \in V(n+k)$ and $Q_G \notin V(n+j)$ for all $0 \leq j < k$. The granularity G induced by $\mathcal{M}$ is such that $G(i)$ is the i-th granule of $\mathcal{M}$ with respect to G, if any, and $G(i) = \emptyset$ otherwise. It is easy to check that G is an internally continuous granularity. Conversely, an internally continuous granularity G corresponds to a G-consistent labelled linear time structure: label every starting point of a granule of G with symbol P_G, and every ending point of a granule of G with symbol Q_G. It is not difficult to see that the resulting labelled linear time structure is G-consistent.

Example 4.3 We give some examples of labelled linear time structures that induce well-defined granularities and one example of a labelled linear time structure that does not correspond to a granularity.

- The structure $(\mathbb{N}, <, V)$ such that $V(i) = \{P_G\}$ iff i is even, and $V(i) = \{Q_G\}$ iff i is odd, induces the uniform, total, and continuous granularity G such that $G(i) = \{2 \cdot i, 2 \cdot i + 1\}$ for any $i \in \mathbb{N}$.
- The structure $(\mathbb{N}, <, V)$, represented in Figure 2, such that $V(0) = V(3) = \{P_G, P_H\}$, $V(1) = \{Q_G\}$, $V(2) = \{Q_H\}$, $V(4) = \emptyset$, $V(5) = \{Q_G, Q_H\}$, $V(6) = \{P_H\}$, $V(7) = \{P_G, Q_G\}$ and $V(8) = \{Q_H\}$ ($V(i) = \emptyset$, for $i \geq 9$) induces two granularities G and H:
    1. the non-uniform, non-total, non-externally continuous, and internally continuous granularity G such that $G(0) = \{0, 1\}$, $G(1) = \{3, 4, 5\}$, $G(2) = \{7\}$, and $G(i) = \emptyset$, for any $i \geq 3$;
    2. the uniform, non-total, and continuous granularity H such that $H(0) = \{0, 1, 2\}$, $H(1) = \{3, 4, 5\}$, $H(2) = \{6, 7, 8\}$, and $H(i) = \emptyset$, for any $i \geq 3$.
- The structure $(\mathbb{N}, <, V)$ such that $V(0) = \{P_G\}$, $V(1) = \{Q_G, P_G\}$, $V(2) = \{Q_G\}$ does not induce any granularity, since it is not G-consistent (indeed, the granules $G(0) = \{0, 1\}$ and $G(1) = \{1, 2\}$ intersect).

In the following we show how a set of granularities can be defined in an intensional declarative manner by means of a formula of a propositional linear time logic (instead of defining it extensively as done in Example 4.3). We will use Past Propositional Linear Time Logic (PPLTL for short) [27, 56], interpreted over labelled linear time structures. We proceed by introducing the syntax and the semantics of PPLTL.

Definition 4.4 (*Syntax of* PPLTL)

Formulas of PPLTL *are inductively defined as follows:*

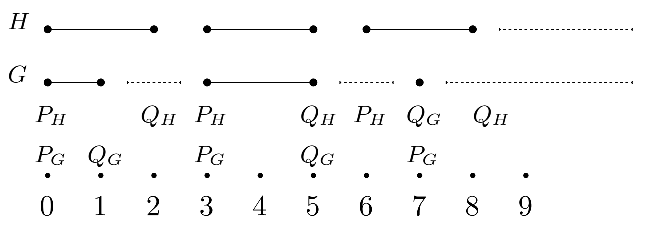

Figure 2. A labelled linear time structure inducing two granularities G and H.

- *any proposition symbol $P \in \mathcal{P}$ is a* PPLTL *formula;*
- *if ϕ and ψ are* PPLTL *formulas, then $\phi \wedge \psi$ and $\neg\phi$ are* PPLTL *formulas;*
- *if ϕ and ψ are* PPLTL *formulas, then $\mathbf{X}\phi$, $\phi\mathbf{U}\psi$, $\mathbf{X^{-1}}\phi$ and $\phi\mathbf{S}\psi$ are* PPLTL *formulas.*

Formulas $\phi \vee \psi$, $\phi \rightarrow \psi$, and $\phi \leftrightarrow \psi$ are defined as $\neg(\neg\phi \wedge \neg\psi)$, $\neg\phi \vee \psi$, and $(\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$, respectively. Moreover, $\mathbf{F}\alpha$ (α will hold in the future), $\mathbf{G}\alpha$ (α will always hold in the future), $\mathbf{P}\alpha$ (α held in the past) and $\mathbf{H}\alpha$ (α always held in the past) are shorthands for, respectively, $\texttt{true}\mathbf{U}\alpha$, $\neg\mathbf{F}\neg\alpha$, $\texttt{true}\mathbf{S}\alpha$ and $\neg\mathbf{P}\neg\alpha$, where $\texttt{true} = P \vee \neg P$, for some $P \in \mathcal{P}$. Temporal operators in $\{\mathbf{X}, \mathbf{U}, \mathbf{X^{-1}}, \mathbf{S}\}$ have priority over Boolean operators $\{\wedge, \vee\}$. Moreover, $\neg$ has priority over $\wedge$ and over $\vee$, and $\wedge$ has priority over $\vee$. For instance, $\neg\alpha \wedge \beta\mathbf{U}\gamma \vee \delta$ reads $((\neg\alpha) \wedge (\beta\mathbf{U}\gamma)) \vee \delta$.

We interpret PPLTL over $\mathcal{P}$-labelled linear time structures. The semantics of PPLTL is as follows.

Definition 4.5 (*Semantics of* PPLTL)

Let $\mathcal{M} = (\mathbb{N}, <, V)$ be a $\mathcal{P}$-labelled linear time structure and $i \in \mathbb{N}$. The truth of a PPLTL*-formula ψ in $\mathcal{M}$ with respect to the point i, denoted $\mathcal{M}, i \models \psi$, is defined as follows:*

$\mathcal{M}, i \models P$ *iff* $P \in V(i)$ *for* $P \in \mathcal{P}$;

$\mathcal{M}, i \models \phi \wedge \psi$ *iff* $\mathcal{M}, i \models \phi$ *and* $\mathcal{M}, i \models \psi$;

$\mathcal{M}, i \models \neg\phi$ *iff it is not the case that* $\mathcal{M}, i \models \phi$;

$\mathcal{M}, i \models \phi\mathbf{U}\psi$ *iff* $\mathcal{M}, j \models \psi$ *for some* $j \geq i$ *and* $\mathcal{M}, k \models \phi$ *for each* $i \leq k < j$;

$\mathcal{M}, i \models \mathbf{X}\psi$ *iff* $\mathcal{M}, i+1 \models \psi$;

$\mathcal{M}, i \models \phi\mathbf{S}\psi$ *iff* $\mathcal{M}, j \models \psi$ *for some* $j \leq i$ *and* $\mathcal{M}, k \models \phi$ *for each* $j < k \leq i$;

$\mathcal{M}, i \models \mathbf{X^{-1}}\psi$ *iff* $i > 0$ *and* $\mathcal{M}, i-1 \models \psi$.

We say that $\mathcal{M}$ is a model of ψ if $\mathcal{M}, 0 \models \psi$. A formula ψ is satisfiable *if and only if there is a $\mathcal{P}$-labelled linear time structure that is a model of ψ; it is* valid *if and only if every $\mathcal{P}$-labelled linear time structure is a model of ψ. Note that ψ is valid if and only if $\neg\psi$ is not satisfiable.*

A PPLTL-formula intensionally defines a possibly infinite set of labelled linear time structures (the set of models of the formula). Since, as shown above, consistently labelled linear time structures correspond to granularities, we can use suitable linear time formulas to define possibly infinite sets of granularities. We now put PPLTL at work. The set of all internally continuous granularities is captured by the following PPLTL-formula

$$\mathbf{IntContGran}(G) = \mathbf{G}((P_G \to \alpha) \wedge (Q_G \to \beta)),$$

where

α is $Q_G \vee \mathbf{X}(\neg(P_G \vee Q_G)\mathbf{U}(\neg P_G \wedge Q_G))$, and

β is $P_G \vee \mathbf{X^{-1}}(\neg(P_G \vee Q_G)\mathbf{S}(P_G \wedge \neg Q_G))$.

Note that the first conjunct $P_G \to \alpha$ captures point (1) of Definition 4.2, whereas the second conjunct $Q_G \to \beta$ captures point (2) of Definition 4.2.

The set of continuous granularities are defined by the following formula that requires two consecutive granules to be adjacent:

$$\mathbf{ContGran}(G) = \mathbf{IntContGran}(G) \wedge \mathbf{G}(Q_G \to \mathbf{X}(P_G \vee \mathbf{G}\neg P_G)).$$

The set of continuous and total granularities are defined by the formula that follows:

$$\mathbf{TotalGran}(G) = \mathbf{ContGran}(G) \wedge P_G \wedge \mathbf{GF}P_G.$$

It states that G is a continuous granularity, starting immediately, and having an infinite number of nonempty granules.

The set of uniform granularities cannot be encoded in our framework. Indeed, to encode uniformity, we have to say that any granule have the same *not fixed* cardinality. This boils down to encode the predicate saying that "X and Y are sets of the same cardinality" over the natural numbers. It is well-known that such a predicate cannot be expressed in the monadic second-order theory of natural numbers (S1S, for short) [57]. Since PPLTL is a fragment of S1S [57], it follows that such a predicate, and hence uniformity, cannot be expressed in PPLTL. However, we are able to encode the set of internally continuous granularities having granules of uniform cardinality k, with $k \geq 1$, as follows:

$$\mathbf{Uniform_k}(G) = \mathbf{IntContGran}(G) \wedge \mathbf{G}(P_G \rightarrow (\mathbf{X^{k-1}}Q_G \wedge \bigwedge_{i=0}^{k-2} \mathbf{X^i}\neg Q_G)),$$

where, $\mathbf{X^0}\phi$ stands for ϕ and, for every $n > 0$, $\mathbf{X^n}\phi$ stands for $\mathbf{XX^{n-1}}\phi$.

A finite number of different granularities may be involved in the same formula by using different pairs of marking proposition symbols. For instance, given a calendar $\mathcal{G} = \{G_1, \ldots, G_n\}$, the formula $\bigwedge_{i=1}^{n} \mathbf{IntContGran}(G_i)$ defines the set of all calendars with n granularities $G_1, \ldots, G_n$. The granularities in a calendar usually have to satisfy certain constraints, such as, to be finer than or to group into other granularities. Such constraints may be expressed by means of binary relations between granularities, as those proposed above. We show how to encode the above introduced relations between granularities in our framework.

The relation $\mathbf{FinerThan}(G_1, G_2)$ is expressed by the following formula:

$$\mathbf{IntContGran}(G_1) \wedge \mathbf{IntContGran}(G_2) \wedge$$
$$\mathbf{G}(P_{G_1} \rightarrow ((\neg P_{G_2} \wedge \neg Q_{G_2})\mathbf{S}P_{G_2} \wedge (\neg P_{G_2} \wedge \neg Q_{G_2})\mathbf{U}Q_{G_1})).$$

The formula requires that each granule of G_1 is included into some granule of G_2. In particular, it allows that more than one granule of G_1 is included into the same granule of G_2. Hence, the calendars with n granularities that are totally ordered with respect to the **FinerThan** relation are defined by the following formula:

$$\bigwedge_{i=1}^{n-1} \mathbf{FinerThan}(G_i, G_{i+1}).$$

The relation $\mathbf{SubGranularity}(G_1, G_2)$ is expressed by the formula that follows:

$$\mathbf{IntContGran}(G_1) \wedge \mathbf{IntContGran}(G_2) \wedge$$
$$\mathbf{G}(P_{G_1} \rightarrow (P_{G_2} \wedge \neg(Q_{G_1} \vee Q_{G_2})\mathbf{U}(Q_{G_1} \wedge Q_{G_2}))).$$

The formula above requires that each granule of G_1 coincides with a granule of G_2.

The relation $\mathbf{GranuleRefinement}(G_1, G_2)$ is expressed by the following formula:

$$\mathbf{IntContGran}(G_1) \wedge \mathbf{IntContGran}(G_2) \wedge$$
$$\mathbf{G}((P_{G_1} \rightarrow \alpha) \wedge (Q_{G_1} \rightarrow \beta) \wedge ((P_{G_2} \wedge (\neg P_{G_1}\mathbf{U}Q_{G_2})) \rightarrow \mathbf{G}(\neg P_{G_1}))),$$

where

α is $\neg(P_{G_2} \vee Q_{G_2})\mathbf{U}(Q_{G_1} \wedge (Q_{G_2} \vee \mathbf{X}(\neg P_{G_1}\mathbf{U}Q_{G_2})))$, and

β is $\neg(P_{G_2} \vee Q_{G_2})\mathbf{S}(P_{G_1} \wedge (P_{G_2} \vee \mathbf{X^{-1}}(\neg Q_{G_1}\mathbf{S}P_{G_2})))$.

The formula above requires that the i-th granule of G_2 contains the i-th granule of G_1, if any. Formulas α and β require that at most one granule of G_1 is included into a granule of G_2. Moreover, the third conjunct of the formula ensures that if some granule of G_2 does not contain any granule of G_1, then all the following granules of G_1 are empty.

The operation of defining a granularity by grouping the granules of another granularity might lead in general to non-internally continuous granularities, which are treated in Section 4.2. However, if we group (without skipping) granules of a continuous granularity, the resulting granularity is still internally continuous. We implement in the following the relation $\mathbf{Group}(G_1, G_2)$, assuming that G_1 is continuous.

$$\mathbf{ContGran}(G_1) \wedge \mathbf{IntContGran}(G_2) \wedge$$
$$\mathbf{G}(P_{G_2} \rightarrow (P_{G_1} \wedge \neg Q_{G_2} \mathbf{U}(Q_{G_1} \wedge Q_{G_2}))).$$

Notice that G_2 is required to be internally continuous but not continuous, since it is not required that each granule of G_1 belongs to a granule of G_2.

We now encode the relation $\mathbf{Group_p}(G_1, G_2)$, assuming again that G_1 is continuous.

$$\mathbf{ContGran}(G_1) \wedge \mathbf{ContGran}(G_2) \wedge (\neg P_{G_1} \wedge \neg P_{G_2})\mathbf{U}(P_{G_1} \wedge P_{G_2}) \wedge \mathbf{G}(P_{G_2} \rightarrow \alpha_{p-1}),$$

where,

$$\alpha_0 \text{ is } P_{G_1} \wedge \neg(Q_{G_1} \vee Q_{G_2})\mathbf{U}(Q_{G_1} \wedge Q_{G_2}),$$

and, for $k > 0$,

$$\alpha_k \text{ is } P_{G_1} \wedge \neg(Q_{G_1} \vee Q_{G_2})\mathbf{U}(Q_{G_1} \wedge \neg Q_{G_2} \wedge \mathbf{X}\alpha_{k-1}).$$

The definitions for $\mathbf{Group_p Skip_q}$ and $\mathbf{Skip_q Group_p}$ are similar. However, the relation $\mathbf{EqualGroup}(G_1, G_2)$ is beyond the expressiveness of our framework. Indeed, it forces the granules of G_2 to be composed of the same *not fixed* number of granules of G_1. To express such a property, we should be able to encode the predicate “X and Y are sets of the same cardinality”. As discussed above, such a predicate is not expressible in our setting.

Finally, we encode the relations that shift a granularity. The relation $\mathbf{Shift_p}(G_1, G_2)$ can be written as follows:

$$\mathbf{IntContGran}(G_1) \wedge \mathbf{IntContGran}(G_2) \wedge \alpha_p,$$

where

$$\alpha_0 \text{ is } \mathbf{G}((P_{G_1} \leftrightarrow P_{G_2}) \wedge (Q_{G_1} \leftrightarrow Q_{G_2})),$$

and, for $q > 0$,

$$\alpha_q \text{ is } \neg(Q_{G_1} \vee P_{G_2})\mathbf{U}(Q_{G_1} \wedge \neg P_{G_2} \wedge \mathbf{X}\alpha_{q-1}).$$

Formula α_p requires that there are no granules of the granularity G_2 till the p-th granule of the granularity G_1. Starting from the $p+1$-th granule onwards, the two granularities G_1 and G_2 coincide (formula α_0).

The unbounded shift relation $\mathbf{Shift}(G_1, G_2)$ is as follows:

$$\mathbf{IntContGran}(G_1) \wedge \mathbf{IntContGran}(G_2) \wedge$$
$$(\neg(P_{G_2} \vee Q_{G_2})\mathbf{UG}((P_{G_1} \leftrightarrow P_{G_2}) \wedge (Q_{G_1} \leftrightarrow Q_{G_2}))).$$

The formula above requires that there are no granules of the granularity G_2 till the n-th granule of the granularity G_1, being n an arbitrarily chosen non negative number. From the n-th granule onwards, the two granularities G_1 and G_2 coincide.

Example 4.6 We model in the proposed granularity framework the examples given in Section 3. We preliminary introduce some useful shorthands. For $0 \leq n \leq m$, $\forall[n,m](\alpha)$ (resp. $\exists[n,m](\alpha)$) stands for "α holds everywhere (resp. somewhere) in the time interval $[n,m]$", and is defined as $\bigwedge_{i=n}^{m} \mathbf{X^i}\alpha$ (resp. $\bigvee_{i=n}^{m} \mathbf{X^i}\alpha$). Finally, $Count(\alpha, n)$ stands for "α holds exactly n times in the future" and is defined as follows: $Count(\alpha, 0) = \mathbf{G}\neg\alpha$ and $Count(\alpha, n) = \neg\alpha\mathbf{U}(\alpha \wedge \mathbf{X}Count(\alpha, k-1))$.

We consider now the clinical example of Section 3. Let us assume that OC (cyclophosphamide), IM (intravenous methotrexate) and FI (5-fluorouracil) are the granularities corresponding to the drugs of the CMF regimen. The formula Ω_{CMF} below defines, on the time domain $\mathbb{N}$ of days, the granularities CMF, OC, IM, and FI according to the recommendation given in Section 3:

$$\mathbf{FinerThan}(IM, CMF) \wedge \mathbf{FinerThan}(OC, CMF) \wedge \mathbf{FinerThan}(FI, CMF) \wedge$$
$$Count(P_{CMF}, 6) \wedge \mathbf{Uniform_{28}}(CMF) \wedge \mathbf{G}((Q_{CMF} \wedge \mathbf{F}P_{CMF}) \rightarrow \exists[1,5]P_{CMF}) \wedge$$
$$\mathbf{G}(P_{CMF} \rightarrow (\forall[0,13](P_{OC} \wedge P_{IM} \wedge Q_{OC} \wedge Q_{IM}) \wedge$$
$$P_{FI} \wedge Q_{FI} \wedge \mathbf{X}^7(P_{FI} \wedge Q_{FI}) \wedge \forall[1,6](\neg P_{FI} \wedge \neg Q_{FI}) \wedge \forall[8,13](\neg P_{FI} \wedge \neg Q_{FI}) \wedge$$
$$\forall[14,27](\neg P_{OC} \wedge \neg P_{IM} \wedge \neg P_{FI} \wedge \neg Q_{OC} \wedge \neg Q_{IM} \wedge \neg Q_{FI}))).$$

The first three conjuncts (first line) say that the three granularities related to specific drugs are finer than the CMF granularity and that all the four granularities are internally continuous. The next three conjuncts (second line) say, respectively, that the granularity CMF consists of 6 granules (cycles), each of 28 elements (days), and each cycle is separated by time intervals not exceeding 5 units. The last big conjunct associates the drugs with each day in the cycle, according to the recommendation (the first 14 days cyclophosphamide and intravenous methotrexate, with 5-fluorouracil only on days 1 and 8, and no drugs during the second 14 days).

Example 4.7 In this example we cope with the representation of the Gregorian Calendar. The main (and most tedious) part consists of the definition of the granularity `Month`: indeed, the overall periodicity of months is 4 centuries and the only solution is to explicitly define the granules corresponding to each month within this period. For the sake of readability, we therefore introduce some shorthands. We start by defining the formula $M_n(\varphi)$, encoding a month of n days, where n is a natural number and φ is a formula, and $P_{\texttt{4Century}}$ is the proposition symbol marking the starting point of a granule of 4 centuries:

$$M_n(\varphi) \text{ stands for } P_{\texttt{Month}} \wedge \forall[0, n-2](\neg Q_{\texttt{Month}} \wedge \mathbf{X}\neg P_{\texttt{4Century}}) \wedge \mathbf{X}^{n-1}(Q_{\texttt{Month}} \wedge \mathbf{X}\varphi).$$

We now define the formulas $Y(\varphi)$ and $LY(\varphi)$, encoding, respectively, the months during a year and the months during a leap year as follows (for the sake of readability, we will write M_nM_m instead of $M_n(M_m)$):

$Y(\varphi)$	stands for	$M_{31}M_{28}M_{31}M_{30}M_{31}M_{30}M_{31}M_{31}M_{30}M_{31}M_{30}M_{31}(\varphi);$
$LY(\varphi)$	stands for	$M_{31}M_{29}M_{31}M_{30}M_{31}M_{30}M_{31}M_{31}M_{30}M_{31}M_{30}M_{31}(\varphi).$

Moreover, we define the formulas $4Y(\varphi)$, $C(\varphi)$, and $4C(\varphi)$ encoding, respectively, the months during a 4-year period (with three years and one leap year), the months during a century (with a sequence of 24 periods of 4-year plus a period of 4 non leap years), and the months during a 4-century period (3 centuries plus a sequence of 25 periods of 4-year):

$4Y(\varphi)$	stands for	$Y^3LY(\varphi);$
$C(\varphi)$	stands for	$4Y^{24}Y^4(\varphi);$
$4C(\varphi)$	stands for	$C^3 4Y^{25}(\varphi),$

where Y^3 stands for YYY, and similarly for the other powers.

Unfolding formula $4C$, we have the following encoding of all the rules of the Gregorian Calendar:

$$((Y^3LY)^{24}Y^4)^3(Y^3LY)^{25}(\varphi).$$

Finally, the granularity `Month` is defined as follows:

$$\phi_{\texttt{Month}} = P_{\texttt{4Century}} \wedge \mathbf{G}(P_{\texttt{4Century}} \rightarrow 4C(P_{\texttt{4Century}})).$$

The granularity `Day` is such that $\phi_{\texttt{Day}} = \mathbf{TotalGran}(\texttt{Day}) \wedge \mathbf{Uniform_1}(\texttt{Day})$. The granularity `Week` can be defined as a group of 7 days, that is, $\phi_{\texttt{Week}} = \mathbf{Group_7}(\texttt{Day}, \texttt{Week})$. The granularity `Year` is a group of 12 months, namely $\phi_{\texttt{Year}} = \mathbf{Group_{12}}(\texttt{Month}, \texttt{Year})$, and the granularity `Century` is a group of 100 years, i.e., $\phi_{\texttt{Century}} = \mathbf{Group_{100}}(\texttt{Year}, \texttt{Century})$. As for the granularity `4Century`, being a group of 4 centuries, we have that $\phi_{\texttt{4Century}} = \mathbf{Group_4}(\texttt{Century}, \texttt{4Century})$ holds. The Gregorian Calendar is finally defined as follows:

$$\phi_{\texttt{Day}} \wedge \phi_{\texttt{Week}} \wedge \phi_{\texttt{Month}} \wedge \phi_{\texttt{Year}} \wedge \phi_{\texttt{Century}} \wedge \phi_{\texttt{4Century}}.$$

It is worth noting that in the above examples only a bounded form of uncertainty is involved. In the clinical example, two successive cycles may be separated by no more than 5 time units (in the chosen granularity). Moreover, in the Gregorian Calendar, no degree of uncertainty is present. However, there exist applications calling for *unbounded uncertainty*. For instance, two therapy cycles that are arbitrarily distant or a therapy starting at an arbitrary instant. Our framework can cope with unbounded uncertainty as well.

4.2 Extending granularities with gaps

The above proposal does not consider granularities with gaps inside the granules (only internally continuous granularities are treated). However, it can be easily extended to cope with such granularities. Let $\mathcal{G}$ be a calendar and $\mathcal{P}_{\mathcal{G}} = \{P_G, Q_G, P_{H_G}, Q_{H_G} \mid G \in \mathcal{G}\}$ be a set of proposition symbols associated with the calendar $\mathcal{G}$. Given an alphabet of proposition symbols $\mathcal{P} \supseteq \mathcal{P}_{\mathcal{G}}$, we shall consider $\mathcal{P}$-labelled linear time structures. We use symbols P_G and Q_G to delimit the granules of a granularity G as before, and we take advantage of symbols P_{H_G} and Q_{H_G} to bound the gaps inside the granules of G. In this way we have that the description of the gaps of G is itself a granularity H_G. Note that H_G is finer than G. Indeed, every granule of H_G (an internal gap of G) is a subset of some granule of G. Moreover, there are no granules of G that are entirely covered by granules of H_G. The extension of the definition of G-consistency is as follows.

Definition 4.8 *A labelled linear time structure $\mathcal{M} = (\mathbb{N}, <, V)$ is G-gap-consistent whenever:*

1. *$\mathcal{M}$ is G-consistent and H_G-consistent (according to Definition 4.2);*
2. *every granule of $\mathcal{M}$ with respect to H_G is a subset of some granule of $\mathcal{M}$ with respect to G;*
3. *no granule of $\mathcal{M}$ with respect to G is the union of some granules of $\mathcal{M}$ with respect to H_G.*

It is easy to show that a G-gap-consistent linear time structure corresponds to a (not necessarily continuous) granularity and vice versa. The set of (not necessarily continuous) granularities can be encoded by the temporal formula $\mathbf{Gran}(G)$ defined as follows:

$$\mathbf{FinerThan}(H_G, G) \wedge \mathbf{G}(P_G \rightarrow \neg\alpha),$$

where α stands for

$$P_{H_G} \wedge ((Q_G \wedge Q_{H_G}) \vee \mathbf{X}((\neg Q_G \wedge (Q_{H_G} \rightarrow \mathbf{X}P_{H_G}))\mathbf{U}(Q_G \wedge Q_{H_G}))).$$

The first conjunct of $\mathbf{Gran}(G)$ states that G and H_G are consistent (point (1) of Definition 4.8) and that each granule of H_G is a subset of a granule of G (point (2) of Definition 4.8). The second conjunct of $\mathbf{Gran}(G)$ claims that each granule of G is not entirely covered by granules of H_G (point (3) of Definition 4.8). As an example, we define the non-internally continuous granularity `BusinessMonth`. The granules of a business month have gaps corresponding to weekends. We define the granularity `BusinessMonth` on top of the previously defined granularity `WeekEnd` as follows:

$\mathbf{Gran}(\texttt{BusinessMonth}) \wedge \mathbf{G}(P_{\texttt{BusinessMonth}} \leftrightarrow P_{\texttt{Month}} \wedge Q_{\texttt{BusinessMonth}} \leftrightarrow Q_{\texttt{Month}}) \wedge$
$\mathbf{G}(P_{H_{\texttt{BusinessMonth}}} \leftrightarrow (P_{\texttt{WeekEnd}} \vee (Q_{\texttt{WeekEnd}} \wedge P_{\texttt{BusinessMonth}})) \wedge$
$Q_{H_{\texttt{BusinessMonth}}} \leftrightarrow (Q_{\texttt{WeekEnd}} \vee (P_{\texttt{WeekEnd}} \wedge Q_{\texttt{BusinessMonth}}))).$

4.3 Expressiveness

A precise upper bound to the expressiveness of our approach is fixed by the expressiveness of PPLTL. In fact, granularities are induced by labelled linear time structures (i.e. ω-sequences over a suitable alphabet) and it is well known that the set of labelled linear time structures which satisfy a PPLTL formula is a ω-regular set, namely a set of ω-sequences accepted by a Büchi automaton (e.g., see [57]); actually, PPLTL allows one to capture a proper fragment of the class of ω-regular sets. In the previous sections we have given examples of relations which are not definable in our setting (for instance, the relation $\mathbf{EqualGroup}(G_1, G_2)$ or the property of uniformity for granularities). An example of a granularity G that is not regular, and hence can not be represented in our framework, is the following: for every $i \geq 0$, $G(i) = \{2^i + 1, 2^i + 2, \ldots, 2^{i+1}\}$ (note that the cardinality of the i-th granule $G(i)$ is 2^i). However, even though PPLTL allows one to capture only a proper fragment of the class of ω-regular sets, we shall show in this section that it allows the description of the class of granularities considered of practical interest. In particular, we shall show that our setting is at least as expressive to define all finite, infinite periodical, and quasi-periodical granularities, thus placing its expressive power beyond that of other widely known frameworks (e.g., the collection formalism). Conversely, we have not yet established whether the class of finite, infinite periodical, and quasi periodical granularities precisely captures the class of granularities definable in our framework (a precise characterization of the class of definable granularities is still missing and will be addressed in the future work).

In [9], Bettini and De Sibi studied the expressive power of the two well known frameworks of collection and slice formalisms, comparing them mainly with respect to the subclass of infinite *periodical* granularities. Periodical granularities can be defined in terms of another granularity by means of a particular type of grouping relationship. The idea is that the defined granularity has a repeating pattern of length R corresponding to a span of P granules of the underlying granularity.

Definition 4.9 *A granularity G_2 is periodical with respect to a granularity G_1 if the relation $\mathbf{Group}(G_1, G_2)$ holds and there exist $R, P \in \mathbb{N}$ (the* pattern*), where R is less than (or equal to) the number of granules of G_2, such that for all $i \in \mathbb{N}$, if $G_2(i) = \bigcup_{j \in S} G_1(j)$ and $G_2(i+R) \neq \emptyset$, then $G_2(i+R) = \bigcup_{j \in S} G_1(j+P)$.*

For instance, if $\mathbf{Group_p}(G_1, G_2)$ (resp., either $\mathbf{Group_p Skip_q}(G_1, G_2)$ or $\mathbf{Skip_q Group_p}(G_1, G_2)$) holds, then G_2 is periodical with respect G_1 with $R = 1$ and $P = p$ (resp., $R = 1$ and $P = p + q$). Given a *basic* total continuous granularity, a granularity G is said to be *periodical* if it is periodical with respect to that basic granularity. The expressive power of the collection and slice formalisms with respect to the class of finite and infinite periodical no-gap granularities is summarized by the following results [9], according to the terminology used in our proposal:

1. For any collection expression, there exists an equivalent internally continuous finite or infinite periodical granularity, and viceversa.
2. For any disjoint slice expression (i.e., where the granules of the defined granularities do not overlap) there exists an equivalent internally continuous finite or infinite periodical granularity, and viceversa.

In the following we show that our formalism allows one to express periodical granularities, thus allowing to prove that it is at least as expressive as collection and (disjoint) slice formalisms.
In fact, for $R, P \in \mathbb{N}$, the relation $\mathbf{PeriodicGroup_{R,P}}(G_1, G_2)$, which holds if G_2 is periodical with respect to G_1 with pattern R, P, is defined as follows:

$$\mathbf{Group}(G_1, G_2) \wedge Count_R In_P \wedge \neg(P_{G_1} \vee P_{G_2}) \mathbf{U} (P_{G_1} \wedge P_{G_2}) \wedge$$
$$\mathbf{G}((P_{G_1} \rightarrow (P_{G_2} \leftrightarrow disp_P(P_{G_2}, P_{G_1}))) \wedge (Q_{G_1} \rightarrow (Q_{G_2} \leftrightarrow disp_P(Q_{G_2}, Q_{G_1}))))$$

where, $Count_0 In_0$ is $True$, and for $k, s > 0$, $Count_k In_0$ is $False$ and

$$Count_0 In_s \text{ is } \neg(P_{G_1} \vee P_{G_2}) \mathbf{U} (P_{G_1} \wedge \neg P_{G_2} \wedge \mathbf{X} Count_0 In_{s-1})$$

and $Count_k In_s$ is

$$(\neg(P_{G_1} \vee P_{G_2}) \mathbf{U} (P_{G_1} \wedge \neg P_{G_2} \wedge \mathbf{X} Count_k In_{s-1})) \vee$$
$$(\neg(P_{G_1} \vee P_{G_2}) \mathbf{U} (P_{G_1} \wedge P_{G_2} \wedge \mathbf{X} Count_{k-1} In_{s-1}))$$

and, for ϕ, ϕ' formulas, $disp_0(\phi, \phi')$ is ϕ , and for $k > 0$

$$disp_k(\phi, \phi') \text{ is } \mathbf{X}(\mathbf{G}(\neg\phi') \vee (\neg\phi'\mathbf{U}(\phi' \wedge disp_{k-1}(\phi)))).$$

In the formula above, the two initial conjuncts require that G_2 is a grouping of granules of G_1 and that the initial span of P granules of G_1 contains exactly R granules of G_2 (notice that in this way G_2 is forced to have at least R granules); the third conjunct requires that the first granule of G_1 starts together with the first granule of G_2; the last conjunct requires that the starting point (resp. the ending point) of the $i - th$ granule of G_1 (for any i) is the starting point (resp. ending point) of a granule of G_2 if and only if the starting point (resp. the ending point) of the $i + P - th$ granule of G_1 (if any) is the starting point (resp. ending point) of a granule of G_2.

Actually, the expressive power of our approach goes beyond the subclass of finite and infinite periodical no-gap granularities, which are expressible by collection and (disjoint) slice formalisms. Indeed, in Section 4.2, we showed that we can capture gap granularities, i.e., with gaps inside granules (provided that also gaps are periodical). Thus, assuming that we define according to the previous formula both periodical granularities G_2 and H_{G_2} with respect to the basic granularity G_1 (provided that G_2 and H_{G_2} define a G-gap-consistent linear time structure, according to Definition 4.8), we can express both gap and no-gap, finite and infinite periodical, granularities.

Furthermore, in [9] a notion of *quasi-periodicity* is introduced which extends the notion of periodicity by imposing the periodicity of grouping in the whole granularity, but inside a finite (fixed) number of intervals.

Definition 4.10 *A granularity G_2 is quasi-periodical with respect to a granularity G_1 if $\mathbf{Group}(G_1, G_2)$ holds and there exist a set of intervals (i.e., sets of consecutive indexes on G_2) $E_1, \ldots E_k$ (the granularity exceptions), and two numbers $R, P \in \mathbb{N}$, with R less than (or equal to) the minimum of the number of granules of G_2 between any two exceptions, such that for all $i \in \mathbb{N}$ not belonging to any granularity exception, if $G_2(i) = \bigcup_{j \in S} G_1(j)$ and $G_2(i + R) \neq \emptyset$, and $i + R < min(\overline{E})$, where $\overline{E}$ is the closest existing exception after i, then $G_2(i + R) = \bigcup_{j \in S} G_1(j + P)$.*

In the following we shall define grouping relations allowing the expression of a quasi-periodical granularity with a granularity exception having either fixed indexes or arbitrarily placed indexes. The definition could be easily generalized to quasi-periodical granularities with a number $k > 0$ of granularity exceptions thus showing that quasi-periodical granularities can be defined in our framework.

We start with defining a relation of periodical grouping $\mathbf{PeriodicGroup_{R,P}Except_{B,E}}(G_1, G_2)$, where $R, P, B, E \in \mathbb{N}$, $B > R$ and B and E represent the starting and ending indexes, respectively, of the exception interval. The relation, which holds if G_2 is quasi-periodical with respect to G_1 with pattern R, P and a granularity exception $[B, E]$, is defined as follows.

$$\begin{gathered}
\mathbf{Group}(G_1, G_2) \wedge Count_R In_P \wedge \neg(P_{G_1} \vee P_{G_2})\mathbf{U}(P_{G_1} \wedge P_{G_2}) \wedge \\
After_{E+1,G_2}(\mathbf{PeriodicGroup_{R,P}}(G_1, G_2)) \wedge \\
((P_{G_1} \rightarrow (P_{G_2} \leftrightarrow disp_P(P_{G_2}, P_{G_1}))) \wedge (Q_{G_1} \rightarrow (Q_{G_2} \leftrightarrow disp_P(Q_{G_2}, Q_{G_1}))))\mathbf{U} At_{B-R}(G_2)
\end{gathered}$$

where, for a granularity G and a formula ϕ, $After_{0,G}(\phi)$ is ϕ, and for $k > 0$

$$After_{k,G}(\phi) \text{ is } \neg P_G \mathbf{U}(P_G \wedge \mathbf{X} After_{k-1,G}(\phi)),$$

and, for a granularity G, $At_0(G)$ is $P_G \wedge \neg \mathbf{X^{-1}}(\mathbf{P}(P_G))$, and, for $k > 0$,

$$At_k \text{ is } P_G \wedge \mathbf{X^{-1}}(\neg P_G \mathbf{S}(P_G \wedge At_{k-1}(G))).$$

The formula above requires that starting from the $E + 1$-th granule, G_2 is obtained by periodically grouping granules of G_1 (fourth conjunct in the formula); moreover, it imposes (last conjunct) the condition that, for all i with $0 \leq i \leq B - R$, $G_2(i) = \bigcup_{j \in S} G_1(j)$ implies $G_2(i + R) = \bigcup_{j \in S} G_1(j + P)$.

We define now a relation of periodical grouping $\mathbf{PeriodicGroup_{R,P}Except_1}(G_1, G_2)$, with $R, P \in \mathbb{N}$, admitting one exception interval of arbitrary length arbitrarily placed in the granularity. The relation is defined as follows.

$$\begin{gathered}
\mathbf{Group}(G_1, G_2) \wedge Count_R In_P \wedge \neg(P_{G_1} \vee P_{G_2})\mathbf{U}(P_{G_1} \wedge P_{G_2}) \wedge \\
((P_{G_1} \rightarrow (P_{G_2} \leftrightarrow disp_P(P_{G_2}, P_{G_1}))) \wedge (Q_{G_1} \rightarrow (Q_{G_2} \leftrightarrow disp_P(Q_{G_2}, Q_{G_1}))))\mathbf{U} \\
P_{G_2} \wedge After_{R,G_2}(True\mathbf{U}(P_{G_2} \wedge \mathbf{PeriodicGroup_{R,P}}(G_1, G_2))).
\end{gathered}$$

Notice that, in general, considering a fixed granularity G_1 exactly one granularity G_2 may satisfy the relation $\mathbf{PeriodicGroup_{R,P}Except_{B,E}}(G_1, G_2)$, whereas a (possibly infinite) set of granularities may satisfy the relation $\mathbf{PeriodicGroup_{R,P}Except_1}(G_1, G_2)$.
It is easy to see that both the relations $\mathbf{PeriodicGroup_{R,P}Except_{B,E}}$ and $\mathbf{PeriodicGroup_{R,P}Except_1}$ can be generalized to treat a number k of granularity exceptions.

Also for quasi-periodical granularities, we can easily move to granularities with gaps, by defining a suitable quasi-periodical granularity H_{G_2}, which forms, together with G_2, a G-gap consistent linear structure.

Finally, we can conclude that the constructions given in this section prove the following expressiveness property.

Theorem 4.11 *Any finite, infinite quasi-periodical, and infinite periodical granularity (with or without gaps) can be defined in the proposed logical formalism.*

We have also shown that we can encode (possibly) infinite sets of granularities in a single formula, thus overcoming the expressiveness of other formalisms proposed in the literature: for example, we can express granularities not anchored to the time domain or having some uncertainty, as in the case of the chemotherapy cycles, which can have up to 5 days of delay between the end of a cycle and the start of the next one.

As previously underlined, it remains for further investigation the validity of the converse property, namely whether any granularity, which can be defined in our setting over the basic granularity, is either finite or infinite periodical or infinite quasi-periodical.

4.4. Reasoning about time granularity

Besides representing sets of granularities and relations among them, our framework permits to automatically reason about the defined granularities. We give some examples of relevant problems that we can automatically solve in our framework.

Consistency, equivalence and classification problems. The *consistency problem* is the problem of deciding whether a granularity representation is well-defined. This problem is relevant whenever the granularities are represented in a declarative way as formulas of a logical language, as in the present approach. On the contrary, if granularities are operationally represented as algebraic expressions, the consistency problem assumes little relevance, since most algebraic formalisms exclude wrong expressions by construction. Let $\varphi(G)$ be a formula using only proposition symbols in the set $\{P_G, Q_G, P_{H_G}, Q_{H_G}\}$. One can verify whether $\varphi(G)$ encodes a set of well-defined granularities by checking the validity of the formula

$$\varphi(G) \rightarrow \mathbf{Gran}(G).$$

The *equivalence problem* is the problem of deciding whether two different representations define the same granularity. Let $\varphi_1(G)$ and $\varphi_2(G)$ be formulas encoding sets of granularities. It is possible to check whether the two sets of granularities are the same by testing the validity of the formula

$$\varphi_1(G) \leftrightarrow \varphi_2(G).$$

Finally, the *classification problem* solves the problem of deciding whether a natural number n, representing a time point, belongs to a granule of a given granularity. Let $\varphi(G)$ be a formula encoding a set of granularities and

n be a natural number. Then n belongs to some granule of any granularity defined by $\varphi(G)$ if and only if

$$\varphi(G) \rightarrow (\alpha_n(G) \wedge \neg\alpha_n(H_G)),$$

is valid, where $\alpha_n(G)$ is as follows:

$$\mathbf{X}^n(P_G \vee Q_G) \vee \mathbf{X}^n(\neg(P_G \vee Q_G)\mathbf{S}P_G \wedge \neg(P_G \vee Q_G)\mathbf{U}Q_G).$$

Automatic verification of granularity properties. Once we have defined a calendar $\mathcal{G}$ by means of a linear time formula $\varphi(\mathcal{G})$, one may verify whether the calendar satisfies a given property p by encoding p as a linear time formula ψ_p and by checking the validity of $\varphi(\mathcal{G}) \rightarrow \psi_p$. This is a generalization of the classification problem. For instance, with reference to our clinical example, a concrete chemotherapy plan can be checked for consistency against the formula Ω_{CMF} describing the chemotherapy regimen. Moreover, with reference to the Gregorian Calendar, one may encode and check the following properties: “is 2000 a leap year?”, “is 6st February 2003 a working day?”, and similar properties.

Automatic generation of granularities. Given a formula $\varphi(G)$ defining a set of granularities, we would like to automatically generate the granularities encoded by $\varphi(G)$. Moreover, we expect to generate the granularities in order of increasing size. For instance, with reference to our clinical example, we would like to obtain some minimal schedules for a chemotherapy according to the regimen encoded by the formula Ω_{CMF}.

The reasoning procedures above reduce either to check the validity of a PPLTL-formula, or to generate its models in increasing size order. We now describe how these two tasks can be performed. As for the problem of automatic generation of models for a given linear time formula, there are two technical difficulties: the models for linear time formulas are infinite structures, and hence they cannot be explicitly generated. Moreover the set of models of a linear time structure may be infinite, and hence it is not possible to generate all the models. We can cope with the former problem by encoding an infinite linear structure into a finite *ultimately periodic structure*, which is a finite (possibly empty) initial segment followed by a finite loop. The unfolding of the periodic structure gives us the original infinite model. To cope with the latter problem, the generation procedure generates models of increasing size, starting from small models and proceeding to bigger and bigger ones, until a maximum size is reached.

As for the validity problem, notice that checking that a formula is true in every model corresponds to check that there is no model in which its negation is true. In other words, checking the validity of a formula is equivalent to verify that the negation of the formula is not satisfiable. Since PPLTL contains negation, the validity and satisfiability problems for it are computationally equivalent.

The satisfiability problem for PPLTL have been extensively studied from a theoretical point of view, and efficient procedures and heuristics for attacking the problem have been devised and implemented. It belongs to the complexity class PSPACE [56], which means that is can be solved using a polynomial amount of space in terms of the length of the input formula. Moreover, it is complete for PSPACE, which means that there is no hope to find a better algorithm. Albeit the space is polynomial, the time taken by the satisfiability procedure to terminate may be exponential in the length of the formula. This could represent a serious drawback whenever the formula is long. Nevertheless, some recent work effectively attacks this problem [2, 10, 43, 32], making it possible to practically verify reasonable long formulas. We briefly describe these contributions.

The authors of [10] propose an alternative model checking technique for propositional linear time logic PLTL called *Bounded Model Checking* (BMC). This technique has been extended to past propositional linear time logic PPLTL in [2] and is implemented in the state-of-the-art symbolic model checker NuSMV [14, 15]. In BMC, an existential model checking instance for PLTL is reduced to an instance of the popular propositional satisfiability problem SAT, and efficient SAT solvers are then used to tackle this problem. More precisely, BMC tackles the following *bounded* version of the existential model checking problem: given a finite model M, a past propositional linear time formula φ, and an integer $k \geq 0$, check whether there exists an ultimately periodic path of length k belonging to the model M that satisfies φ. If such a periodic path exists, it can be unfolded obtaining an infinite path in the model M that satisfies the formula φ. The bounded existential model checking problem can be efficiently (in particular, polynomially) reduced to SAT. The latter can be efficiently attacked by exploiting the impressive power of state-of-the-art propositional solvers. This approach solves both the satisfiability problem and the model generation one for PPLTL. Indeed, it is well-known that the linear time satisfiability problem can be embedded into the linear time existential model checking problem. It is sufficient to use in the model checking instance a fictitious structure encoding all possible paths. Now we can interactively solve a bounded model checking instance of size k, for $k = 0, 1, \ldots$. This generates models of the formula in increasing size order. Moreover, since a PPLTL formula φ is satisfiable if and only if it is true in an ultimately periodic path of length exponential in the length of φ [56], the generation procedure gives also a constructive way to solve the satisfiability problem for φ. The advantage is that in many practical cases a small model for a formula is detected soon by the generation procedure, and thus the procedure can stop without performing an exponential number of steps.

The contribution of [43] relevant to the current discussion is the result that the model and satisfiability checking problems for future and past temporal logic, that is PPLTL without Since, Until, Next-time and Previous-time operators, is NP-complete, instead of PSPACE-complete. The proof exploits the *linear size witness property* for future and past temporal logic: a future and past temporal formula is satisfiable if and only is if is true in an ultimately periodic path of size *linear* in the length of the formula. Exploiting this result in the BMC technique described above, we have the guarantee that after a linear number of bounded model checks, either we have found a model for the formula (and hence the formula is satisfiable) or we can conclude that the formula is not satisfiable.

Finally, in [32], the authors show that a limited version of Since, Until, Previous-time and Next-time operators is still possible without sacrificing the nice computational behaviour. As soon as we avoid Since, Until, Previous-time and Next-time operators in the scope of universal temporal ones, like **G** and the universal part of Since and Until, we have that the satisfiability problem is still in NP (and the linear size witness property is preserved). Dually, as soon as we avoid Since, Until, Previous-time and Next-time operators in the scope of existential temporal ones, like **F** and the existential part of Since and Until, we have that the validity problem is in coNP, and hence its dual is in NP. Notice that all the formulas that we used in Section 4 are of this latter kind.

5. Discussion

In this section we summarize the comparison, we performed throughout Sections 1, 2, 4, of our approach with the related ones. As discussed in Section 2, there are at least three main approaches to represent and reason about time granularity: the algebraic one by Jajodia et al., the logical one by Montanari et al., and the string-based one by Wijsen and Dal Lago et al.

The starting points of the approach proposed in this paper and that of the algebraic approach coincide: it is the classical and general definition of time granularity given in [8]. However, our approach differs from the algebraic one since, in the latter, granularities are algebraic expressions, whereas we encode granularities by means of logical formulas. Moreover, we are able to speak of possibly infinite sets of granularities, symbolically encoded by logical formulas. This feature permits us to represent *unanchored granularities*, that is, granularities that are not anchored to the underlying time domain. Typical examples of unanchored granularities are a repeating pattern that can start at an arbitrary time point or two finite repeating patterns arbitrarily distant from each other. On the contrary, the algebraic approach can encode only *anchored granularities*. Our approach is fully automatic: reasoning about granularities reduce to solving well-known validity problems in linear time logic. On the contrary, a major weakness of the algebraic approach is that reasoning methods basically reduce to granule conversions and semantic translations of statements, and little attention has been devoted to other forms of reasoning (like equivalence checking).

Our approach differs from the logical one by Montanari et al. [31, 44, 46, 49] for the following reason: while Montanari et al. model different time granularities by using multi-layered mathematical structures and use temporal logic formulas to capture *properties* of time granularities, we model both time granularities and their properties by using temporal logic formulas. To allow nice computational properties, the logical approach makes strict assumptions about the interrelations among granularities in the layered structures; for example, in the work of

Montanari et al., all granularities are total, uniform, internally and externally continuous, and linearly ordered with respect to the ‘finer than’ relation. Our solution is much more flexible: we can represent non-total, non-continuous, non-uniform granularities, partially ordered with respect to the ‘finer than’ relation. We only require that granularities show some form of periodicity. Moreover, the time granularity structure may be changed by simply modifying the logical formula that defines it, and the properties of the time granularity structure may be defined in the same logical language. One advantage of the logical approach of Montanari et al. is that it can represent calendars with an infinite number of layers, corresponding to infinite granularities linearly ordered with respect to the ‘finer than’ relation, whereas we only capture calendars possibly having infinite granularities, but on a finite number of layers, according to the ordering induced by the ‘finer than’ relation.

Our approach is mostly related to the string-based approaches by Wijsen [60], and Dal Lago, Montanari, and Puppis [22, 23, 24]. All these approaches and our approach represent granularities as infinite labelled structures, that is, infinite strings. One main difference is that we can encode unanchored granularities by representing them with (possibly) infinite sets of granularities, whereas the string-based approaches allow one only to represent single (anchored) granularities. This increase in the expressiveness is however paid in terms of a complexity blow-up. Reasoning about granularities in our framework has polynomial space but exponential time complexity, while reasoning about granularities in the string-based framework has polynomial time (and space) complexity. The feeling is that for some application involving (un)bounded uncertainty, our framework is what is needed, but in some other cases our framework is too much expressive (and computationally complex). For instance, the Gregorian Calendar can be represented in the simpler string-based approach as well, since no degree of uncertainty is required.

A preliminary version of our approach has been described in [17]. In that paper, we did not provide the exhaustive encoding of the relationships between granularities we gave here in Section 4. Moreover, we discussed here in some detail the most relevant frameworks proposed for specifying time granularities and showed analogies and differences with our proposal. Finally, the discussion about the expressiveness of our proposal is completely new, as well as the discussion on its computational features.

6. Conclusions and Future Work

In this paper, we proposed an original approach to represent and to reason about different time granularities.

We identified a time granularity with a discrete linear time structure properly labelled with proposition symbols marking the starting and ending points of the corresponding granules and of their (possible) internal gaps. We adopted the linear time logic PPLTL, interpreted over labelled linear time structures, to model possibly infinite sets of time granularities.

In particular, the proposed approach allows one to overcome some specific limits of the algebraic and logical frameworks in expressing real-world granularities, as shown in Section 3 for a clinical domain: indeed, it is possible to express unanchored granularities, i.e., granularities not anchored to a fixed origin on the time domain or having some finite parts which can be arbitrarily distant (possibly, within a given range).

In general, the proposed formalism permits to model a large set of regular granularities and to algorithmically solve the consistency, the equivalence, and the classification problems in a uniform way by reducing them to the validity problem for the considered linear time logic, which is known to be decidable in polynomial space.

As for future work, we shall investigate the problem of assessing a characterization of the class of granularities definable in our approach. Moreover, we aim at integrating our approach with the string-based one in order to obtain a more tuned framework for time granularity with respect to real-world applications. The starting point could be the following question. We know that linear time formulas can be converted into equivalent Büchi automata of size exponential in the length of the formula [57]. Is there an encoding of single-string linear time formulas, that is, formulas with exactly one model (like the formula encoding the Gregorian Calendar), into single-string automata of size polynomial in the length of the formula?

A further research direction is towards the integration of the proposed formalism within the context of temporal databases: we will explore how to exploit our approach for expressing and verifying temporal functional dependencies involving several granularities or, more generally, for expressing and checking integrity constraints.

References

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. L. P. Chen, editors, *Proceedings of the International Conference on Data Engineering*, pages 3–14. IEEE Press, 6–10 Mar. 1995.

[2] M. Benedetti and A. Cimatti. Bounded model checking for past LTL. In *Proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 2619 of *Lecture Notes in Computer Science*, pages 18–33. Springer, 2003.

[3] C. Bettini, A. Brodsky, S. Jajodia, and X. S. Wang. Logical design for temporal databases with multiple granularities. *ACM Transactions on Database Systems*, 22(2):115–170, June 1997.

[4] C. Bettini, S. Jajodia, J. Lin, and X. S. Wang. Discovering frequent event patterns with multiple granularities in time sequences. *IEEE Transactions on Knowledge and Data Engineering*, 10(2):222–237, 1998.

[5] C. Bettini, S. Jajodia, and X. Wang. A general framework and reasoning models for time granularity. In *Proceedings of the International Workshop on Temporal Representation and Reasoning*, pages 104–111. IEEE Computer Society Press, 1996.

[6] C. Bettini, S. Jajodia, and X. S. Wang. Testing complex temporal relationships involving multiple granularities and its application to data mining. In *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, volume 15, pages 68–78. ACM Press, 1996.

[7] C. Bettini, S. Jajodia, and X. S. Wang. A general framework for time granularity and its application to temporal reasoning. *Annals of Mathematics and Artificial Intelligence*, 22:29–58, 1998.

[8] C. Bettini, S. Jajodia, and X. S. Wang. *Time Granularities in Databases, Data Mining, and Temporal Reasoning*. Springer, 2000.

[9] C. Bettini and R. D. Sibi. Symbolic representation of user-defined time granularities. *Annals of Mathematics and Artificial Intelligence*, 30(1-4):53–92, 2000.

[10] A. Biere, A. Cimatti, E. M. Clarke, M. Fujita, and Y. Zhu. Symbolic model checking using SAT procedures instead of BDDs. In *Proceedings of Design Automation Conference*, volume 1579 of *Lectures Notes in Computer Science*, pages 193–207, 1999.

[11] P. Blackburn and J. Bos. *Representation and Inference for Natural Language*. Studies in Logic, Language, and Information. CSLI Press. Forthcoming.

[12] R. Chandra, A. Segev, and M. Stonebraker. Implementing calendars and temporal rules in next generation databases. In A. K. Elmagarmid and E. Neuhold, editors, *Proceedings of the International Conference on Data Engineering*, pages 264–273, Houston, TX, Feb. 1994. IEEE Computer Society Press.

[13] E. Ciapessoni, E. Corsetti, A. Montanari, and P. San Pietro. Embedding time granularity in a logical specification language for synchronous real-time systems. *Science of Computer Programming*, 20:141–171, 1993.

[14] A. Cimatti, E. M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella. NuSMV 2: an opensource tool for symbolic model checking. In *Proceedings of the International Conference on Computer Aided Verification*, volume 2404 of *Lecture Notes in Computer Science*, pages 359–364. Springer, 2002.

[15] A. Cimatti, E. M. Clarke, F. Giunchiglia, and M. Roveri. NuSMV: a new symbolic model verifier. In *Proceedings of the International Conference on Computer Aided Verification*, volume 1633 of *Lecture Notes in Computer Science*, pages 495–499. Springer, 1999.

[16] C. Combi and L. Chittaro. Temporal granularity and indeterminacy in reasoning about actions and change: an approach based on the event calculus. *Annals of Mathematics and Artificial Intelligence*, 36:81–119, 2002.

[17] C. Combi, M. Franceschet, and A. Peron. A logical approach to represent and reason about calendars. In *Proceedings of the International Symposium on Temporal Representation and Reasoning*, pages 134–140. IEEE Computer Society Press, 2002.

[18] C. Combi and G. Pozzi. A Temporal Data Model Managing Intervals with Different Granularities and Indeterminacy from Natural Language Sentences. *The VLDB Journal*, 9:294–311, 2001.

[19] E. Corsetti, E. Crivelli, D. Mandrioli, A. Montanari, A. Morzenti, P. S. Pietro, and E. Ratto. Dealing with different time scales in formal specifications. In *Proceedings of the International Workshop on Software Specification and Design*, pages 92–101. IEEE Computer Society Press, 1991.

[20] E. Corsetti, A. Montanari, and E. Ratto. Dealing with different time granularities in formal specifications of real-time systems. *The Journal of Real-Time Systems*, 3:191–215, 1991.

[21] D. Cukierman and J. Delgrande. Expressing time intervals and repetition within a formalization of calendars. *Computational Intelligence*, 14(4):563–597, 1998.

[22] U. Dal Lago and A. Montanari. Calendars, time granularities, and automata. In *Proceedings of the International Symposium on Spatial and Temporal Databases*, volume 2121 of *Lectures Notes on Computer Science*, pages 279–298, Los Angeles, CA, USA, 2001.

[23] U. Dal Lago, A. Montanari, and G. Puppis. Time granularities, calendar algebra, and automata. Technical Report 4/2003, Department of Mathematics and Computer Science, University of Udine – Italy, 2003.

[24] U. Dal Lago, A. Montanari, and G. Puppis. Towards compact and tractable automaton-based representations of time granularity. In *Proceedings of the Eighth Italian Conference on Theoretical Computer Science (ICTCS03)*, Lecture Notes in Computer Science. Springer, 2003.

[25] W. Dreyer, A. K. Dittrich, and D. Schmidt. Research perspectives for time series management systems. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 23(1):10–15, Mar. 1994.

[26] C. E. Dyreson and R. T. Snodgrass. Temporal granularity. In R. T. Snodgrass, editor, *The TSQL2 Temporal Query Language*, pages 347–385. Kluwer Academic Press, 1995.

[27] E. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Vol. B*, pages 995–1072. Elsevier Science Publishers B.V., 1990.

[28] J. Euzenat. An algebraic approach for granularity in qualitative space and time representation. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 894–900. Morgan Kaufmann, 1995.

[29] J. L. Fiadeiro and T. Maibaum. Sometimes ”tomorrow” is ”sometime”: Action refinement in a temporal logic of objects. In D. M. Gabbay and H. J. Ohlbach, editors, *Proceedings of the International Conference on Temporal Logic*, volume 827 of *Lectures Notes on Artificial Intelligence*, pages 48–66, Berlin, July 1994. Springer.

[30] D. Foster, B. Leban, and D. McDonald. A representation for collections of temporal intervals. In *Proceedings of the American National Conference on Artificial Intelligence*, pages 367–371, 1986.

[31] M. Franceschet. *Dividing and conquering the layered land*. PhD thesis, Department of Mathematics and Computer Science, University of Udine, 2001.

[32] M. Franceschet, M. de Rijke, and B.-H. Schlingloff. Hybrid logics on linear structures: Expressivity and complexity. In *Proceedings of the International Symposium on Temporal Representation and Reasoning and of the International Conference on Temporal Logic*, pages 164–171. IEEE Computer Society Press, 2003.

[33] M. Franceschet and A. Montanari. Branching within time: an expressively complete and elementarily decidable temporal logic for time granularity. *Research on Language and Computation*, 1(3/4):229–263, 2003.

[34] D. Fum, G. Guida, A. Montanari, and C. Tasso. Using levels and viewpoints in text representation. In *Proceedings of the International Conference on Artificial Intelligence and Information-Control Systems of Robots*, pages 37–44, 1989.

[35] I. Goralwalla, Y. L. M. Ozsu, D. Szafron, and C. Combi. Temporal Granularity: Completing the Puzzle. *Journal of Intelligent Information Systems*, 16:41–63, 2001.

[36] J. Greer and G. McCalla. A computational framework for granularity and its application to educational diagnosis. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 477–482. Morgan Kaufmann, 1989.

[37] S. Jajodia, W. Litwin, and G. Wiederhold. Integrating temporal data in a heterogeneous environment. In A. Tansel et al., editor, *Temporal Databases: Theory, Design and Implementation*, pages 563–579. Database Systems and Applications Series, Benjamin/Cummings Pub. Co., Redwood City, CA, 1993.

[38] S. Jajodia, V. S. Subrahmanian, and X. S. Wang. Temporal modules: An approach toward federated temporal databases. *Information Sciences*, 82:103–128, 1995.

[39] P. Ladkin. The completeness of a natural system for reasoning with time intervals. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 462–467. Morgan Kaufmann, 1987.

[40] L. Lamport. On interprocess communication. Technical Report Research Report 8, SRC, Palo Alto, CA, 1985.

[41] M. Levine, C. Sawka, and D. Bowman. Clinical practice guidelines for the care and treatment of breast cancer: 8. adjuvant systemic therapy for women with node-positive breast cancer (2001 update). *Canadian Medical Association Journal*, 164, 2001.

[42] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovering Frequent Episodes in Sequences. In U. M. Fayyad and R. Uthurusamy, editors, *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, Montreal, Canada, Aug. 1995. AAAI Press.

[43] N. Markey. Past is for free: on the complexity of verifying linear temporal properties with past. In *Proceedings of the International Workshop on Expressiveness in Concurrency (EXPRESS'2002)*, volume 68.2 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science, 2002.

[44] A. Montanari. *Metric and Layered Temporal Logic for Time Granularity*. ILLC Dissertation Series 1996-02, Institute for Logic, Language and Computation, University of Amsterdam, 1996.

[45] A. Montanari and B. Pernici. Temporal reasoning. In A. Tansell, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. Snodgrass, editors, *Temporal Databases: Theory, Design and Implementation*, Database Systems and Applications Series, chapter 21, pages 534–562. Benjamin/Cummings Pub. Co., Redwood City, CA, 1993.

[46] A. Montanari, A. Peron, and A. Policriti. Decidable theories of ω-layered metric temporal structures. *Logic Journal of the IGPL*, 7(1):79–102, 1999.

[47] A. Montanari, A. Peron, and A. Policriti. The taming (timing) of the states. *Logic Journal of the IGPL*, 8(5):681–699, 2000.

[48] A. Montanari, A. Peron, and A. Policriti. Extending Kamp's theorem to model time granularity. *Journal of Logic and Computation*, 12:641–677, 2002.

[49] A. Montanari and A. Policriti. Decidability results for metric and layered temporal logics. *Notre Dame Journal of Formal Logic*, 37:260–282, 1996.

[50] E. Mota and D. Robertson. Representing interaction of agents at different time granularities. In *Proceedings of the International Workshop on Temporal Representation and Reasoning*, pages 72–79. IEEE Computer Society Press, 1996.

[51] M. Niezette and J. Stevenne. An efficient symbolic representation of periodic time. In *Proceeding of the International Conference on Information and Knowledge Management*, volume 752 of *Lecture Notes in Computer Science*, pages 161–168, Baltimore, Maryland, Nov. 1993.

[52] P. Ning, S. Jajodia, and X. S. Wang. An algebraic representation of calendars. *Annals of Mathematics and Artificial Intelligence*, 36:5–38, 2002.

[53] M. Poesio and R. J. Brachman. Metric constraints for maintaining appointments: Dates and repeated activities. In *Proceedings of the National Conference on Artificial Intelligence*, pages 253–259. MIT Press, July 1991.

[54] A. Segev and R. Chandra. A data model for time-series analysis. In *Advanced Database Systems*, volume 759 of *Lecture Notes in Computer Science*, pages 191–212. Springer, 1993.

[55] Y. Shahar. Dynamic temporal interpretation contexts for temporal abstraction. In *Proceedings of the International Workshop on Temporal Representation and Reasoning*, pages 64–71. IEEE Computer Society Press, 1996.

[56] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985.

[57] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Vol. B*, pages 133–191. Elsevier Science Publishers, 1990.

[58] J. Wijsen. Reasoning about qualitative trends in databases. *Information Systems*, 23(7):469–493, 1998.

[59] J. Wijsen. Temporal FDs on complex objects. *ACM Transactions on Database Systems*, 24(1):127–176, Mar. 1999.

[60] J. Wijsen. A string based-model for infinite granularities. In *Proceedings of the AAAI Workshop on Spatial and Temporal Granularity*, pages 9–16. AAAI Press, 2000.