An Efficient Algorithm for Minimizing Time Granularity Periodical Representations

Claudio Bettini Sergio Mascetti
DICo - University of Milan,
via Comelico 39, 20135 Milan, Italy
{bettini,mascetti}@dico.unimi.it

Abstract

This paper addresses the technical problem of efficiently reducing the periodic representation of a time granularity to its minimal form. The minimization algorithm presented in the paper has an immediate practical application: it allows users to intuitively define granularities (and more generally, recurring events) with algebraic expressions that are then internally translated to mathematical characterizations in terms of minimal periodic sets. Minimality plays a crucial role, since the value of the recurring period has been shown to dominate the complexity when processing periodic sets.

1 Introduction

Periodicity is a property used to describe and reason about temporal phenomena, and it has been the subject of research in knowledge representation (see e.g., [8, 12]), in databases, (see e.g., [7, 14, 13, 11]), and in temporal logic (see e.g., [15]). Periodicity can be used, for example, to represent recurring events in a knowledge base, or to store data with a recurring validity period in a database. Periodicity in particular plays a relevant role for the finite representation of time granularities; for example, the set of Mondays can be represented by a specific date for one Monday and by a recurring period of one week (7 days). Considering that months have different lengths, years have leap years and their exceptions, and that granularities also include non standard ones like *banking days* or *academic semesters*, it is quite clear that the recurring period can become large and possibly difficult to identify. Several symbolic formalisms have been proposed in the literature to represent granularities without having the user to explicitly give the formulas defining the periodicity. For example, the *slicing* and *dicing* operators proposed in [6] can be used to represent the set of all second and third week of each month of January by the expression `[2,3]/Weeks:during:January`. Similarly, the expression `all.Years + {2,4}.Months + {1}.Days` $\triangleright$ `2.Days` in the formalism proposed in [10] denotes the first 2 days of February and April of each year. Similar, but more expressive formalisms are defined in [1], which is an extension of [6], and in [9] that introduces a new rich set of algebraic operators.

A mapping from the algebraic expressions to the corresponding mathematical periodic representations is not provided in the above papers. A mapping from the algebra in [9] is provided in [3]. A mapping from the algebras in [6] and [10] is embedded in the proofs of [1]. However, these mappings do not guarantee to return a *minimal representation*, i.e., a representation using the smallest possible period value, and devising such a minimal mapping turns out to be a difficult task. On the other side, the period value greatly affects the performance of operations on periodic sets. For example, the GSTP system for granularity constraint reasoning ([2]) performs constraint satisfaction by applying operations on granularities represented as sets of periodic sets of integers. The complexity is dominated by the least common multiple of the period values of the involved granularities. A minimal periodical representation is not only desirable for processing granularities, but it would also provide a common low-level representation for expressions specified by users in their favorite symbolic formalism.

The technical contribution of this paper is an efficient algorithm to reduce the periodic representation of a time granularity to a minimal representation. An immediate application of the algorithm is in a post-processing step in the conversion from an algebraic specification of a time granularity to its mathematical periodic representation. From a practical point of view, this result allows users to intuitively define granularities (and more generally, recurring events) and having the system processing the underlying periodic sets as efficiently as possible. The algorithm has a worst case complexity of $O(n^{3/2})$ where n is the period of the input granularity representation.

The rest of the paper is organized as follows: in Section 2

we formally introduce time granularities and other notions that are needed to formulate our technical results. In Section 3 we present the algorithm for period minimization, and in Section 4 we show its applications and briefly report on its implementation. Section 5 concludes the paper.

2 Periodic Representation of Granularities

A comprehensive formal study of time granularities and their relationships can be found in [4]. In this paper, for lack of space we only introduce notions that are essential to show our results.

In particular, we report here the notion of *labeled granularity* which was proposed for the specification of a calendar algebra [9] [1].

Granularities are defined by grouping sets of instants into *granules*. For example, each granule of the granularity `day` specifies the set of instants included in a particular day. A label is used to refer to a particular granule. The whole set of time instants is called *time domain*, and for the purpose of this paper the domain can be an arbitrary infinite set with a total order relationship, $\leq$.

Definition 1 *A* labeled granularity *is a pair $(\mathcal{L}, G)$, where $\mathcal{L}$ is a subset of the integers, and G is a mapping from $\mathcal{L}$ to the subsets of the time domain such that for each pair of integers i and j in $\mathcal{L}$ with $i < j$, if $G(i) \neq \emptyset$ and $G(j) \neq \emptyset$, then (1) each element in $G(i)$ is less than every element of $G(j)$, and (2) for each integer k in $\mathcal{L}$ with $i < k < j$, $G(k) \neq \emptyset$.*

When $\mathcal{L}$ is exactly the integers, the granularity is called "full-integer labeled". When $\mathcal{L} = \mathbb{Z}^+$ we have the same notion of granularity as used in several applications (e.g., [4]). However, in general, the set $\mathcal{L}$ of labels can be an arbitrary subset of (possibly noncontiguous) integers and these labels are used to identify granules. Note that each labeled granularity can use a different set of labels.

For example, following this labeling schema, if we assume to map `day`(1) to the subset of the time domain corresponding to January 1, 2001, `day`(32) would be mapped to February 1, 2001, `b-day`(6) to January 8, 2001 (the sixth business day), and `month`(15) to March 2002.

Several interesting relationships can be defined among granularities. The first of these is called *group into* and defines a partial order over the set of all granularities.

Definition 2 *If G and H are granularities, then H is said to* group into *G, denoted $H \trianglelefteq G$, if for each non-empty granule $G(j)$, there exists a (possibly infinite) set S of labels of H such that $G(j) = \bigcup_{i \in S} H(i)$.*

Intuitively, $H \trianglelefteq G$ means that each granule of G is a union of some granules of H. For example, `day` $\trianglelefteq$ `week` since a week is composed of 7 days and `day` $\trianglelefteq$ `b-day` since each business day is a day.

Granularities are said to be bounded when $\mathcal{L}$ has a first or last element or when $G(i) = \emptyset$ for some $i \in \mathcal{L}$. In this paper, for simplicity, we assume that all granularities are unbounded. We assume the existence of an unbounded bottom granularity, denoted by $\perp$ which is full-integer labeled and groups into every other granularity in the system.

Since we are particularly interested in granularities which can be expressed as periodic repetitions of granules of other granularities (in particular a bottom granularity), we formally define the following relationship[2]

Definition 3 *A labeled granularity H groups periodically into a labeled granularity G if H groups into G and there exist positive integers N and P such that (1) for each label i of G, $i + N$ is a label of G unless $i + N$ is greater than the greatest label of G, and (2) for each label i of G, if $G(i) = \bigcup_{r=0}^{k} H(j_r)$ and $G(i + N)$ is a non empty granule of G then $G(i + N) = \bigcup_{r=0}^{k} H(j_r + P)$, and (3) if $G(s)$ is the first non-empty granule in G (if it exists), then $G(s+N)$ is non empty.*

The *groups periodically into* relationship is a special case of the *group-into* relationship characterized by a periodic repetition of the "grouping pattern" of granules of H into granules of G. Its definition may appear complicated but it is actually quite simple. Since H groups into G, any granule $G(i)$ is the union of some granules of H; for instance assume it is the union of the granules $H(a_1), H(a_2), \ldots, H(a_k)$. Condition (1) ensures that the label $i + N$ exists (if it is not greater than the greatest label of G) while condition (2) ensures that, if $G(i + N)$ is not empty, then it is the union of $H(a_1 + P), H(a_2 + P), \ldots, H(a_k + P)$. Condition (3) simply says that there is at least one of these repetitions.

We call the parameters P and N in Definition 3, a *period* and its associated *period label distance*, respectively. We also denote by R the number of granules of G corresponding to each group of P consecutive granules of $\perp$. More formally, R is equal to the number of labels of G greater than or equal to i and smaller than $i + N$ where i is an arbitrary label of G. Note that R is not affected by the value of i.

Note that the period is an integer value. For simplicity we also use the expression "*one* period of a granularity G" to denote a set of R consecutive granules of G.

In general, the periodically-groups-into relationship guarantees that granularity G can be finitely described (in

[1] *Labeled granularities* are an extension of the more standard notion of *granularities* provided in [4].

[2] This is simply an extension to labeled granularities of the analogous relation defined for "regular" granularities (see e.g., [4]).

terms of granules of H), providing the following information: (i) a value for P and N; (ii) the set $\overline{\mathcal{L}}_G$ of labels of G in one period of G; (iii) for each $j \in \overline{\mathcal{L}}_G$, the finite set S_j of labels of H, describing the composition of $G(j)$; (iv) the labels of first and last non-empty granules of G, if their values are not infinite. In the following, we call *explicit granules* the granules that have a label in $\overline{\mathcal{L}}_G$.

A granularity G can have several periodical representations in terms of the bottom granularity; Indeed if P is a period value for G, then any multiple of P is also a period for G. Moreover, for each period value, different sets of consecutive granules can be used to describe the explicit granules contained in one period. Among all possible pairs (P, N) characterizing a periodically-groups-into relationship, there exist a pair (P', N') such that P' is the smallest period value in all pairs. The value P' is called the *minimal period* for the G and any representation adopting that value for the period is called *minimal*. In order to distinguish different representations of the same granularity, the notation "$\bar{G}^1$", "$\bar{G}^2$", ... is used (read "representation 1 of G", "representation 2 of G", ...).

Example 1 *Figure 1 shows how the same granularity can be represented using (1) period equal to* 4 *and the granules labeled with* 1 *and* 2 *as the explicit granules; (2) period equal to* 4 *and the granules labeled with* 2 *and* 3 *as the explicit granules or (3) period equal to* 8 *and the granules labeled with* 2, 3, 4 *and* 5 *as the explicit granules. In the figure, the dotted curly brackets represent the explicit granules.*

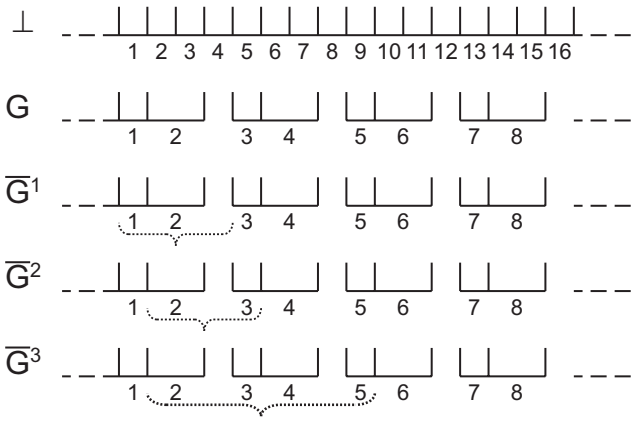

Figure 1. Different periodical representations of the same granularity

2.1 Granularity Conversions

When dealing with granularities, we often need to determine the granule (if any) of a granularity H that covers a given granule z of another granularity G. For example, we may wish to find the month (an interval of the absolute time) that includes a given week (another interval of the absolute time).

This transformation is obtained with the *up* operation. Formally, for each label $z \in \mathcal{L}_G$, $\lceil z \rceil_G^H$ is undefined if $\nexists z' \in \mathcal{L}_H$ s.t. $G(z) \subseteq H(z')$; otherwise, $\lceil z \rceil_G^H = z'$, where z' is the unique index value such that $G(z) \subseteq H(z')$. The uniqueness of z' is guaranteed by the monotonicity of granularities. The notation $\lceil z \rceil^H$ is used when the source granularity can be left implicit (e.g., when we are dealing with a fixed set of granularities having a distinguished bottom granularity).

Another direction of the above transformation is the *down* operation: Let G and H be granularities such that $G \trianglelefteq H$, and z an integer. Define $\lfloor z \rfloor_G^H$ as the set S of labels of granules of G such that $\bigcup_{j \in S} G(j) = H(z)$. This function is useful for finding, e.g., all the days in a month.

3 An Algorithm for Period Minimization

In this section we present an algorithm that, given a granularity representation $\bar{G}^1$, computes the minimal period for G; in Section 3.4 we show how, given this value, a full characterization of a minimal representation can be obtained. The **input** is a periodical representation $\bar{G}^1$ in terms of the bottom granularity; i.e. the period $P_{\bar{G}^1}$, the period label distance $N_{\bar{G}^1}$, and the set of sets S_k with $k = 0 \dots N_{\bar{G}^1} - 1$ such that, for an arbitrary $\alpha \in \mathbb{Z}$, $S_k = \lfloor k + \alpha \rfloor^G$ if $k \in \mathcal{L}_{\bar{G}^1}$, $S_k = \emptyset$ otherwise. Note that, given this representation, the number R of granules in one period can be trivially computed.

3.1 The algorithm

The main idea of the algorithm is that if $\bar{G}^1$ is not minimal, then there exists a minimal representation $\bar{G}^2$ such that $P_{\bar{G}^1} = P_{\bar{G}^2} \cdot m$ with $m \in \mathbb{N}^+$. Therefore, the goal of the algorithm is finding m: once it is found the output is simply $\frac{P_{\bar{G}^1}}{m}$. If $\bar{G}^1$ is minimal, then $m = 1$.

Clearly, m must be a divisor of $P_{\bar{G}^1}$ and we will prove that m must also be a divisor of $N_{\bar{G}^1}$ and $R_{\bar{G}^1}$. Hence, the algorithm first computes the set S of possible values of m, then, for each $n \in S$, it checks if there exists a periodical representation $\bar{G}^3$ such that $P_{\bar{G}^3} = \frac{P_{\bar{G}^1}}{n}$ and $N_{\bar{G}^3} = \frac{N_{\bar{G}^1}}{n}$. Since the value of $P_{\bar{G}^3}$ is inversely proportional to the value of n, the algorithm starts considering the integers in S from the biggest down to the smallest. The execution terminates when the first representation $\bar{G}^3$ is found.

A non trivial part of the algorithm is checking if G can be represented using a period P and a period label distance N. In general, this requires to prove that P and N satisfy the three conditions of Definition 3; however in this particular

Algorithm 1 minimizePeriod

- **Input**: a periodical representation $\bar{G}^1$;
- **Output**: the minimal period for G;
- **Method**:

1: compute the set S whose elements are $gcd(P_{\bar{G}^1}, N_{\bar{G}^1}, R_{\bar{G}^1})$ and its factors.
2: $b := min(\lfloor min(\overline{\mathcal{L}}_{\bar{G}^1})\rfloor^G)$; $t := max(\lfloor max(\overline{\mathcal{L}}_{\bar{G}^1})\rfloor^G)$
3: **for** all $n \in S$, from the biggest down to the smallest **do**
4: $P := P_{\bar{G}^1}/n$; $N := N_{\bar{G}^1}/n$; failed := **false**
5: **for** ($k = b$; $k < t \wedge$ failed=**false**; k++) **do**
6: **if** ($\lceil k \rceil^G$ is undefined) **then**
7: **if** ($\lceil k + P \rceil^G$ is defined) **then** failed := **true**
8: **else**
9: **if** ($\lceil k + P \rceil^G$ is undefined) **then** failed := **true**
10: **else if** ($\lceil k + P \rceil^G \neq \lceil k \rceil^G + N$) **then** failed := **true**
11: **end if**
12: **end for**
13: **if** (failed= **false**) **then** return P **end if**
14: **end for**
15: **return** $P_{\bar{G}^1}$

instance of the problem, it is known that G admits the periodical representation $\bar{G}^1$ and that $\exists k \in \mathbb{N}^+$ s.t. $P = \frac{P_{\bar{G}^1}}{k}$ and $N = \frac{N_{\bar{G}^1}}{k}$. Therefore, it can be derived that the third condition of Definition 3 is always satisfied and the other two conditions are verified if and only if $\forall k \in K$:

$$\lceil k + P \rceil^G = \begin{cases} \text{undefined} & \text{if } \lceil k \rceil^G \text{is undefined} \\ \lceil k \rceil^G + N & \text{otherwise} \end{cases} \quad (1)$$

where $K = \{j \in \mathcal{L}_\perp | min(\lfloor min(\overline{\mathcal{L}}_{\bar{G}^1})\rfloor^G) \leq j < max(\lfloor max(\overline{\mathcal{L}}_{\bar{G}^1})\rfloor^G)\}$. Note that, since $\mathcal{L}_\perp = \mathbb{Z}$, K is an integers interval and, by definition of K and $\overline{\mathcal{L}}$, $|K| \leq P$. Therefore, the problem can be solved by verifying (1) with k ranging on a finite set.

Example 2 *Figure 2 shows granularity G and its non-minimal periodical representation $\bar{G}^1$ (the dotted curly bracket indicates the explicit granules of $\bar{G}^1$). Since $P_{\bar{G}^1} = 12$ and $N_{\bar{G}^1} = R_{\bar{G}^1} = 6$, the algorithm derives $S = \{2, 3, 6\}$. It is not possible to represent G with a period $P = 12/6 = 2$ and a period label distance $N = 6/6 = 1$ since $\lceil 2 \rceil^G$ is defined while $\lceil 2 + P \rceil^G = \lceil 4 \rceil^G$ is undefined. Analogously, it is not possible to represent G with a period $P = 12/3 = 4$ and a period label distance $N = 6/3 = 2$ since $\lceil 2 \rceil^G$ is defined while $\lceil 2 + P \rceil^G = \lceil 2 + 4 \rceil^G$ is undefined. However it is possible to represent G with a period $P = 12/2 = 6$ and a period label distance $N = 6/2 = 3$; For instance, $\lceil 1 \rceil^G = 1$ and $\lceil 1 + P \rceil^G = \lceil 7 \rceil^G = 4 = 1 + N$; $\lceil 4 \rceil^G$ is undefined and $\lceil 4 + P \rceil^G = \lceil 10 \rceil$ is undefined.*

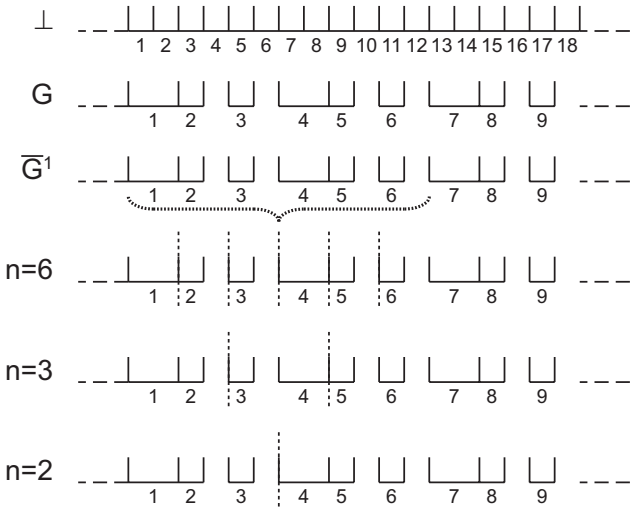

Figure 2. Graphical representation of the granules involved in Example 2

3.2 Correctness

Theorem 1 *Given a periodical representation of granularity G, the algorithm minimizePeriod computes the minimal period for G.*

Proof. To prove the theorem it is necessary to show three intermediate results.

Lemma 1 *Given a granularity G, $\exists \lambda, \lambda' \in \mathbb{R}^+$ s.t. for each representation $\bar{G}^i$ of G, $\frac{P_{\bar{G}^i}}{N_{\bar{G}^i}} = \lambda$ and $\frac{P_{\bar{G}^i}}{R_{\bar{G}^i}} = \lambda'$.*

Lemma 2 *If $\bar{G}^1$ and $\bar{G}^2$ are two periodic representations of granularity G and $\bar{G}^1$ is minimal, then $\exists \alpha \in \mathbb{N}^+$ s.t. (i)$P_{\bar{G}^2} = \alpha P_{\bar{G}^1}$, (ii)$N_{\bar{G}^2} = \alpha N_{\bar{G}^1}$; (iii)$R_{\bar{G}^2} = \alpha R_{\bar{G}^1}$.*

Lemma 3 *Let G be a granularity, $\bar{G}^1$ one of its possible representations and n a positive integer. It is possible to represent G with a period $P = \frac{P_{\bar{G}^1}}{n}$ and a period label distance $N = \frac{N_{\bar{G}^1}}{n}$ if and only if Condition (1) in Section 3.1 is verified.*

Assuming $\bar{G}^1$ is the representation of G given as input to the algorithm, from Lemma 2 follows that if there exists a minimal representation $\bar{G}^2$ s.t. $P_{\bar{G}^2} < P_{\bar{G}^1}$, then $\exists n \in \mathbb{N}^+$ s.t. $n > 1$, $P_{\bar{G}^1} = nP_{\bar{G}^2}$, $N_{\bar{G}^1} = nN_{\bar{G}^2}$ and $R_{\bar{G}^1} =$

$nR_{\bar{G}^2}$. Clearly the set of possible values for n is the set S containing $gcd(P_{\bar{G}^1}, N_{\bar{G}^1}, R_{\bar{G}^1})$ and its factors.

The condition $P = P_{\bar{G}^1}/n$ with $n \in S$ is necessary but not sufficient for the existence of a periodical representation that has P as the period. We now show that the algorithm correctly verifies if there is a periodical representation having $P_{\bar{G}^1}/n$ as the period. For each $n \in S$, the value of the variable *failed* is first set to **false**; then Condition (1) is checked for each $k \in K$, and, if the condition is not satisfied, then the value of *failed* is set to **true**. Therefore if *failed* = **false** when the **for** cycle terminates, then Condition (1) is verified and, by Lemma 3, there exists a periodical representation having $P_{\bar{G}^1}/n$ as period.

Finally we show that it is correct to stop the evaluation of values in S as soon as a valid representation is found. Indeed, let $S' \subseteq S$ be the set s.t. for each $i \in S'$ there exists a representation of G having $P_{\bar{G}^1}/i$ as period. Then the representation having $P = P_{\bar{G}^1}/max(S')$ as period is minimal. Suppose by contradiction that P is not the minimal representation, then $\exists P'$ s.t. $P' < P$ and P' is the period of the minimal representation of G. From Lemma 2 follows that $\exists m, n \in \mathbb{N}^+$ s.t. $P' = \frac{P}{m}$; and $P = \frac{P_{\bar{G}^1}}{n}$; Then $P' = \frac{P_{\bar{G}^1}}{m \cdot n}$. This leads to a contradiction since $m \cdot n \in S'$, $m \cdot n > n$ and $n = max(S')$.

The last step of the algorithm is correct since if it is not possible to identify a representation for any $n \in S$, then the representation given as input is minimal and its period is returned.

□

3.3 Time complexity analysis

Before presenting our formal result on the time complexity of the algorithm *minimizePeriod*, we show how it is possible to perform the *up* operation ($\lceil \cdot \rceil$) in constant time. Indeed, from the explicit granules of G, it is possible to create an array A of size $P_{\bar{G}^1}$ that represents how the granules of $\perp$ are mapped into the explicit granules of $\bar{G}^1$. If $b = min(\lfloor min(\overline{\mathcal{L}}_{\bar{G}^1}) \rfloor^G)$, then, for each $j = 0 \dots P_{\bar{G}^1} - 1$, $A[j]$ = **null** if $\lceil b + j \rceil^G$ is undefined, $A[j] = \lceil b + j \rceil^G$ otherwise[3]. Using this data structure for each i, $\lceil i \rceil^G$ can be computed as $A[j] + \alpha N_{\bar{G}^1}$ where $\alpha = \left\lfloor \frac{i-b}{P_{\bar{G}^1}} \right\rfloor$ and $j = i - b - \alpha P_{\bar{G}^1}$.

Theorem 2 *The worst case time complexity of the algorithm minimizePeriod is $O(n^{\frac{3}{2}})$ where n is the period of the input periodical representation.*

Proof. The set S can be computed in time $O(\sqrt{n})$ by considering each integer from 2 to $\sqrt{n}$ and checking if it is a divisor of P, N and R.

[3]Note that A can be built in time $O(P_{\bar{G}^1})$.

In the worst case, the number of times the algorithm performs the innermost **for** cycle (Algorithm 1, line 5) is $|S|$. By definition of S, it follows that $|S| < d(P_{\bar{G}^1})$ where $d(n)$ indicates the number of divisors of n. The innermost **for** cycle performs, for each k from b to t, two *up* operations ($\lceil \cdot \rceil$). By definition of b and t, it follows that $t - b \leq P_{\bar{G}^1}$, and since the *up* operation can be executed in constant time, the **for** cycle can be performed in time $O(P_{\bar{G}^1})$. As well known in number theory, if $d(n)$ is the number of divisors of n, then, $d(n) < 2\sqrt{n}$. Hence, the **for** cycle is always executed a number of times less than $2\sqrt{P_{\bar{G}^1}}$, then the thesis follows. □

Note that a better upper bound for the dimension of S can be found. Indeed, let $g = gcd(P_{\bar{G}^1}, N_{\bar{G}^1}, R_{\bar{G}^1})$, then $|S| = d(g) - 1$ [4]. Clearly $g \leq P_{\bar{G}^1}$ and, since g is a divisor of $P_{\bar{G}^1}$, $d(g) \leq d(P_{\bar{G}^1})$.

Despite a detailed average-case analysis of time complexity is out of the scope of this paper, note that, in most practical applications of the algorithm, $g << P_{\bar{G}^1}$; Therefore, $d(g) << d(P_{\bar{G}^1})$. Moreover, Theorem 318 in [5] states: "$\sum_{i=1}^{n} d(n) \sim n \ln n$"; hence, in the average case, $|S| << d(P_{\bar{G}^1})$ where $d(P_{\bar{G}^1}) \sim \ln(P_{\bar{G}^1})$.

3.4 Characterization of a minimal representation

Here we show how, given a granularity G, its representation $\bar{G}^1$ and its minimal period P, it is possible to fully characterize a minimal representation $\bar{G}^2$ of G.

Clearly $P_{\bar{G}^2} = P$ and, from Lemma 2, $N_{\bar{G}^2} = N_{\bar{G}^1} \cdot P / P_{\bar{G}^1}$. The set of the explicit granules of $\bar{G}^2$ is the set of granules of G having labels in $\overline{\mathcal{L}}_{\bar{G}^2} = \{i \in \overline{\mathcal{L}}_{\bar{G}^1} | min(\overline{\mathcal{L}}_{\bar{G}^1}) \leq i < min(\overline{\mathcal{L}}_{\bar{G}^1}) + N_{\bar{G}^2}\}$. Since $\overline{\mathcal{L}}_{\bar{G}^2} \subset \overline{\mathcal{L}}_{\bar{G}^1}$, the composition of each explicit granule $G(j)$ with $j \in \overline{\mathcal{L}}_{\bar{G}^2}$ in terms of $\perp$ is the same provided in $\bar{G}^1$. Finally, note that if G is bounded, the value of the bounds is the same independently from the representation.

4 Implementation and applications

In the last years, an application for performing temporal constraint reasoning with granularities has been developed at the University of Milan, and it is currently available as a web service (GSTP, see [2]). The system still misses a user friendly formalism and interface to define new granularities. Recently, the GRC (Granularity Representation Converter) application has been integrated with GSTP; its task is to perform conversions from the Calendar Algebra of [9] to the periodical representation, following the results in [3]. GRC allows users to specify the granularities appearing in the constraints by Calendar Algebra expressions. Since

[4]The "-1" comes from the consideration that the value 1 is not included in S.

the performance of GSTP is strongly affected by the period value of the granularities appearing in the constraints, a central task of GRC is to generate minimal representations. Because the results in [3] do not guarantee minimality, the implementation of the *minimizePeriod* algorithm becomes an essential module of GRC. A stand alone version of the software has been realized too, and it is mainly used for testing the correctness of the implementation and its performance.

The empirical performance results we obtained confirm an almost linear behavior. To give an idea of the actual performance, we consider the representation of a granularity involving leap years and leap year exceptions. In this case, the input representation has a period of 400years in terms of hours (about 3.5millions of hours), and the algorithm runs in less than a second on a standard PC (a Pentium M 1,7 Ghz).

5 Conclusions and future works

We presented an algorithm that, given a periodic representation of a time granularity G, computes the minimal period, and hence provides a minimal representation of G. The algorithm can be used to ensure the minimality of representations that are directly generated by a user or that are the result of a conversion from an algebraic expression.

As a future work, we are planning to implement a graphical user interface that supports the user in the definition of calendar algebra expressions; The *minimizePeriod* algorithm will be used to ensure minimality of the underlying periodical representations. Analogous tools may be developed, by using the same algorithm, for other symbolic formalisms (e.g., [6, 10]). The various tools will have the advantage of interoperability, since they will be based on a common underlying periodic representation.

6 Acknowledgements

This work has been partially supported by Italian MIUR (FIRB "Web-Minds" project N. RBNE01WEJT_005). The authors also wish to thank Lavinia Egidi for her support on some issues in number theory.

References

[1] C. Bettini, R. De Sibi. Symbolic Representation of User-Defined Time Granularities, *Annals of Mathematics and Artificial Intelligence*, 30(1-4):53–92, 2000.

[2] C. Bettini, S. Mascetti. V. Pupillo A system prototype for solving multi-granularity temporal CSP. CSCLP 2004, LNAI 3419, pp. 142–156, Springer, 2005.

[3] C. Bettini, S. Mascetti, X. Wang. Mapping Calendar Expressions into Periodical Granularities. In Proc. of *11th International Symposium on Temporal Representation and Reasoning*, pp. 96–102, IEEE Computer Society, 2004.

[4] C. Bettini, X. Wang, S. Jajodia, Solving Multi-Granularity constraint networks, *Artificial Intelligence*, 140(1-2):107–152, 2002.

[5] G. H. Hardy, E. M. Wright, *An Introduction to the Theory of Numbers*. Oxford University Press, 1960.

[6] B. Leban, D. Mcdonald, and D. Foster, A representation for collections of temporal intervals, in *Proc. of the American National Conference on Artificial Intelligence*, pp. 367–371, AAAI Press, 1986.

[7] F. Kabanza, J.-M. Stevenne, P. Wolper. Handling Infinite Temporal Data. In Proc. of *ACM PODS*, pp. 392–403, 1990.

[8] Robert A. Morris, William D. Shoaff, Lina Khatib. Domain-Independent Temporal Reasoning with Recurring Events. *Computational Intelligence*, 12: 450–477, 1996.

[9] P. Ning, X. Wang, S. Jajodia. An Algebraic Representation of Calendars. *Annals of Mathematics and Artificial Intelligence* 36(1-2): 5–38, 2002.

[10] M. Niezette and J. Stevenne, An efficient symbolic representation of periodic time, in *Proc. of International Conference on Information and Knowledge Management*, pp. 161–168, ACM Press, 1992.

[11] P. Revesz, M. Cai Efficient Querying of Periodic Spatiotemporal Objects, in *Lecture Notes in Computer Science*, Volume 1894, Page 396, Springer, 2000.

[12] Paolo Terenziani, Integrating Calendar Dates and Qualitative Temporal Constraints in the Treatment of Periodic Events. *IEEE Trans. on Knowledge and Data Engineering*, 9(5): 763–783, 1997.

[13] D. Toman, J. Chomicki Datalog with Integer Periodicity Constraints, in *Journal of Logic Programming*, 35:3, 263-90, 1998.

[14] Alexander Tuzhilin, James Clifford. On Periodicity in Temporal Databases. *Information Systems*, 20(8): 619–639, 1995.

[15] Pierre Wolper. Temporal Logic Can Be More Expressive. *Information and Control*, 56(1/2): 72–99, 1983.