Querying Multi-granular Compact Representations

Romāns Kasperovičs and Michael Böhlen

Free University of Bozen - Bolzano,
Dominikanerplatz - P.zza Domenicani 3,
39100 Bozen - Bolzano, Italy

Abstract. A common phenomenon of time-qualified data are temporal repetitions, i.e., the association of multiple time values with the same data. In order to deal with finite and infinite temporal repetitions in databases we must use compact representations. There have been many compact representations proposed, however, not all of them are equally efficient for query evaluation. In order to show it, we define a class of simple queries on compact representations. We compare a query evaluation time on our proposed multi-granular compact representation GSequences with a query evaluation time on single-granular compact representation PSets, based on periodical sets. We show experimentally how the performance of query evaluation can benefit from the compactness of a representation and from a special structure of GSequences.

1 Introduction

A temporal repetition takes place when the same data are associated with multiple time values. Table 1 shows a temporal repetition of a meeting of DB Group that takes place every Monday in January 2005.

When a temporal repetition is infinite (or infeasible large), some finite representer is used to store it in a database. Table 2 shows a representer for the temporal repetition from Table 1 with our proposed compact representation *GSequences*.

The name of GSequences stands for 'granularity sequences', because it consists of finite sequences of periodicities over granularities.

We assume a point-based representation of time, when the time domain is a discrete lineary-ordered set of time points forming the 'bottom granularity'. Additional granularities are partitionings of the bottom granularity defined with functions, allowing non-regular granules, when it is necessary.

A periodicity over a granularity is a five-element tuple, where the first element refers to the granularity and the remaining part defines a periodical repetition over the granularity. For example, periodicity $\langle \text{days}, 2, 1, 10, 20\rangle$ defines a repetition of days described by the function $f(x) = 2x + 1$, where $10 \leq f(x) \leq 20$.

When two or more periodicities are combined into a sequence, each periodicity, except for the rightmost, is related to the following periodicity. We refer the

M.L. Lee, K.L. Tan, and V. Wuwongse (Eds.): DASFAA 2006, LNCS 3882, pp. 111–124, 2006.

Table 1. Temporal Repetition of DB Group Meeting

Group	Room	Time
DB Group	204	2005-01-03 14:00
DB Group	204	2005-01-10 14:00
DB Group	204	2005-01-17 14:00
DB Group	204	2005-01-24 14:00
DB Group	204	2005-01-31 14:00

Table 2. Representer of DB Group Meetings using GSequences

Group	Room	Time
DB Group	204	$(\langle \text{hours}, 1, 0, 14, 16\rangle, \langle \text{days}, 1, 0, 1, 1\rangle, \langle \text{weeks}, 1, 0, *, *\rangle, \langle \text{years}, 1, 0, 2005, *\rangle)$

rightmost periodicity as *absolute* and the rest of periodicities as *relative*. Informally, each relative periodicity is happening *during* each granule of the following periodicy. For example, sequence $(\langle \text{months}, 1, 0, 1, 3\rangle, \langle \text{years}, 1, 0, 2005, 2006\rangle)$ defines first three months during year 2005 and during year 2006.

If the limits of an absolute periodicity are unset, the represented repetition is infinite. For relative periodicities, unset limits imply the limits of a granule of the following periodicity. For example, $\langle \text{months}, 2, 1, 1, *\rangle$ represents an infinite repetition of every second month starting from month 1. Sequence $(\langle \text{hours}, 3, 1, *, *\rangle, \langle \text{days}, 2, 1, 10, 20\rangle)$ is equivalent to $(\langle \text{hours}, 3, 1, 1, 24\rangle, \langle \text{days}, 2, 1, 10, 20\rangle)$, because every day has 24 hours.

There have been many compact representations created during the last two decades [2, 4, 5, 6, 7, 8, 9, 10]. All of them can be used to store temporal repetitions in databases, however, depending on a representation, different performance results might be achieved evaluating the queries. All related works listed do not explore this particular issue.

Many temporal repetitions use common time granularities (e.g., hours, days, years) and periodicity (e.g., every 3rd, every 10th starting from 2nd). As a result, most popular compact representations use periodicity and/or granularities. Works [2, 5, 7, 8, 9, 10] combine multiple granularities in their proposed representation, whereas work [6] uses a single time granularity.

Most of the representations were shown to have the expressiveness equal to eventually periodical sets [5, 6, 7, 8, 10]. An eventually periodical set consists of a finite non-periodical subset and a periodical subset. A periodical set, consequently, is a possibly infinite set, each element of which can be obtained by adding or subtracting positive number p from some other element of this set. For example, eventually periodical set $\{1, 3, 4, 5, 10, 15, 20, \ldots\}$ consists of finite non-periodical subset $\{1, 3, 4\}$ and infinite periodical subset $\{5, 10, 15, 20, \ldots\}$.

Many of compact representations are based on algebraic expressions, where set operations are most common [2, 4, 7, 8, 9, 10]. Both, compact representation values and the relations between granularities, are defined with algebraic expressions. As a result, proposed algorithms assume inductive inference which might badly impact the performance of queries. However, the complexity of algorithms has been estimated only at the theoretical level and the real query evaluation time on different compact representations has never been compared.

Some of the works mentioned address implementation issues, suggest query evaluation algorithms or describe implemented prototypes. Work [3] describes an implementation of the representation proposed in [7] in a real database for use in temporal rules. Work [6] describes algorithms for the evaluation of relational operations on proposed representations. The work [8] describes an efficient algorithm for the evaluation of joins on proposed compact representation. Work [10] describes methods of simplification of representations at the symbolic level. Work [1] describes a simplification algorithm for minimising representations of periodical granularities.

We practically show that not all representations are equally efficient for query evaluation. We use single-granular representation, referred as PSets, with expressiveness equal to eventually periodical sets to compare the query evaluation time with GSequences. We define the compactness property of a representer and we prove that a representer with GSequences is as much or more compact than a representer with PSets of the same temporal repetition. The queries we consider have two boundaries and the target granularity. An example of such a query on a representer shown in Table 2 is 'days with meetings between 2005-03-01 and 2005-05-31'. The experiments we run confirm that the structure and the compactness of GSequences gives an advantage during query evaluation.

Section 2 defines compact representations GSequenses and PSets along with all necessary concepts we use in these definitions. In section 3 we analyse the compactness of both representations. Section 4 defines a class of simple queries on compact representation and gives complexity estimation for query evaluation algorithms on both compact representation. Section 5 contains the results of our experiments. Section 6 finishes this article with conclusions and future work.

2 Compact Representations

2.1 Time Domain and Granularities

All related works we refer assume discrete lineary-ordered time domain. For our representation we use the same assumption.

Definition 1 (time domain). *A time domain T is a discrete, lineary ordered set, infinite in the future and bounded in the past.*

Example 1. Sample time values are $1, 3, 10, 55009440$.

We define granularities as a partitioning of the time domain T.

Definition 2 (granularity). *Let $\mathcal{G} = \mathbb{N}$ be an index set and let $g \in \mathcal{G}$. A mapping $M_g : \mathbb{N} \mapsto 2^T$ is a granularity with an index g if*

1. $\forall i \in \mathbb{N} : (M_g(i) \neq \emptyset)$;
2. $\forall i \in \mathbb{N}$ *: $M_g(i)$ is a finite set;*

3. $\forall i, j \in \mathbb{N} : (i \neq j \Rightarrow (M_g(i) \cap M_g(j) = \emptyset))$;
4. $\forall i, j \in \mathbb{N} : (i < j \Rightarrow (\forall m \in M_g(i), n \in M_g(j) : m < n))$;
5. $\bigcup_i M_g(i) = T$.

The first and the second conditions require all partitions to be non-empty and finite. The third condition disallows partitions to overlap. The fourth and the fifth conditions require that there are no gaps between the partitions.

Example 2. According to Def. 2, proper granularities are days, weeks, Gregorian months, Gregorian years, moon months, milliseconds, centuries, summer and winter time periods, etc. Weekends, leap years, etc., are not granularities, because they allow gaps.

Definition 3 (base granularity). *A granularity $M_b : \mathbb{N} \mapsto 2^T$ is a base granularity iff $\forall i \in \mathbb{N}(M_b(i) = \{i\})$.*

Example 3. If the base granularity is equal to 'days', granularities 'weeks' and 'month' group the time domain T into the partitions as it is shown in Fig. 1.

Our definition of granularity allows us to specify mapping M_g with function $\mu_{g \to T} : \mathbb{N} \to T$, or just μ_g, where $\mu_g(x)$ returns the first element of T of granule x and $\mu_g(x+1) - 1$ returns the last element of T of granule x.

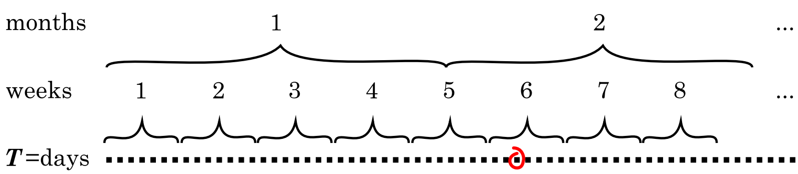

Fig. 1. Granularity as a Partitioning of the Time Domain T

In further sections we use also reverse mapping M_g^{-1} that can be defined with function $\mu_{g \to T}^{-1} : T \to \mathbb{N}$, or just μ_g^{-1}, that returns index $i \in \mathbb{N}$ of a granule of granularity g for given $t \in T$ if $\mu_g(i) \leq t < \mu_g(i+1)$.

Example 4. For the base granularity equal to 'days', function $\mu_{\text{years}}(x)$ returns the first day of year x. Function $\mu_{\text{years}}^{-1}(y)$ returns a year to which day y belongs.

$$
\begin{aligned}
\mu_{\text{years}}(x) &= 365x + \lfloor x/4 \rfloor - \lfloor x/100 \rfloor + \lfloor x/400 \rfloor \\
\mu_{\text{years}}^{-1}(y) &= 400 \lfloor \frac{y}{146097} \rfloor + \\
&\qquad +100 \min(3, \lfloor \frac{y \bmod 146097}{36524} \rfloor) + \\
&\qquad +4 \lfloor \frac{(y \bmod 146097) \bmod 36524}{1461} \rfloor + \\
&\qquad + \min(3, \lfloor \frac{((y \bmod 146097) \bmod 36524) \bmod 1461}{365} \rfloor)
\end{aligned}
$$

A new granularity g can be defined using one already defined granularity h. In other words $\mu_g(x) = \mu_h(\mu_{g\to h}(x))$ and $\mu_g^{-1}(x) = \mu_h^{-1}(\mu_{g\to h}^{-1}(x))$. In this case to define g we define only $\mu_{g\to h}(x)$ and $\mu_{g\to h}^{-1}(x)$.

Example 5. If months are already defined with functions $\mu_{\text{months}}(x)$ and $\mu_{\text{months}}^{-1}(x)$, we can define years with $\mu_{\text{years}\to\text{months}}(x) = 12(x-1)+1$ and $\mu_{\text{years}\to\text{months}}^{-1}(x) = \lfloor x/12 \rfloor + 1$.

2.2 Temporal Repetition and Compact Representation

Let A be some combination of non-temporal domains and let a denote some element of A. Let T be the time domain.

Definition 4 (temporal repetition). *A temporal repetition of some data a is a relation $r_a \subseteq \{a\} \times T$.*

Example 6. Table 3 illustrates a temporal repetition of a bus no. 2 in Bozen-Bolzano. This temporal repetition is infinite, because buses are supposed to go forever.

Table 3. Bus no. 2 Schedule in Bozen-Bolzano

No.	Station	Time
2	Stazione 1	2005-01-03 7:48
2	Stazione 1	2005-01-03 8:00
2	Stazione 1	2005-01-03 8:12
...	...	...
2	Stazione 1	2005-01-04 7:48
2	Stazione 1	2005-01-04 8:00
2	Stazione 1	2005-01-04 8:12
...	...	...

Definition 5 (compact representation). *Let X be some domain. Let $\upsilon : X \to 2^T$ be a function that takes an element of X and returns a subset of time domain T. A compact representation is pair $\langle X, \upsilon \rangle$, where X is called the domain of the representation and υ is called the unfold operation of the representation.*

In sections 2.4 and 2.5 we define two particular compact representations PSets and GSequences, showing the use of this definition.

Definition 6 (relational unfold). *Let $\bar{r}_a \subset \{a\} \times X$ and $\bar{R} = \bigcup_{a,i} \bar{r}_{a,i}$ be a set of all possible $\bar{r}_{a,i}$. Let $r_a \subset \{a\} \times T$ and $R = \bigcup_{a,i} r_{a,i}$ be a set of all possible $r_{a,i}$. A relation operation $\Upsilon : \bar{R} \to R$ is relational unfold operation for the compact representation $\langle X, \upsilon \rangle$, if $\forall \bar{r} \in \bar{R}(\Upsilon(\bar{r}) = \{\langle a, t\rangle \mid \exists x \in X(\langle a, x\rangle \in \bar{r} \wedge t \in \upsilon(x))\})$.*

Definition 7 (representer). *A relation $\bar{r} \subset \{a\} \times X$ is a representer with the domain X of a temporal repetition r if there's such a relational unfold operation Υ, that $\Upsilon(\bar{r}) = r$ and $\bar{r}$ is finite and $|\bar{r}| \leq |r|$.*

Example 7. Let $\bar{r} = \{\langle a, x_1\rangle, \langle a, x_2\rangle\}$ be some representer, where $x_1, x_2 \in X$. Let $\upsilon(x_1) = \{2, 3, 4\}$ and $\upsilon(x_2) = \{7, 9\}$. The result of $\Upsilon(\bar{r})$ is a temporal repetition $\{\langle a, 2\rangle, \langle a, 3\rangle, \langle a, 4\rangle, \langle a, 7\rangle, \langle a, 9\rangle\}$.

Definition 8 (compactness). *The compactness of a representer* $\bar{r} \subset \{a_1\} \times \ldots \times \{a_n\} \times X$ *is its size in bytes occupied in a database.*

Example 8. Having attribute 'Group' as a fixed length character string of length 10, having an attribute 'Room' as a natural number and having granularity indexes encoded by natural numbers, the representer given in Table 2 has a length of $10 + 1 \cdot s + 4 \cdot 5 \cdot s$ bytes, where s is a size of one natural number in bytes.

2.3 Periodical Sets and Periodical Granularities

Definition 9 (periodical set). *Set* $S \subseteq \mathbb{N}$ *is a periodical set if there exists some* $p \in \mathbb{N}$*, called period, and finite subset* $S' \subseteq S$*, called repeating subset, such that:*

1. $\forall i \in S \setminus S'(\exists r \in S', x \in \mathbb{N}(i = r + xp))$;
2. $\forall i \in S(\exists r \in S', x \in \mathbb{N}(i = r + xp) \Rightarrow i \in S \setminus S')$.

The first condition of the definition ensures that all elements of set $S \setminus S'$ can be obtained by consequently adding period p to the elements of subset S' of S. The second condition ensures that S' is 'minimal'. In other words, no element of S' can be expressed subtracting or adding the same period to another element of subset S'.

Example 9. Set $S = \{2, 3, 6, 8, 13, 14, 17, 19, \ldots, 46, 47, 50, 52, \ldots\}$ is a periodical set with repeating subset $S' = \{2, 3, 6, 8\}$ and period $p = 11$.

With the appropriate base granularity many other granularities, including Gregorian calendar granularities, are periodical.

Definition 10 (periodical granularity). *Granularity* M_g *is a periodical granularity if* $\bigcup_{i=1}^{\infty} \mu_g(i)$ *is a periodical set.*

Example 10. If the base granularity is 'days', Gregorian years form a periodical set with a repeating subset of 400 elements (years) and the period equal to 146097 days. Gregorian months form a periodical set with a repeating subset of 400x12 elements (months) and the same period equal to 146097 days.

2.4 PSets

Definition 11 (periodicity). *Let* $\mathbb{N}^* = \mathbb{N} \cup *$*. A periodicity is a five tuple* $\langle g, p, o, l, h\rangle$*, where* $g \in \mathcal{G}$ *is a granularity index,* $p, o \in \mathbb{N}$ *are respectively a period and an offset of a linear function* $f(x) = px + o$ *and* $l, h \in \mathbb{N}^*$ *are respectively lower and upper bounds on the value of* $f(x)$.

Example 11. Sample periodicities are $\langle \text{minutes}, 11, 2, 0, 200\rangle$, $\langle \text{seconds}, 11, 3, 0, *\rangle$ and $\langle \text{days}, 11, 6, *, *\rangle$.

Definition 12 ($X_{\mathbf{PS}}$). *The domain of the compact representation PSets $X_{\text{PS}} = \mathcal{G} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N}^* \times \mathbb{N}^*$ is a set of all possible periodicities.*

Definition 13 ($\upsilon_{\mathbf{PS}}$). *Let $x \in X_{\text{PS}}$ and let $x = \langle g, p, o, l, h\rangle$.*

- *for $l = * \land h = *$:*
  $\upsilon_{\text{PS}}(x) = \{t \in T \mid \exists i \in \mathbb{N}(f = pi + o \land t \in M_g(f))\}$;
- *for $l = * \land h \neq *$:*
  $\upsilon_{\text{PS}}(x) = \{t \in T \mid \exists i \in \mathbb{N}(f = pi + o \land t \in M_g(f) \land f \leq h)\}$;
- *for $l \neq * \land h = *$:*
  $\upsilon_{\text{PS}}(x) = \{t \in T \mid \exists i \in \mathbb{N}(f = pi + o \land t \in M_g(f) \land l \leq f)\}$;
- *for $l \neq * \land h \neq *$:*
  $\upsilon_{\text{PS}}(x) = \{t \in T \mid \exists i \in \mathbb{N}(f = pi + o \land t \in M_g(f) \land l \leq f \leq h)\}$.

Example 12. Table 4 shows a fragment of a representer with PSets of the temporal repetition of the bus no 2 shown in Table 3. The values of the rightmost column encode a periodical set with a period 1 week = 10080 minutes, and repeating subset $\{7665, 7680, 7695\}$, where 7665 corresponds to 07:45 of the first Saturday, 7680 to 08:00, and 7695 to 08:15.

Table 4. Representer of Bus no. 2 Schedule in Bozen-Bolzano using PSets

No.	Station	X_{PS}
2	Stazione 1	minutes,10080,7665,*,*
2	Stazione 1	minutes,10080,7680,*,*
2	Stazione 1	minutes,10080,7695,*,*
...	...	...

2.5 GSequences

Definition 14 ($X_{\mathbf{GS}}$). *Let P be a finite sequence of periodicities. The domain of the compact representation GSequences $X_{\text{GS}} = \bigcup_i P_i$ is a set of all possible P_i.*

To define υ_{GS} we introduce some helper functions. Function $\xi : \mathcal{G} \times \mathbb{N} \times \mathcal{G} \to \mathbb{N}$ takes tuple $\langle e, i, g\rangle$, where e and g are granularity indexes and i is an index of a partition of granularity e, and returns an index of a partition of granularity g:

$$\xi(e, i, g) = \begin{cases} \mu_g^{-1}(\mu_e(i)), \text{ if } \mu_g(\mu_g^{-1}(\mu_e(i))) \geq \mu_e(i); \\ \mu_g^{-1}(\mu_e(i)) + 1, \text{ if } \mu_g(\mu_g^{-1}(\mu_f(i))) < \mu_e(i). \end{cases}$$

Let x be some element of X_{GS}. Let x' denote x without the leftmost periodicity and let $r_1 \in x$ be the leftmost periodicity in x. For example, if $x = (\langle days, 1, 0, 1, 1\rangle, \langle weeks, 1, 0, *, *\rangle, \langle months, 1, 0, 1, 1\rangle, \langle years, 1, 0, 2005, 2005\rangle)$, then $x' = (\langle weeks, 1, 0, *, *\rangle, \langle months, 1, 0, 1, 1\rangle, \langle years, 1, 0, 2005, 2005\rangle)$ and $r_1 = \langle days, 1, 0, 1, 1\rangle$.

Let $\bar{\upsilon} : X_{\mathrm{GS}} \to 2^{\mathcal{G}\times\mathbb{N}}$ be a function and let $\bar{\upsilon}(P)$ be defined as follows.

1. if $(r_1 = \langle g, p, o, l, h\rangle) \wedge (x' = \emptyset)$:
   - for $l \neq * \wedge h \neq *$, $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid l \leq i \leq h \wedge \exists j \in \mathbb{N}(i = pj + o)\}$;
   - for $l = * \wedge h \neq *$, $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid 1 \leq i \leq h \wedge \exists j \in \mathbb{N}(i = pj + o)\}$;
   - for $l \neq * \wedge h = *$, $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid i \geq l \wedge \exists j \in \mathbb{N}(i = pj + o)\}$;
   - for $l = * \wedge h = *$, $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid i \geq 1 \wedge \exists j \in \mathbb{N}(i = pj + o)\}$;
2. if $r_1 = \langle g, p, o, l, h\rangle \wedge (x' \neq \emptyset)$:
   - for $l \neq * \wedge h \neq *$,
     $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid \exists\langle e, k\rangle \in \bar{\upsilon}(x')(\langle g, i\rangle \in \bar{\upsilon}(g, p, o, l + \xi(e, k, g), h + \xi(e, k, g)))\}$
   - for $l = * \wedge h \neq *$,
     $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid \exists\langle e, k\rangle \in \bar{\upsilon}(x')(\langle g, i\rangle \in \bar{\upsilon}(g, p, o, 1, h + \xi(e, k, g)))\}$
   - for $l \neq * \wedge h = *$,
     $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid \exists\langle e, k\rangle \in \bar{\upsilon}(x')(\langle g, i\rangle \in \bar{\upsilon}(g, p, o, l + \xi(e, k, g), \xi(e, k + 1, g)))\}$
   - for $l = * \wedge h = *$,
     $\bar{\upsilon}(x) = \{\langle g, i\rangle \mid \exists\langle e, k\rangle \in \bar{\upsilon}(x')(\langle g, i\rangle \in \bar{\upsilon}(g, p, o, 1, \xi(e, k + 1, g)))\}$

Finally, $\upsilon_{\mathrm{GS}}(x) = \{t \in T \mid \exists\langle b, i\rangle \in \bar{\upsilon}(\langle b, 1, 0, *, *\rangle \cup x)(t = i)\}$, where b denotes the base granularity.

Example 13. Let us take $x = \{\langle weeks, 1, 0, *, *\rangle, \langle months, 1, 0, 1, 1\rangle\}$ as an example. An expression $\bar{\upsilon}(months, 1, 0, 1, 1)$ returns a set of one month $\{\langle \text{months}, 1\rangle\}$. An expression $\bar{\upsilon}(\{\langle weeks, 1, 0, *, *\rangle, \langle months, 1, 0, 1, 1\rangle\})$ returns a set of weeks whose index is between $\xi(\text{months}, 1, \text{weeks})$ and $\xi(\text{months}, 1, \text{weeks})$.

3 Compactness Analysis

As it is shown in [2], periodicity $\langle g, p, o, h, l\rangle$ over periodical granularity g with granularity period p_g and granularity repeating subset S'_g forms periodical set S_f with the period $p_f = p_g$ and repeating subset S'_f, $|S'_f| = |S'_g|/p$, if the $|S'_g|$ is divisible by p.

In other case resulting periodical set S_f has period $p_f = \frac{p_g \cdot \mathrm{LCM}(|S'_g|, p)}{|S'_g|}$ and repeating subset S'_f, $|S'_f| = \mathrm{LCM}(|S'_g|, p)/p$, where LCM stands for the least common multiple.

Example 14. A periodicity $\langle \text{months}, 3, 0, *, *\rangle$ with the granularity period of 146097 days and the granularity repeating subset of 4800 months forms on $T =$ 'days' a periodical set S_f with a period of $p_f = 146097$ days and a repeating subset of $146097/3 = 48699$ elements.

For a periodicity $\langle \text{months}, 11, 0, *, *\rangle$ in the same conditions $p_f = 146097 \cdot 11 = 1607067$ and $|S'_f| = (1607067 \cdot 11)/11 = 146097$ elements.

Lemma 1. *For any representer $\bar{r}_{\mathrm{PS}}$ there's a representer $\bar{r}_{\mathrm{GS}}$ of the same temporal repetition, that is as compact as $\bar{r}_{\mathrm{PS}}$.*

Proof. PSets is a trivial case of GSequences, when a sequence of periodicities contains only one periodicity. Hence, any representer $\langle g, p, o, l, h\rangle \in X_{\mathrm{PS}}$ can be constructed a representer $(\langle g, p, o, l, h\rangle) \in X_{\mathrm{GS}}$ of the same temporal repetition and with the same compactness.

Lemma 2. *For any periodical granularities* $g_1, \ldots, g_m$ *expression* $\bar{\upsilon}_{\mathrm{GS}}$ $((\langle g_1, p_1, o_1, l_1, h_1\rangle, \ldots, \langle g_m, p_m, o_m, l_m, h_m\rangle))$ *returns a periodical set of elements of time domain* T*.*

The idea of the proof of Lemma 2 is that the resulting periodical set has a period equal to the least common multiple of the periods of each $\langle g_i, p_i, o_i, l_i, h_i\rangle$.

Lemma 3. *For any periodical granularities* $g_1, \ldots, g_m$ *representer* $\bar{r}_{\mathrm{GS}} = \{\langle a_1, \ldots, a_n, x\rangle\}$*, where* $x = (\langle g_1, p_1, o_1, l_1, h_1\rangle, \ldots, \langle g_m, p_m, o_m, l_m, h_m\rangle$*, is more compact than a representer of the same temporal repetition with the domain* X_{PS}*, if the size of a periodical subset in* $\bar{\upsilon}_{\mathrm{GS}}$ $((\langle g_1, p_1, o_1, l_1, h_1\rangle, \ldots, \langle g_m, p_m, o_m, l_m, h_m\rangle))$ *is bigger than* m*.*

Theorem 1. *For any representer* $\bar{r}_{\mathrm{PS}}$ *there's a representer* $\bar{r}_{\mathrm{GS}}$ *of the same temporal repetition, which is as compact as* $\bar{r}_{\mathrm{PS}}$ *or even more compact.*

Proof. According to Lemma 1 for each representer with PSets there exists a representer with GSequences which is as compact as the representer with PSets. According to Lemma 3, if there is a periodical granularity g with periodical subset $|S'_g| > 1$, there exists a representer $\bar{r}_{\mathrm{GS}}$ which is more compact than any representer with PSets of the same temporal repetition.

4 Queries on Compact Representations

In this paper we investigate two types of queries on compact representations. The first type of query has a form

$$\pi[\xi(b, \mathrm{time}, g)](\sigma[C_1 \leq \mathrm{time} \leq C_2](\Upsilon(\bar{r}))) , \qquad (1)$$

where where σ is a selection operation, $\bar{r}$ is a compact representation, time is an attribute of $\bar{r}$ of compact representation domain, C_1 and C_2 are some constants of the time domain, π is a projection operation, b is an index of the base granularity, g is an index of some given granularity and $\xi : \mathcal{G} \times \mathbb{N} \times \mathcal{G} \rightarrow \mathbb{N}$ is a granularity convertion function used in previous sections.

The second type of query has a form

$$\sigma[C_1 \leq \mathrm{time} \leq C_2](\Upsilon(\bar{r})) . \qquad (2)$$

This type of queries is a specific case of the first type with $g = b$.

We distinguish two different approaches of the evaluation of queries 1 or 2. The naive approach is first to evaluate an operation $\Upsilon(\bar{r})$ and then to proceed with a regular query on temporal repetition. This naive approach fails in cases when a temporal repetition is infinite, because $\Upsilon(\bar{r})$ never stops.

4.1 Query Evaluation on PSets

The approach we use to evaluate queries on PSets produces a temporal repetition already inside given bounds. The algorithm contains two nested cycles (see Listing 1.1). The outer cycle goes through the tuples in a representer and the inner cycle produces tuples of the resulting temporal repetition. We can evaluate the complexity of this algorithm as $O(\Upsilon_{\mathrm{PS}}) = n^2$.

Listing 1.1. PSets Evaluation Algorithm

```
procedure ϒPS(r̄,C1,C2,g)
   for each tuple=(someA, periodicity) in r̄
      υPS(periodicity,C1,C2,g);
procedure υPS(periodicity,C1,C2,g)
   with periodicity = (e,p,o,l,h) do
     [a;b] = intersection ([l;h],[C1;C2]);
     xmin = minimum argument value of p*x+o in [a;b];
     xmax = maximum argument value of p*x+o in [a;b];
     for x = xmin to xmax
       result = convert(p*x+o→g);
       print result;
```

4.2 Query Evaluation on GSequences

The query evaluation algorithm for GSequences is shown in Listing 1.2. It consists of two procedures. The first procedure goes sequentially through the input tuples and calls the second procedure for each tuple. The second procedure generates the tuples of the resulting temporal repetition recursively going through the sequence of periodicities. The maximal depth of a recursion is equal to the length of the sequence. For each periodicity the procedure runs the cycle through its granularity values. Therefore, we can evaluate the complexity of the algorithm, as $O(\Upsilon_{\mathrm{GS}}) = n^n$.

It seems that the performance of Υ_{GS} should be worse than of Υ_{PS}, however, (1) the number of input tuples for υ_{PS} is normally bigger than for υ_{GS}, in order to produce the same output, (2) we use some specifics from the input to reduce the number of operations in υ_{GS}.

Sequential application of bounds. Because of a sequential structure of GSequences given bounds C_1 and C_2 are applied starting from the rightmost periodicity reducing the range for the following periodicities.

Example 15. Let us take a compact representation shown in Table 2 and a query σ[2005-01-01 00:00 $\leq$ time $\leq$ 2005-01-31 23:59]($\Upsilon(\bar{r})$). Evaluating this query the bounds are applied first to the years and then to the weeks of remaining years only, etc.

Listing 1.2. GSequences Evaluation Algorithm

```
procedure ϒGS(r̄,C1,C2,g)
   for each tuple=(someA,gsequence) in r̄
       υGS(gsequence,C1,C2,g,null);
procedure υGS(sequence,C1,C2,g,parent)
   with sequence=(X,(e,p,o,l,h)) do
       if parent is set then offset = convert(parent→e);
       else offset = 0;
       [a;b]=intersection([l+offset;h+offset],[C1;C2]);
       if parent is set then
           highparent = convert(parent+1→e);
           [a;b]=intersection([a;b],[a;highparent −1]);
       xmin = minimum argument value of p∗x+o in [a;b];
       xmax = maximum argument value of p∗x+o in [a;b];
       for x = xmin to xmax
           if X is empty then
               result = convert(p∗x+o→g);
               print result;
           else if e = g then print result;
           else υGS(X,C1,C2,g,p∗x+o);
```

Hierarchical definition of granularities. We define new granularities using already defined granularities. This allows us to perform granularity convertion operations without going to the base granularity (we avoid getting intermediate results with very big indexes).

Example 16. Let us take a compact representation shown in Table 2 in a database where the base granularity is equal to milliseconds. If both years and weeks are defined or transitively defined in terms of days, an index of the first week of the year can be calculated with the formula $\mu^{-1}_{\text{weeks}\to\text{days}}(\mu_{\text{years}\to\text{days}}(i))$.

Truncating sequences. When the target granularity is present in a sequence of periodicities, we process a sequence only till this periodicity.

Example 17. Let us take a compact representation shown in Table 2 and a query $\pi[\xi(\text{time}, \text{days})](\sigma[\text{2005-01-01 00:00} \leq \text{time} \leq \text{2005-01-31 23:59}](\Upsilon(\bar{r})))$. Since there's a periodicity over days in the sequence of periodicities, the evaluation process stops on this periodicity without processing the remaining periodicities.

5 Experiments

In this section we show the results of three experiments in which we compare the performance of queries to GSequences and PSets representers.

In each experiment the same query (query 1 when the target granularity is given or query 2 otherwise) is applied to GSequences and PSets representers of

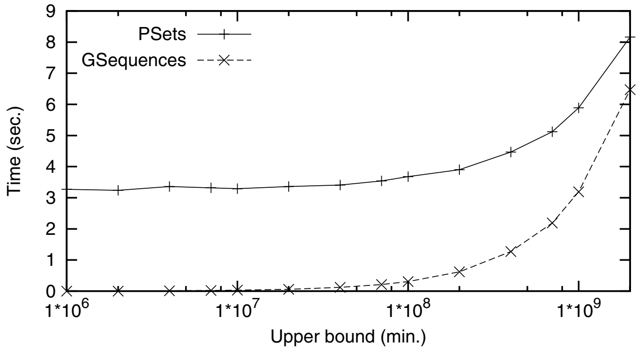

Fig. 2. Results of Experiment 1

the same temporal repetition. According to our assumptions, both representers might have different number of input tuples, but the query results are always identical.

Since it is not very convinient to control the number of output tuples through the query parameters, we compare the query processing time to the given bounds. In other words, horisontal axis in the following plots represents the difference $C_2 - C_1$, or just C_2 (upper bound) with $C_1 = 0$.

In all experiments the base granularity is equal to 'minutes'.

Experiment 1. A GSequences representer of a temporal repetition is illustrated in Table 5. A PSets representer of the same repetition consists of 327,000 tuples with a period equal to 210,378,241 minutes. The results of the experiment show, that the huge size of a representer with PSets gives a big advantage to GSequences.

Table 5. Representer with GSequences for Experiment 1

Some Domain	**Time**
Some Value	$(\langle \text{minutes}, 1, 0, 1, 10\rangle$, $\langle \text{hours}, 1, 0, 1, 10\rangle$, $\langle \text{days}, 1, 0, 1, 10\rangle$, $\langle \text{months}, 11, 0, *, *\rangle)$

Experiment 2. In this experiment we used a real temporal repetition of a bus no. 2 of Bozen-Bolzano. A representer with GSequences consists of 51 tuples. A representer with PSets consists of 456 tuples. From the results of the experiment showed in Fig. 3 it is obvious that the difference in compactness is not sufficient for this kind of queries to beat the difference in complexity of the methods.

Experiment 3. For this experiment we took the same temporal repetition as for Exp.1 and we set the target granularity equal to days. The results illustrated in Fig. 4 show the advantage of truncating sequences.

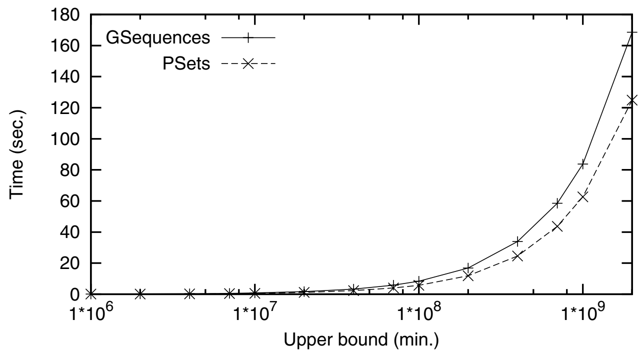

Fig. 3. Results of Experiment 2

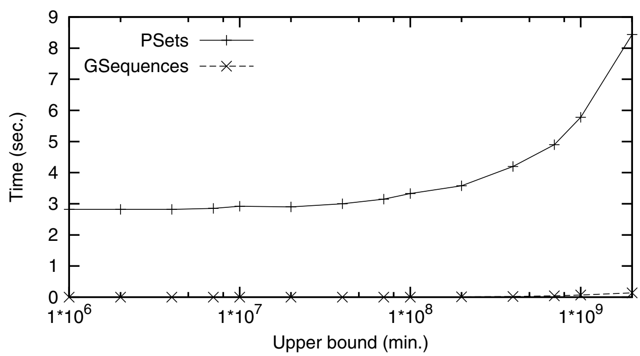

Fig. 4. Results of Experiment 3

6 Conclusions

Compact representations are used to store temporal repetitions in databases. It is essential that compact representations can be queried in the same way as they were temporal repetition, and query evaluation algorithm should take an advantage of querying compact representations.

In this paper we presented a compact representation of temporal repetitions GSequences that combines periodicity with a use of multiple temporal granularities. On this representation we showed that a query evaluation can benefit from a structure of a compact representation. To support this result we experimentally compared GSequences with other compact representation PSets with more simple structure. We introduced the compactness property of compact representations. We proved that besides more sophisticated structure of GSequences it has equal or better compactness than PSets. We also showed and proved experimentally that a compactness of a representer can significantly impact the query evaluation time.

In the future work we aim to implement more complicated queries containing joins and aggregation operations.

References

1. C. Bettini and S. Mascetti. An efficient algorithm for minimizing time granularity periodical representations. In *TIME*, pages 20–25, 2005.
2. C. Bettini and R. D. Sibi. Symbolic representation of user-defined time granularities. In *Proceedings of TIME'99*, pages 17–28. IEEE Computer Society, 1999.
3. R. Chandra, A. Segev, and M. Stonebraker. Implementing calendars and temporal rules in next generation databases. In *Proceedings of the Tenth International Conference on Data Engineering*, pages 264–273, Washington, DC, USA, 1994. IEEE Computer Society.
4. D. R. Cukierman and J. P. Delgrande. The sol theory: A formalization of structured temporal objects and repetition. In *Proceedings of TIME'04*. IEEE Computer Society, 2004.
5. L. Egidi and P. Terenziani. A mathematical framework for the semantics of symbolic languages representing periodic time. In *Proceedings of TIME'04*. IEEE Computer Society, 2004.
6. F. Kabanza, J.-M. Stevenne, and P. Wolper. Handling infinite temporal data. In *PODS*, pages 392–403, 1990.
7. B. Leban, D. D. McDonald, and D. R. Forster. A representation for collections of temporal intervals. In *Proceedings of AAAI'86*, pages 367–371, August 1986.
8. M. Niezette and J.-M. Stevenne. An efficient symbolic representation of periodic time. In *Proceedings of the First International Conference on Information and Knowledge Management*, November 1992.
9. P. Ning, X. S. Wang, and S. Jajodia. An algebraic representation of calendars. *Ann. Math. Artif. Intell.*, 36(1-2):5–38, 2002.
10. P. Terenziani. Symbolic user-defined periodicy in temporal relational databases. *IEEE TKDE*, 15(2), March/April 2003.