CERIAS Tech Report 2009-05
Adaptive Management of Multigranular Spatio-Temporal Object Attributes

by Elena Camossi, Elisa Bertino, Giovanna Guerrini, Michela Bertolotto
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Adaptive Management of Multigranular Spatio-Temporal Object Attributes [*]

E. Camossi[1]   E. Bertino[2]   G. Guerrini[3]   M. Bertolotto[1]

[1] School of Computer Science and Informatics - University College Dublin, Belfield, Dublin 4, Ireland. Phone: +353 (0)1 7162-944/913. Fax: +353 (0)1 2697-262. e-mail: {elena.camossi,michela.bertolotto}@ucd.ie
[2] CERIAS - Purdue University, 250 N. University Street West Lafayette, Indiana, USA 47907-2066. Phone: +1 765 496-2399. Fax: +1 765 494-0739. e-mail: bertino@cs.purdue.edu
[3] DISI - Università degli Studi di Genova, Via Dodecaneso 35, 16146 Genova, Italy. Phone: +39 010 353-6701. Fax:+39 010 353-6699. e-mail: guerrini@disi.unige.it

Abstract. In applications involving spatio-temporal modelling, granularities of data may have to adapt according to the evolving semantics and significance of data. To address such a problem, in this paper we define ST^2_ODMGe, a multigranular spatio-temporal model supporting *evolutions*, which encompass the dynamic adaptation of attribute granularities, and the deletion of attribute values. Evolutions are specified as *Event - Condition - Action* rules and are performed at run-time. The event, the condition and the action may refer to a period of time and a geographical area. Periodic evolutions may be specified, referring to both transaction and valid time dimensions. The evolution may also be constrained by the attribute values. Evolutions greatly enhance flexibility in multigranular spatio-temporal data handling but require revisiting the notion of object consistency with respect to class definitions and access to multigranular object values.

1   Introduction

The ability of representing datasets with respect to both their spatial layout and their historical evolution is crucial when performing analysis and monitoring changes in the spatial configuration of geographical areas. Moreover, approaches able to present data at different granularities [3] represent an effective solution to facilitate information analysis [1].

The granularity to represent information depends on the application task, and on the data domain and semantics. The selection of the appropriate granularity allows the system to store the minimal amount of data thus reducing

storage costs. Often the selection of attribute granularities is based on a trade-off between application efficiency and modelling requirements. Therefore, the model at hand must support the ability to dynamically set and change the spatio-temporal granularity. Current multigranular models only support a static definition of attribute granularities in the database schema. For instance, in a spatio-temporal database for environmental monitoring, the collection of meteorological parameters like the amount of rainfall, the strength and direction of the wind, the value of atmospheric pressure, must be collected more frequently in the presence of exceptional events like hurricanes and storms. Furthermore, such a granularity modification may involve only specific geographical areas (e.g., those affected by the phenomenon), and is required for limited periods of time (e.g., the time when the phenomenon occurs).

In our effort to address these issues we have defined ST^2_ODMGe (Spatio-(Bi)Temporal ODMG supporting Evolutions), a spatio-temporal data model that enables the *evolution* of attributes values, that is, the modification of the granularities used in attribute definitions, and the deletion of attribute values at run-time.

Evolutions reflect modifications about data significance. Such modifications arise for several reasons, including 1) periodic phenomena (e.g., rain and snowfalls usually increase during predetermined seasons); 2) modification of the value of an attribute, or its occurrence (e.g., in monitoring systems); 3) the execution of an operation (e.g., in diagnostic systems); 4) data aging (e.g., older data may be aggregated and then maintained at coarser granularities); 5) privacy restrictions (e.g., individual information on user locations, which are collected in traffic analysis, must be aggregated to coarser granularities in order to become public). Hence, evolutions enhance the flexibility in the management of multigranular spatio-temporal data. They allow one to dynamically modulate adapt the granularities to dynamic events and situations, reflected by updates on the spatio-temporal attribute values and the execution of operations.

In this paper we describe the types of evolutions supported by ST^2_ODMGe. These include: granularity evolution, granularity acquisition, and value deletion. *Granularity evolution* aggregates existing detailed data at a coarser granularity (e.g., older data that may be stored for future reference), or even refines information at a finer granularity (e.g., in data analysis)[1]. By contrast, *granularity acquisition* re-defines the granularity of an attribute changing at run-time the granularity used when inserting new values in the database, whenever the domain conditions change (e.g., sales recording during Christmas). Finally, *value deletion* removes attribute values from the database, whenever they are no longer useful at a given granularity (e.g., detailed data).

The ST^2_ODMGe model design extends our previous multigranularity models ST_ODMG [7] and T_ODMGe [6] with respect to the data definition language, the type system, and multigranular conversions by providing support of both the bitemporal domain and the evolution of spatio-temporal values.

---

[1] This second operation would introduce indeterminacy on converted data, as discussed in the paper.

Granularity evolutions and value deletions had been originally defined for historical data [6], and are herein extended to the spatio-temporal domain. According to our new model, they may be specified and executed at run-time, based on the execution model of active databases, instead of being defined statically in the database schema. Unlike T_ODMGe [6], evolutions of an attribute value may be triggered according to database conditions involving also other attributes, as well as relying on execution of methods, thus making our evolution approach very flexible. Moreover, unlike our previous model [6], where granularity evolutions enable to summarize older data at coarser granularities, in ST^2_ODMGe they may be specified also to refine data at finer granularities. Furthermore, granularity acquisition is a novel feature introduced in ST^2_ODMGe in order to remove one of the major limiting assumption of our previous model [6], that is, that the granularity used for acquiring new data is immutable. ST^2_ODMGe enriches the expressive power of our previous models, and provides a flexible and comprehensive support for run-time modifications of attribute granularities.

In the following, we first discuss related work in Section 2. In Section 3 we introduce the ST^2_ODMGe type system, and the formalisation of objects and classes. Therefore, in Section 4 we address the definition of evolutions for spatio-temporal data, introducing their syntax and discussing their execution model by means of illustrative examples. In Section 5 we investigate how object consistency is affected by evolutions. Indeed, as a consequence of the execution of granularity evolutions and acquisitions, the run-time type of a multigranular object attribute is a Cartesian product of multigranular types at different granularities. We define in Section 6 the access strategies to take advantage of attribute run-time values at multiple granularities. We demonstrate that, under certain assumptions, object access is invariant to the execution of evolutions. In particular, the stored information may be preserved after value deletion, because the same value may be present in the database at a different granularity, and recovered when needed. Furthermore, object access may benefit from evolutions with respect to both effectiveness and efficiency. The values resulting from the execution of granularity conversions are already materialized in the database, thus improving the performance of queries involving aggregates and granularity refinement. The existence in the database of values at different granularities makes it possible to apply two different strategies for object access. Such strategies optimize, respectively, execution efficiency and result accuracy. Finally, Section 7 concludes the paper outlining future research directions.

2 Related Work

ST^2_ODMGe assumes and extends previous work on computing efficiently historical aggregates for the On-Line Analytical Processing (OLAP) of spatio-temporal data streams [16, 13, 12, 11]. Zhang et al. [16] defined a spatio-temporal extension of the *SB-Tree* [15] structure, that, like our previous work [6], proposes an aggregated indexing approach whereby older data are stored using coarser granularities than recent data. Tao and Papadias [13], relying on a seminal work on

aggregate R^*-trees [10], presented over the years several indexing structures for the efficient historical aggregation of spatio-temporal data. Recent work focuses on the issues of aggregates on moving objects trajectories [12, 11]. Unlike those approaches, ST^2_ODMGe supports different time granularities and multiple levels of aggregation and refinement, that is, different indexing forms; moreover the appropriate level can be selected on a per-attribute basis thus supporting different semantics (i.e., different queries). Furthermore, our notion of evolution refers to the bounds of granules at a given granularity, instead of referring to a given amount of time. Moreover, ST^2_ODMGe relies on an agreed notion of temporal granularity [5] that considers granularities as data integrity constraints and formalises how different granularities are related to each other.

The approach to deletion we adopt has been inherited from literature on temporal databases, where data removal is an issue because answers against historical queries must be preserved [8, 14]. Garcia-Molina et al. [8] have addressed data deletion in historical databases by proposing an approach whereby data may be removed (i.e., they *expire*) without affecting related views. A similar approach has been proposed by Toman [14] for historical data warehouses, whereby automatic data deletion is supported by preserving, at the same time, answers to a known and fixed set of first-order queries. This approach assumes that conditions for data evolution are not declared in the schema, rather they are inferred from a given set of queries. Such an approach is adequate if no information about data evolution is known at schema definition time. Only deletions from detailed data are supported. This approach is not exclusive with respect to ours, rather it may complement our work, since conditions for data evolution may be inferred for those attributes for which they are not known at schema definition time.

3 Preliminaries

In this section we illustrate the main characteristics of ST^2_ODMGe, that is, the spatio-temporal dimensions, the granularities formalization, the multigranular type system, and granularity conversions. Moreover, we describe ST^2_ODMGe classes and objects.

3.1 Time, Space, and Granularities

The ST^2_ODMGe model is a 4-dimensional multigranular spatio-temporal model that supports two-dimensional space and two temporal dimensions: *valid time* and *transaction time*. In the following, valid time dimension in ST^2_ODMGe (denoted by $\mathcal{VT}$) refers to the time a fact is true in the reality [9]. Transaction time dimension (denoted by $\mathcal{TT}$) represents the time at which database transactions are executed [9]. Moreover, ST^2_ODMGe supports *two-dimensional space* denoted by $\mathcal{S}$, that refers to the space in which an object is actually located. Spatio-temporal attributes values refer to valid time and to the space dimensions. By contrast, database events, including those triggering the evolutions of

attributes, refer to transaction time. Unlike the valid time dimension, transaction time includes references to the current time denoted by the NOW variable.

In each ST^2_ODMGe database a set of temporal granularities [5] $\mathcal{G}_\mathcal{T}$ and a set of spatial granularities $\mathcal{G}_\mathcal{S}$ are defined. We further distinguish between valid time granularities $\mathcal{G}_{\mathcal{VT}}$ and transaction time granularities $\mathcal{G}_{\mathcal{TT}}$. Temporal and spatial granularities are mappings from an index set $\mathcal{IS}$ to the power sets of the temporal and the spatial domains, respectively. For instance, $days$, $weeks$, $years$ are temporal granularities; $meters$, $kilometers$, $feet$, $yards$, $provinces$ and $countries$ are spatial granularities. Valid time and transaction time are totally ordered. Temporal and spatial granularities in $\mathcal{G}_{\mathcal{VT}}$, $\mathcal{G}_{\mathcal{TT}}$ and $\mathcal{G}_\mathcal{S}$ are used to represent ST^2_ODMGe objects attributes and database events at different levels of detail, with respect to the corresponding dimension.

A (temporal or spatial) *granule* is a subset of a (temporal or spatial) domain corresponding to a single granularity mapping, i.e., given a granularity G and an index $i \in \mathcal{IS}$, $G(i)$ is a granule of G that identifies a subset of the corresponding domain. Granules of the same granularity have disjoint interiors. Moreover, non-empty temporal granules must preserve the order of the temporal domains.

Temporal granules give the temporal bounds of spatio-temporal values: specifically, valid time granules bound attribute values, while transaction time granules bounds database events. Similarly, spatial granules specify the geographical areas where spatio-temporal attribute values are defined. For instance, the value of the daily temperature in Rome may be defined for the first and the second day of January. In this example, the labels "01/01", "02/01", and "*Rome*" may be used to denote two temporal granules at granularity $days$ and one spatial granule at granularity $municipalities$, respectively, that refer to the given time and space.

Granularities differ according to how they partition their domain of reference. Granularities in $\mathcal{G}_\mathcal{T}$ and in $\mathcal{G}_\mathcal{S}$ are related by the *finer-than* transitive relationship [5] and its inverse *coarser-than*. According to this relationship, for example, granularity $days$ is finer-than $months$, which in turn is finer-than $years$ (accordingly, $years$ is coarser-than $months$, which is coarser-than $days$). Likewise, $municipalities$ is finer-than $countries$. The finer-than relationship is denoted by $\preceq$, while $\prec$ denotes the anti reflexive finer-than.

Given two multigranular values, one at granularity G and one at granularity H such that G and H are not directly related by finer-than, such values may be compared if the two values may be represented (i.e., converted) at the same granularity K, that is finer-than G and H. K is chosen as the granularity that minimizes the number of conversions applied. If K is the coarsest, among the granularities finer-than G and H, K is referred to as the *greatest lower bound* (GLB) of G and H.

3.2 Multigranular Types and Conversions

Besides conventional database values, a multigranular spatio-temporal database schema may include multigranular spatial, temporal, and spatio-temporal values. Multigranular values are defined as partial functions from the set of gran-
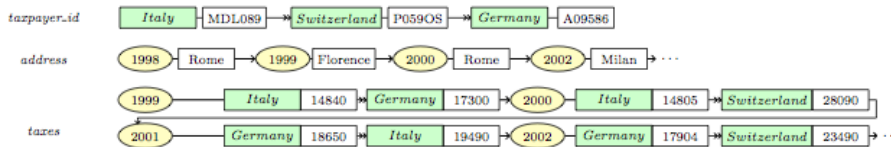
Fig. 1. Examples of multigranular attribute values

ules of the corresponding granularity(ies) to the set of values of the given inner type. Fig. 1 illustrates examples of multigranular attributes: `taxpayer_id` is spatial, with type $Spatial_{countries}(\texttt{string})$; `address` is temporal, with type $Temporal_{years}(\texttt{string})$; finally, `taxes` is spatio-temporal, and its functional type specification is $Temporal_{years}(Spatial_{countries}(\texttt{float}))$.

In a multigranular database data may be converted at different granularities to improve or reduce the level of detail of data. In ST^2_ODMGe, the conversion of multigranular geometrical features is obtained through the compositions of model-oriented and cartographic map generalisation operators that guarantee topological consistency [7], and refinement operators that perform the inverse functions (e.g., *merge*, *split*, *abstraction*, *add feature*). On the other hand, to retrieve, for instance, the annual trend of a phenomenon having a daily representation (e.g., the values of sales in shops located in several countries), also conversion operations for non geometric attribute values are provided (e.g., *average*, *sum*, *selection*, *aggregation*, *restriction*, *split*).

An interesting property for evaluating the correctness of attribute access refers to conversions *invertibility* [4]. When converting a temporal value to a different granularity, and then performing the inverse conversion, we would expect the original value to be returned. Unfortunately, when converting from a finer to a coarser granularity, we loose some details that we cannot usually recover by applying the inverse conversion to the finer granularity. By contrast, when converting from a coarser to a finer granularity, we introduce some details that we should be able to forget; thus we can recover the original value. Given a pair of conversion functions, we denote this pair as *quasi-inverse* or *inverse* functions, according to whether they refer to the first or the second situation, respectively. In the first case, a measurable indeterminacy is introduced. For example, the pair (`avg`,`split`) is quasi-inverse, while the pair (`split`,`sum`) is inverse.

3.3 ST^2_ODMGe Classes and Objects

Given the multigranular type system described above, in the following example we illustrate an ST^2_ODMGe class specification.

Example 1. Given an object type for describing *taxpayers*, reporting the value of taxes paid by a person over time in different countries, its definition will include: a spatial attribute `taxpayer_id` at granularity *countries* to store the fiscal identifiers that the taxpayer holds in different countries; a temporal attribute

`address` at granularity *years* to store the history of his/her fiscal domiciles; and a spatio-temporal attribute `taxes`, defined at temporal granularity *years* and at spatial granularity *countries*, which stores the amount of taxes the taxpayer pays every year in each country where he/she works (for simplicity we suppose the values are stored according to the same currency). □

Given an ST^2_ODMGe class, such as the one described in the previous example, an ST^2_ODMGe object is formally defined as follows.

Definition 1. *(ST^2_ODMGe Object). Given a class c, an ST^2_ODMGe object o of c is defined as a 6-tuple $(id, N, v, c, \Upsilon_{\mathcal{VT}}^{G_{IT}} \times \Upsilon_{\mathcal{S}}^{G_{IS}}, \Upsilon_{\mathcal{TT}}^{G_{IT}})$ where: id is the object identifier, unique in the database; N is the set of object names; v is the object state, given as a tuple of attribute values: $(a_1{:}v_1, \ldots, a_n{:}v_n)$; c is the class to which the object belongs; $\Upsilon_{\mathcal{VT}}^{G_{IT}} \times \Upsilon_{\mathcal{S}}^{G_{IS}}$ is the spatio-temporal object lifespan, represented as set of granules at the temporal chronon and the spatial quantum granularities[2] , with respect to valid dimensions; $\Upsilon_{\mathcal{TT}}^{G_{IT}}$ is the transactional temporal lifespan of the object.* ◇

Example 2. Let o be an object of class `taxpayer` as described in Example 1. According to Definition 1, the values of attributes `taxpayer_id`, `address` and `taxes` in Fig. 1 define a legal object state v for o. An example of spatio-temporal lifespan for o is $G_{IT}(\{1998, 1999, \ldots, 2030\}_{\mathcal{VT}}^{years}) \times G_{IT}(\{1998, 1999, \ldots, NOW\}_{\mathcal{TT}}^{years}) \times G_{IS}(\{Italy, Germany, Switzerland\}_{\mathcal{S}}^{countries})$, where $G(\Upsilon^{G'})$ denotes the conversion of the set of $G'-$granules $\Upsilon^{G'}$ to granularity G. □

4 Evolutions

Evolutions are defined and executed at run time on ST^2_ODMGe objects. They perform three different operations that affect multigranular object attribute values and definitions: *granularity evolution*, *granularity acquisition*, *value deletion*.

Granularity evolutions and acquisitions modify the granularity(ies) of an attribute. The granularity evolution operation, previously introduced by us [6] in a more limited form, allows one to define a new portion of an attribute value, specified at different granularities. The new value (*target*) is obtained by converting values already stored in the database at different granularities (*source*), through the application of granularity conversions. By contrast, granularity acquisitions do not change the database state, but re-define the granularity(ies) that can then be used to insert new attribute values. They have the same effect of a modification of the database schema, likewise an SQL ALTER statement would be executed. Finally, a value deletion eliminates portions of an attribute value at a given granularity.

Evolutions are performed according to the general execution model of active databases. Given an instance of an ST^2_ODMGe database and a set of evolutions specified for it, a continuous monitoring of the database is performed. The execution of database transactions modifies the database state and triggers the

---

[2] These are the finest granularities on the spatio-temporal domain.

evolutions whose events refer to such transactions. Therefore, the corresponding conditions are evaluated. For those triggered evolutions whose conditions evaluate to TRUE, the corresponding actions are executed. An evolution action is a sequence of operations that may modify the attribute granularities and delete the attribute values. As a consequence, the database state (or schema, in case of granularity acquisition) may be modified.

The temporal behaviour of ST^2_ODMGe evolutions differs according to their recurrence. ST^2_ODMGe supports periodic and non-periodic evolutions. *Periodic evolutions* repeat regularly over time. They are triggered by periodic events, or result from the evaluation of periodic conditions. *Non-periodic evolutions* are triggered by extemporary events and conditions on database values. In both cases, evolution elements (i.e., event, condition and action, and all the sub-elements that compose them) may explicitly refer to both the ST^2_ODMGe temporal time dimensions, i.e., transactional and valid time, thus enabling evolutions with different temporal behaviours. Specifically, *temporal events* refer to transaction time, and may be either periodic or not. *Valid time checks* are temporal conditions that refer to valid time.

These different behaviours may be further characterized with the support of *spatio-temporal bounds*. Bounds may apply to each of the elements of an evolution, restricting the occurrence of the evolution event, the evaluation of the condition, and the effects of the action to given temporal periods and geographical areas. As a consequence of the execution of evolutions, the type of an object state in the ST^2_ODMGe model, that is, of the values of their attributes, changes dynamically. Let a be a multigranular attribute defined in class c. In the general case, the run-time type of a is a Cartesian product of multigranular types, as illustrated by the following example.

Example 3. Let o be the identifier of an object of class `taxpayer` described in Example 1. An example state of o is shown in Fig. 2. The example state includes a spatial value for attribute `taxpayer_id`, representing the different fiscal identifiers the contributor has in different countries, and a temporal value representing the history of the contributor residence. The value of attribute `taxes` shown in Fig. 2 is a set of spatio-temporal values at different spatial and temporal granularities. The first value is the value corresponding to the attribute definition, and is given at temporal granularity *years* and at spatial granularity *countries*. The other two values are obtained from this value through granularity evolutions. They are specified at granularities 5*years* and *countries*, and *years* and *ecAlliances* (i.e., economic alliances), respectively. According to the different granularities, they temporally and spatially aggregate the value defined at granularities *years* and *countries*. The domain of attribute `taxes` is thus: $Temporal_{years}(Spatial_{countries}(\texttt{float})) \times Temporal_{5years}(Spatial_{countries}(\texttt{float})) \times Temporal_{years}(Spatial_{ecAlliances}(\texttt{float}))$. □

Evolutions have the form: `ON` *Event* [`IF` *Condition*] `DO` *Action*. An evolution defined on an attribute a is specified on a single value in the Cartesian product that defines the value of a. Each value is referred to as *granularity level*. More precisely, given an object o of class c, the value of attribute a at a given (either
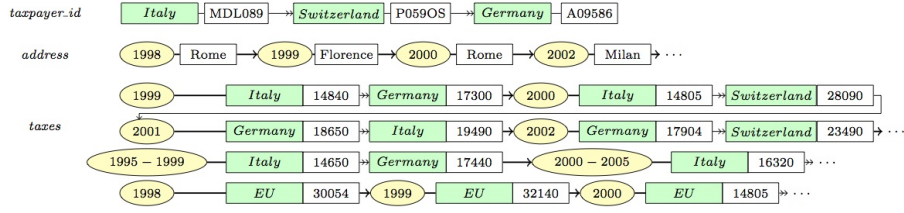
Fig. 2. Example of object state

temporal or spatial) granularity G (at temporal granularity G_t and at spatial granularity G_s, respectively) is referred to as the *granularity level* $< G >$ of a (respectively, the *granularity level* $< G_t, G_s >$ if the attribute is spatio-temporal). Given for instance attribute `taxes` of Example 3, with the object state depicted in Fig. 2, we have three different granularity levels: $< years, countries >$, $< 5years, countries >$, $< years, ecAlliances >$. In the following example we illustrate the syntax to define evolutions.

Example 4. Given class `taxpayer` of Example 1, the following evolution summarises the older record of taxes at a coarser temporal granularity, according to the current Italian fiscal law.

```
ON update taxpayer.taxes< years, countries > during years({1995,1996,...,2014}ᵈᵉᶜᵃᵈᵉˢ_𝒱𝒯)
IF every 5ʸᵉᵃʳˢ_𝒱𝒯
DO evolve < years, countries > to < 5years, countries > using
   avg_{years→5years}, restr_{5years→years}
   in {Italy}^{countries} .
```

The evolution is defined for attribute `taxes`, which evolves from granularity level $< years, countries >$ to granularity level $< 5years, countries >$. It is triggered by the update of the evolution source granularity level, and involves groups of 5 years of data recorded for this level, as specified by the the valid time check `every 5`$^{years}_{\mathcal{VT}}$. The evolution involves only the taxes paid between year 1995 and 2014 in Italy, according to the event temporal bound $years(\{1995, 1996, \ldots, 2014\}\,^{decades}_{\mathcal{VT}})$, and by the action spatial bound `in {Italy}`^{countries}. Note that the spatio-temporal behavior of the evolution is completely specified by the spatio-temporal bounds and the temporal condition. Granularity conversion $average_{years \to 5years}$ is applied for creating the target level, and conversion $restriction_{5years \to years}$, is used to recover the original values whenever these are deleted from the database.

Now suppose that the following evolution is specified from granularity level $< years, countries >$ to granularity level $< years, ecAlliance >$, where $ecAlliance$

is a granularity that represent (non-overlapping) economical alliances among different countries:

```
ON update taxpayer.taxes< years, countries >
IF after 1_{VT}^{years}
DO evolve < years, countries > to < years, ecAlleance > using
    sum_{countries→ecAlliance}, split_{ecAlliance→countries}
    in countries({EC}^{ecAlliance}) .
```

Note that the spatial bound `in countries({EC}`^{ecAlliance}`)` constraints the action execution, and accordingly the evolution aggregates only the tax logs that refer to European Countries. The evolution is executed periodically, according to the valid time check `after 1`$_{\mathcal{VT}}^{years}$.

After some years, because of the evolutions, the state of the attribute is as shown in Fig. 2. □

5 Object Consistency

Consistency is a critical property for data usability. As such it should be formalized and preserved within a database model. Evolutions as defined in ST^2_ODMGe affect the conventional notion of object consistency. Indeed, an object o of a class c, after the execution of evolutions, at run-time may be inconsistent with respect to c, because its state no longer matches the class definition. As a consequence, a new formalization of object consistency is required for ST^2_ODMGe objects, in order to take into account how evolutions modify their state.

In the following, we introduce some preliminary notion to illustrate the consistency conditions of an ST^2_ODMGe object. Then, we formally define the consistency of ST^2_ODMGe objects.

5.1 Preliminary Notions

Let a be a multigranular attribute defined in class c. The granularity levels that compose the value of a are pairwise linked by pairs of quasi-inverse granularity conversions, to form an *acyclic* graph that we refer to as the *granularity levels graph (GLG)* of the attribute, formalised by the following definition. In this definition, and in the rest of the paper, for simplicity we consider a multigranular attribute that refers to either the spatial or the temporal domain, whenever the case of spatio-temporal values may be inferred straightforwardly. Whenever needed, we point out the differences of the spatio-temporal case.

Definition 2. (Granularity level graph - GLG) *Given a set of temporal or spatial granularity levels $< G_i >$ defined for attribute a, where $\forall i = 1 \ldots n$, $G_i \in \mathcal{G}$ is either a temporal or a spatial granularity, the granularity level graph of a, denoted by a_{GLG}, is an acyclic graph (V, E) such that $V = \{< G_1 >, \ldots, < G_n >\}$, and $E = \{ < G_q > \rightarrow < G_r >$, if $G_q \prec G_r$ or $G_r \prec G_q$, and two quasi-inverse granularity conversions $f_{G_q \rightarrow G_r}$ and $g_{G_r \rightarrow G_q}$ have been defined; $1 \leq q \leq n$, $1 \leq r \leq n$ }.* ◇

Similarly, given a spatio-temporal attribute a and the set of its spatio-temporal granularity levels $< G_{t_i}, G_{s_i} >$, and given the granularity conversions defined among these granularity levels through evolution specifications, a GLG is defined for a. Given an attribute a, let a_{GLG} denote its GLG. In the following, the set of nodes and edges of a_{GLG} are denoted by $a_{GLG}.V$ and $a_{GLG}.E$, respectively.

Example 5. Given attribute `taxes` of class `taxpayer` of Example 1, and given the evolutions of Example 4 have been specified. Hence, $taxes_{GLG} = (V, E)$ is specified as follows:

$taxes_{GLG}.V = \{< years, countries >, < 5years, countries >, < years, ecAlliance >\};$
$taxes_{GLG}.E = \{< years, countries > \rightarrow < 5years, countries >,$
$\qquad\qquad < years, countries > \rightarrow < years, ecAlliance >\}.$ □

The following property formalizes the notion that, given two granularity levels in an attribute GLG, it is always possible to compare them, directly or indirectly, through finer-than, even if they are not directly related by the finer-than relationship, and even if the granularity levels are not directly linked through granularity conversions in the GLG. As a consequence, to solve an attribute access, we may navigate among the values defined in the attribute GLG by using the defined granularity conversions, as we will see in the following section.

Property 1. Let $< G_i >$ and $< G_j >$ be two granularity levels in a_{GLG}. Then, one of the following conditions holds:

- $G_i \prec G_j$;
- $G_j \prec G_i$;
- $< G_{GLB(i,j)} > \in a_{GLG}.V$, where $G_{GLB(i,j)}$ is the GLB of G_i and G_j. ▽

We introduce also the concepts of *bottom* and *top* granularities for an attribute in the ST^2_ODMGe model.

Definition 3. (Bottom and Top granularities in a GLG). *$G^\perp$ is the set of the (temporal or spatial) bottom granularities of a, that is the finest granularities of the granularity levels in a_{GLG} for which no granularity G, $< G > \in a_{GLG}.V$ exists such that, $\forall G' \in G^\perp$, $G \prec G'$. Symmetrically, $G^\top$ is the set of the (temporal or spatial) top granularities of a, that is, the coarsest granularities of the granularity levels in a_{GLG} for which no granularity H, $< H > \in a_{GLG}.V$ exists such that, $\forall H' \in G^\top$, $H' \prec H$.* ◇

Example 6. Given attribute `taxes` of Example 1 with the GLG of Example 5, $G^\perp_{\mathcal{VT}} = \{years\}$ and $G^\top_{\mathcal{VT}} = \{5years\}$. Similarly, $G^\perp_{\mathcal{S}} = \{countries\}$, while $G^\top_{\mathcal{S}} = \{ecAlliance\}$. □

5.2 Consistency Conditions for ST^2_ODMGe objects

Relying on attribute GLGs we now revisit the consistency of ST^2_ODMGe objects. We define consistency constraints that are useful to define the access strategies and must be preserved when manipulating object states. Such constraints

are expressed with respect to all the dimensions supported by the ST^2_ODMGe model. To guarantee object consistency, every ST^2_ODMGe attribute value must satisfy the following conditions:

1. each attribute value belongs to the set of legal values of the corresponding type;
2. whenever the attribute value is an object identifier, the referred object exists in the database sometimes during the temporal transactional lifespan of the object;
3. the spatial and/or the temporal domain of the attributes of an object at each granularity level does not exceed the spatial and temporal lifespan of the object; thus for each defined value, the corresponding granule intersects the object lifespan[3];
4. the edges of a GLG must reflect the anti-reflexive finer-than (and coarser-than) relationship holding among granularity levels.

The previous constraints are formalised in Definition 4, which expresses the notion of run-time consistency for objects in an ST^2_ODMGe database.

Definition 4. *(ST^2_ODMGe Consistent Instance). Let o be a ST^2_ODMGe object defined as $(id, N, (a_1 : v_1, \ldots, a_p : v_p), c, \Upsilon_{\mathcal{VT}}^{G_{IT}} \times \Upsilon_{\mathcal{S}}^{G_{IS}}, \Upsilon_{\mathcal{TT}}^{G_{IT}})$. Let c be a class and attr its attribute specification $\{(b_1, \tau_1), \ldots, (b_m, \tau_m)\}$, where $\forall j$, $1 \le j \le m$, b_j be an attribute name and τ_j be an attribute type. Let $\mathcal{LT}$ and $\mathcal{OT}$ be the sets of literal and object types, respectively, and let $\mathcal{T}_{geom}$ be the set of geometric vector types (e.g., point, line, polygon). Let $[\![\tau]\!]$ be the set of legal values for type τ, and $[\![\tau']\!]_i^{G_{IT}}$ be the set of legal values defined for τ' in granule $G_{IT}(i)$. Object o is a consistent instance of c if the following conditions hold:*

1. *$\forall i, 1 \le i \le p, \exists(b, \tau) \in attr$ such that $b = a_i$;*
2. *$\forall(b, \tau) \in attr, \exists k, 1 \le k \le p$, such that $b = a_k$ and the following conditions hold:*

   (a) *if $\tau \in \mathcal{LT}$, $v_k \in [\![\tau]\!]$*

   (b) *if $\tau \in \mathcal{OT} \cup \mathcal{T}_{geom}$, $v_k \in \bigcup_{\Upsilon_{\mathcal{TT}}^{G_{IT}}} \{ [\![\tau]\!]_h^{G_{IT}} \mid h \in \mathcal{IS}\}$;*

   (c) *if τ is a multigranular type at granularity G, all the following conditions hold:*

       i. *$v_k = (v_{k_1}, v_{k_2}, \ldots, v_{k_n})$, with $n \ge 1$*

       ii. *$\forall j, 1 \le j \le n$, such that v_{k_j} is defined,*

           A. *$\exists \tau_j$, where τ_j is a multigranular type at granularity G_j, such that $v_{k_j} \in [\![\tau_j]\!]$;*

           B. *$\forall i \in \mathcal{IS}$ such that $v_{k_j}(i)$ is defined, $G_j(i) \cap (\bigcup_{\Upsilon_{\mathcal{VT}}^{G_{IT}} \times \Upsilon_{\mathcal{S}}^{G_{IS}}} \{G_{IS}(h) \mid h \in \mathcal{IS}\}) \ne \emptyset$;*

       iii. *a granularity (level) graph (V_k, E_k) is defined, such that:*

           A. *$V_k = \{< G_1 >, \ldots, < G_n > \mid v_{k_j} \in [\![\tau_j]\!]$ is defined, with $1 \le j \le n\}$;*

---

[3] Border granules may not be completely included in the object lifespan, but their intersection with it must be non-empty.

$B.$ $E_k = \{< G_q > \rightarrow < G_r >$, if $G_q \prec G_r$ or $G_r \prec G_q\}$, with $1 \leq q \leq n$, $1 \leq r \leq n$. $\diamond$

Example 7. Given object o of Example 3, we assume the evolutions of Example 4 have been executed on $o.\texttt{taxes}$, with $\texttt{taxes}_{GLG}$ as defined in Example 4. Given $G_{IT}(\{1998, 1999, \ldots, 2030\}^{years}_{\mathcal{VT}}) \times G_{IT}(\{1998, 1999, \ldots, NOW\}^{years}_{\mathcal{TT}}) \times G_{IS}(\{Italy, Germany, Switzerland\}^{countries}_{\mathcal{S}})$ the spatio-temporal lifespan of o, if updates on o have been executed after 1998, then object o, with the object state of Fig. 2, is a consistent instance of class $\texttt{taxpayer}$ according to Definition 4. By contrast, it would be inconsistent if its lifespan was $G_{IT}(\{2000, 2001, 2002\}^{years}_{\mathcal{VT}}) \times G_{IS}(\{Italy\}^{countries}_{\mathcal{S}}) \times G_{IT}(\{1998, 1999, \ldots, NOW\}^{G_{IT}}_{\mathcal{TT}})$, because it does not intersect nor the values defined before year 2000, neither the countries different from Italy. $\square$

6 Object Access

In this section we discuss the access to attribute values in $ST^2\_$ODMGe. We consider a basic form of access that requires the attribute value defined in a single granule. The syntax and semantics of such an access are first introduced. We further distinguish two forms of access, qualified and unqualified, depending on a granularity conversion be specified. The strategies to solve these accesses are discussed separately. Therefore, we discuss the invariance of object accesses with respect to evolutions, and characterize unsolvable object accesses.

6.1 Qualified and Unqualified Access

The object access we discuss is formalised by the following definition.

Definition 5. $(ST^2\_$ODMGe object access). *Let o be an object identifier, and let a be the name of an attribute defined for o. If a is a multigranular temporal attribute, let G be a temporal granularity. If a is a multigranular spatial attribute, let G be a spatial granularity. Given a granule label l^G, an* object access *is an expression of the form $o.a \downarrow^{[f]} l^G$, requiring the value of attribute a of object o in granule l^G. If a granularity conversion f is specified, f is applied to compute the access result. The latter access request is referred to as* qualified. *Otherwise it is* unqualified. $\diamond$

The (qualified) access to a multigranular spatio-temporal attribute a is expressed as $o.a \downarrow^{[f]} l^{G_t} \downarrow^{[f']} l^{G_s}$, where G_t and G_s are a temporal and a spatial granularity, respectively, l^{G_t}, l^{G_s} are two granule labels for G_t and G_s, f and f' are granularity conversions.

Example 8. Given the class $\texttt{taxpayer}$ of Example 1 and object o whose state is shown in Fig. 2. $o.\texttt{taxes} \downarrow \{1998\}^{years}_{\mathcal{VT}} \downarrow \{Italy\}^{countries}_{\mathcal{S}}$ is the unqualified access to the payments made during 1998 to the Italian revenue service. By contrast, object access $o.\texttt{taxes} \downarrow \{1998\}^{years}_{\mathcal{VT}} \downarrow^{split[p(x)]} \{Italy\}^{countries}_{\mathcal{S}}$ is the qualified access to the same payments, requiring that application of the refinement function $split[p(x)]$, where $p(x)$ is the probability distribution: $p(x) = \{(Italy, 0.5), (Germany, 0.5)\}$. $\square$

6.2 Solving Unqualified Object Access $o.a \downarrow l^G$

To solve the unqualified object access $o.a \downarrow l^G$ we check whether the requested value is available, i.e., if $< G >$ is a granularity level defined for a and the value of a for o at l^G is defined. If so, such value, that we denote as $o.a_G(l^G)$, is returned, where $o.a_G$ is the granularity level $< G >$ defined for a. Otherwise, the requested value must be computed starting from the values, stored in other granularity levels, that intersect the requested granule.

In the latter case, two different strategies may be applied for solving $o.a \downarrow l^G$, depending on whether the user wants to maximize the accuracy of the result or the efficiency to return it. The *efficiency* maximization strategy minimizes the number of intermediate accesses needed to solve the user access. According to this strategy the application of conversion functions from coarser to finer granularities is preferred to the inverse function, because just one value is required to solve the access. By contrast, when maximizing *accuracy*, the highest precision is required in computing the result. Therefore the application of granularity conversions from finer to coarser granularities takes precedence, because they minimize the indeterminacy in the returned values.

Fig. 3 summarizes the execution strategy for solving $o.a \downarrow l^G$. The strategy to solve the spatio-temporal access $o.a \downarrow l^{G_t} \downarrow l^{G_s}$ follows straightforwardly. We assume that the granularity levels in a GLG are ordered according to the finer-than relationship. Spatio-temporal granularity levels are ordered first according to temporal granularities, and then with respect to spatial granularities. `ACCURATE` denotes that an accurate answer is preferred, whilst efficiency is the default.

```
if ∃o.a_G(l^G) ≠⊥ then return o.a_G(l^G)
else if ACCURATE then
        while ∃o.a_K s.t. K ⪯ G and ∀l_k^K ∈ K(l^G)
            s.t. o.a_K(l_k^K) ≠⊥
            return f_{K→G}(o.a_K)(l^G)
        return null
    else while ∃o.a_H s.t. G ⪯ H
            return g_{H→G}(o.a_H)(l^G)
        while ∃o.a_K s.t. K ⪯ G
            return f_{K→G}(o.a_K)(l^G)
        return null
```

Fig. 3. Algorithm for object access $o.a \downarrow l_G^i$

The computational complexity of the algorithm in Fig. 3 is $O(n)$. Indeed, assuming that the set of granularity levels defined for each attribute value is finite, and the time required for the application of granularity conversions is linear, the complexity of the algorithm is mainly given by the sequential access to a given value in a granularity level. If we assume that indexing is applied on granularity levels (e.g., BTree$^+$ for temporal values and R-Tree for spatial

values), the complexity may improve to $O(log(n))$ if the internal nodes of the R-Tree do not overlap. An optimal worst-case complexity is guaranteed also if the indices for spatial data are, for example, PR-Trees [2].

An important result of our work is thus that the introduction of evolutions does not increase the complexity of the access, with respect to the conventional multigranular case. Furthermore, complexity improves whenever the access involves the application of granularity conversions computation for granularities for which granularity levels are already defined, because the access result is already pre-computed in the database.

In both execution strategies, the access follows an iterative approach, and to solve it we may need to move across several granularity levels. Once a value is found (or a set of values, in the accuracy maximization strategy) that satisfies the access, a sequence of conversions must be performed. If some precomputed value is already available at an intermediate granularity, these values need not to be recomputed, thus improving performance.

Example 9. Given access $o.\texttt{taxes} \downarrow \{1998\}_{\mathcal{VT}}^{years} \downarrow \{Italy\}_{\mathcal{S}}^{countries}$ introduced in Example 8, and object o of class $\texttt{taxpayer}$ whose state is shown in Fig. 2. The access results in $14,650$. $\qquad\square$

6.3 Solving Qualified Object Access $o.a \downarrow^f l^G$

If the access is qualified by a granularity conversion f, this function will be used to compute the access result, taking precedence over the functions already specified in granularity evolutions and acquisitions. Differently from unqualified access, if the accuracy maximization strategy is adopted, an existing value for the specified granule is discarded, if it was constructed with a different function. The value would be used instead by the efficiency maximization strategy. If this value is not defined, we distinguish whether f is a conversion to a coarser granularity (namely, a coercion function, CF), or to a finer granularity.

Fig. 4 reports the algorithm for solving qualified accesses. The spatio-temporal object access $o.a \downarrow^f l_t^G \downarrow^{f'} l_s^G$ follows straightforwardly. As above, $\texttt{ACCURATE}$ denotes that an accurate answer is preferred.

As in the case of unqualified access, the algorithm for qualified access shown in Fig. 4 has computational complexity $O(n)$, which may reach the optimum if indexing is used on the granularity level values as in the previous case.

Example 10. Given the access $o.\texttt{taxes} \downarrow \{1998\}_{\mathcal{VT}}^{years} \downarrow^{split[p(x)]} \{Italy\}_{\mathcal{S}}^{countries}$ of Example 8, with $p(x) = \{(Italy, 0.5), (Germany, 0.5)\}$, and object o of class $\texttt{taxpayer}$ with the object state depicted in Figure 2. When accuracy is required, the access results in 15,027. This value is computed starting from the aggregate value at granularities $< years, ecAlliances >$. $\qquad\square$

6.4 Evolution Invariant Object Accesses

In order to preserve the consistency of the query answers, evolution execution must not affect access results. In what follows, after a preliminary definition

```
if ∃o.a_G(l^G) ≠⊥ then
      if ACCURATE then
            if ∃o.a_K s.t. K ⪯ G
                  and f_{K→G} is defined between o.a_K and o.a_G
                  then return o.a_G(l^G)
      else return o.a_G(l^G)
if f is a CF then
      while ∃o.a_K s.t. K ⪯ G
            if ACCURATE then
                  if ∀l_k^K ∈ K(l^G) s.t. o.a_K(l_k^K) ≠⊥ then
                        return f_{K→G}(o.a_K)(l^G)
                  else return null
            else return f_{K→G}(o.a_K)(l^G)
      return null
else while ∃a_H s.t. G ⪯ H
      return f_{H→G}(o.a_H)(l^G)
```

Fig. 4. Algorithm to solve the qualified object access $o.a \downarrow^f l_G^i$

introducing the notion of evolution invariant access, we show that unqualified object access is invariant with respect to the three forms of evolution discussed in this paper, given a bounded approximation introduced by granularity conversions.

Suppose that $< G >$ is one of the granularity levels defined for attribute a, and suppose that from $< G >$ an evolution has been executed involving granule l^G. In the case of acquisitions we consider the insertion of new values in the target granularity level.

Suppose the evolution has not been performed yet. Assuming that no updates occurred, if the access $o.a \downarrow l^G$ results in the same value when executed just before and just after the evolution execution, the access is referred to as *evolution invariant*. Considering how we build granularity levels, and the specification of granularity conversions, the following result holds.

Proposition 1. *Given a granularity level $< G >$ defined for a, and provided that a granularity level $< G' >$ exists such that $o.a_{G'}(G'(l^G))$ is defined, every object access $o.a \downarrow l_G^i$ is evolution invariant.* ◇

In case of granularity evolutions, in the two cases, i.e., just before and just after the evolution execution, the access will result in the same value if the granularity level $< G >$ is the target level of a granularity evolution defined from $< G' >$. By contrast, for granularity acquisitions and deletions the access is evolution invariant but with a bounded imprecision, which is due to the application of granularity conversions. Indeed, if the value defined for granule l^G is deleted, we can recover it if the value has been involved in a granularity evolution to granularity level $< G' >$. In the case of granularity acquisition, the old and the new acquisition levels are related by a pair of (quasi-inverse) granularity conversions, which guarantees the value consistency among the two levels, modulo a bounded error.

6.5 Unsolvable Object Accesses

In this section we characterize ST^2_ODMGe object accesses that can be statically detected as unsolvable. So far, `null` is returned whenever not enough information is available to solve the access. However, we can distinguish between accesses that are statically known to be unsolvable, that is, for which no database state exists such that these accesses will produce a value different from `null`, and accesses that can produce or not an answer depending on the actual content of the database. Detecting object accesses that are statically unsolvable reduces query execution times, because the system does not need to execute them, but it may return immediately `null`. Given an object access $o.a \downarrow l^G$ (the case of $o.a \downarrow l_t^G \downarrow l_s^G$ follows straightforwardly), the following result holds.

Proposition 2. *Given attribute a defined for an object o, and given value v for a, such that a_{GLG} includes the granularity levels $< G_1 >, \ldots, < G_n >$, the object access $o.a \downarrow l^G$ is unsolvable if one of the following conditions holds:*
  – G is not related by $\preceq$ to any of $G_1, \ldots, G_n$;
  – $G \prec K, K \in G^\perp$;
  – $H \prec G, H \in G^\top$.   ◇

7 Concluding Remarks

In this paper we have investigated issues related to the evolution of multigranular spatio-temporal objects. The approach we propose supports an adaptive management of multigranular spatio-temporal attributes. The main contribution of our research is the definition of ST^2_ODMGe, a multigranular spatio-temporal model that allows the run-time modification of the attribute granularities, and the deletion of attribute values. Evolutions adopt the model of active databases: they are executed at run-time, whenever the specified events occur and the corresponding conditions are satisfied. The events that trigger an evolution may involve also complex conditions on attribute values and their occurrence, as well as the execution of user-defined operations. Differently from our previous temporal evolution model [6], we introduce a new form of evolution, that is, granularity acquisition, and extend granularity evolutions and value deletion to the spatio-temporal domain, enhancing the flexibility of the evolution specification. Our approach to evolutions allows one to model a large variety of situations. As a consequence, consistency constraints on attribute values have been relaxed with respect to the previous version of our model. In the current model, the run-time value of a multigranular attribute is a Cartesian product of multigranular values, linked in a connected acyclic graph through the specification of granularity conversions. Relying on such a structure, the object accesses may be solved according to different strategies and error tolerances.

The ST^2_ODMGe model may be considered as a basis for future investigations on issues involved in evolutions on multigranular spatio-temporal objects. In particular, the development of a prototype of the model will allow us to investigate the trade off between the flexibility, provided by the model, and the consistency that is guaranteed by the statical specification of evolutions.

References

1. G. Andrienko, D. Malerba, M. May, and M. Teisseire. Mining spatio-temporal data. *Journal of Intelligent Information Systems*, 27(3):187–190, 2006.
2. L. Arge, M. de Berg, H. J. Haverkort, and K. Yi. The Priority R-Tree: A Practically Efficient and Worst-Case Optimal R-Tree. In *Proc. of SIGMOD International Conference on Management of Data*, pages 347–358. ACM, 2004.
3. A. Belussi, C. Combi, and G. Pozzani. Towards a Formal Framework for Spatio-Temporal Granularities. In *Proc. of the 15th Int'l Symposium on Temporal Representation and Reasoning*, pages 49–53. IEEE, 2008.
4. E. Bertino, E. Camossi, and G. Guerrini. Access to Multigranular Temporal Objects. In *Proc. of the 6th Int. Conf. On Flexible Query Answering Systems*, number 3055 in LNAI, pages 320–333. Springer-Verlag, 2004.
5. C. Bettini, S. Jajodia, and X. Wang. *Time Granularities in Databases, Data Mining, and Temporal Reasoning*. Springer-Verlag, 2000.
6. E. Camossi, E. Bertino, G. Guerrini, and M. Mesiti. Handling Expiration of Multigranular Temporal Objects. *Special Issue of Journal of Logic and Computation*, 14(1):23–50, 2004.
7. E. Camossi, M. Bertolotto, and E. Bertino. A multigranular Object-oriented Framework Supporting Spatio-temporal Granularity Conversions. *Int'l Journal of Geographical Information Science*, 20(5):511–534, 2006.
8. H. Garcia-Molina, W.J. Labio, and J. Yang. Expiring Data in a Warehouse. In *Proc. of the 24th Int'l Conf. on Very Large Data Bases*, pages 500–511, 1998.
9. C.S. Jensen, C.E. Dyreson, M. Bohlen, J. Clifford, and al. A Consensus Glossary of Temporal Database Concepts. In O. Etzion, S. Jajodia, and S.Sripada, editors, *Temporal Databases: Research and Practice*, number 1399 in Lecture Notes in Computer Science, pages 366–405. Springer-Verlag, 1998.
10. M. Jürgens and H.-J. Lenz. The R_a^*-tree: An improved R^*-tree with Materialized Data for Supporting Range Queries on OLAP-Data. In *Proc. of the 9th Int'l Workshop on Database and Expert Systems Applications*, pages 186–191. IEEE, 1998.
11. B. Kuijpers and A.A. Vaisman. A Data Model for Moving Objects Supporting Aggregation. In *STDM Workshop, IEEE 23rd International Conference on Data Engineering, 2007*, pages 546–554. IEEE, 2007.
12. S. Orlando, R. Orsini, A. Raffaeta, A. Roncato, and C. Silvestri. Spatio-temporal Aggregations in Trajectory Data Warehouses. In *Proc. of 9th Int'l Conf. on Data Warehousing and Knowledge Discovery*, number 4654 in Lecture Notes in Computer Science, pages 66–77. Springer-Verlag, 2007.
13. Y. Tao and D. Papadias. Historical spatio-temporal aggregation. *ACM Transactions on Information Systems*, 23(1):61–102, 2003.
14. D. Toman. Expiration of Historical Databases. In *Proc. of the 8th Int'l Symposium on Temporal Representation and Reasoning*, 2001.
15. J. Yang and J. Widom. Incremental computation and maintenance of temporal aggregates. *The VLDB Journal The International Journal on Very Large Databases*, 12(3):262–283, 2003.
16. D. Zhang, D. Gunopulos, V.J. Tsotras, and B. Seeger. Temporal and spatio-temporal aggregation over data streams using multiple time granularities. *Information Systems*, 28(1-2):61–84, 2003.