Gabriele Pozzani

Modeling and querying spatio-temporal clinical databases with multiple granularities

Ph.D. Thesis

May 6, 2011

Università degli Studi di Verona
Dipartimento di Informatica

Advisor:
Prof. Carlo Combi

Series N°: **TD-03-11**

Università degli Studi di Verona
Dipartimento di Informatica
Strada le Grazie 15, 37134 Verona
Italy

Ουτος ὑμων, ω ανθρωποι, σοφώτατός ἐστιν, οστις ωσπερ Σωκράτης εγνωκεν οτι οὐδενὸς αξιός ἐστι τῃ ἀληθείᾳ πρὸς σοφίαν.

He, O men, is the wisest, who, like Socrates, knows that his wisdom is in truth worth nothing.

Plato, "The Apology of Socrates", 23b

Summary

In several research fields, temporal, spatial, and spatio-temporal data have to be managed and queried with several purposes. These data are usually composed by classical data enriched with a temporal and/or a spatial qualification. For instance, in epidemiology spatio-temporal data may represent surveillance data, origins of disease and outbreaks, and risk factors. In order to better exploit the time and spatial dimensions, spatio-temporal data could be managed considering their spatio-temporal dimensions as meta-data useful to retrieve information. One way to manage spatio-temporal dimensions is by using spatio-temporal granularities (i.e., partitions of a spatio-temporal dimension). This dissertation aims to show how this is possible, in particular for epidemiological spatio-temporal data. For this purpose, in this thesis we propose a framework for the definition of spatio-temporal granularities, with the aim to improve the management and querying of spatio-temporal data. The framework includes the theoretical definitions of spatial and spatio-temporal granularities (while for temporal granularities we refer to the framework proposed by Bettini et al. [23, 27, 215]) and all related notions useful for their management, e.g., relationships and operations over granularities. Relationships are useful for relating granularities and then knowing how data associated with different granularities can be compared. Operations allow one to create new granularities from already defined ones, manipulating or selecting their components.

We show how granularities can be represented in a database and can be used to enrich an existing spatio-temporal database. For this purpose, we conceptually and logically design a relational database for temporal, spatial, and spatio-temporal granularities. The database stores all data about granularities and their related information we defined in the theoretical framework. This database can be used for enriching other spatio-temporal databases with spatio-temporal granularities. We introduce the spatio-temporal psychiatric case register, developed by the Verona Community-based Psychiatric Service (CPS), for storing and managing information about psychiatric patient, their personal information, and their contacts with the CPS occurred in last 30 years. The case register includes both clinical and statistical information about contacts, that are also temporally and spatially qualified. We show how the case register database can be enriched with spatio-temporal granularities both extending its structure and introducing a spatio-temporal query lan-

guage dealing with spatio-temporal data and spatio-temporal granularities. Thus, we propose a new spatio-temporal query language, by defining its syntax and semantics, that includes ad-hoc features and constructs for dealing with spatio-temporal granularities. Finally, using the proposed query language, we report several examples of spatio-temporal queries on the psychiatric case register, showing the "usage" of granularities and their role in spatio-temporal queries useful for epidemiological studies.

Acknowledgments

Come molti miei colleghi studenti di dottorato, pensavo non sarei mai arrivato alla fine del mio dottorato. Invece eccomi qui, dopo aver appreso insieme ai miei compagni d'ufficio un importante insegnamento, a concludere la stesura della mia tesi ricordando chi, in questi anni, mi ha aiutato e mi è stato vicino.

Innanzitutto vorrei iniziare ringraziando colui che mi ha seguito prima, durante e anche oltre il dottorato, il mio supervisore, il Prof. Combi, Carlo. Carlo, vorrei ringraziarti perché, non solo hai avuto fiducia nel mio lavoro e in me, ma soprattutto, dal mio punto di vista, perché sei stato e sei un ottimo mentore. Mi hai seguito, mi hai diretto nel mio lavoro, mi hai a volte "atteso" e mi hai assecondato. Ho iniziato il dottorato che praticamente non sapevo cosa fosse la ricerca e penso che se ora ne ho intravisto (capito mi sembra eccessivo ☺) il significato è soprattutto grazie a te. Ma ci sono due cose per cui vorrei ringraziarti in modo particolare. La prima è l'avermi dato la possibilità, durante questo periodo, di vivere esperienze diverse da quelle del "semplice" dottorato, quali ad esempio l'aiutare nell'organizzazione di una conferenza, il poter gestire un laboratorio, il poter assistere durante le lezioni di laboratorio (e sai quanto la didattica mi piaccia) e tanti altri piccoli e grandi "lavoretti" che mi hanno permesso di apprendere molte cose nuove e diverse e di sentire che stavo dando almeno un piccolo contributo al funzionamento dell'Università. La seconda cosa per cui ti ringrazio maggiormente è l'avermi assecondato nel mio cammino di "crescita". Soprattutto nell'ultimo periodo del dottorato, quando ho dovuto decidere come proseguire la mia carriera, mi hai concesso i miei (lunghi) tempi di indecisione e riflessione, senza forzare alcuna mia scelta ma anzi aiutandomi a capire. E una volta presa la decisione, l'hai assecondata anche se non rispecchia la più classica delle strade nella carriera di un post-doc. Mi hai supportato e mi hai fornito tutte le possibilità per portare avanti la mia scelta (di cui stranamente ☺ sono felice e ancora convinto). Potresti pensare che non hai fatto poi molto e che è il tuo lavoro, ma non è così. Lo dico perché per me è stato molto importante e perché, ascoltando le esperienze di altri dottorandi, non è così scontato che un supervisore si comporti in questo modo. Quindi grazie di tutto quello che hai fatto e che continui a fare.

La seconda persona che vorrei ringraziare è stata innanzitutto un'amica e poi, dal punto di vista temporale, una collega. La mia amica Barbara. Mi hai effetti-

vamente introdotto nel dottorato, parlandomene quando ancora pensavo che non sarei mai uscito vivo dalla laurea specialistica e quando non sapevo minimamente che esistesse il dottorato e poi presentandomi a Carlo. Anche se non abbiamo lavorato molto insieme durante questo periodo, sei sempre stata presente e pronta per scambiare due chiacchiere, ascoltare i miei dubbi, supportarmi e, quando necessario, tirarmi le orecchie ☺. Per me sei stata e rimarrai la mia mecenate.

Vorrei poi ringraziare tutte le altre persone che mi hanno seguito più o meno a lungo nel mio percorso di dottorato. I miei tutor di dottorato, Alberto Belussi e Luca Viganò, che, con i loro consigli e la loro collaborazione, mi hanno aiutato a "camminare" nella direzione giusta.

I would like to thank also Prof. Esteban Zimányi, who hosted me in his Department, giving me the opportunity to work with him and to live a very nice (and relaxing) experience and period in Brussels. I still remember with gladness those three months, and I will remember them forever.

Vorrei ora ringraziare coloro senza cui in questo momento non sarei qui, che mi hanno seguito (e continuano a farlo) per tutta la mia vita, la mia famiglia: mamma, papà e Massimiliano. Nonostante le tante incomprensioni, i miei tanti difetti e mancanze, mi siete sempre stati accanto dandomi la certezza di avere qualcuno a cui rivolgermi quando ne sento il bisogno. Vi voglio bene.

Infine vorrei ringraziare i tanti amici che mi hanno accompagnato (e in alcuni casi mi stanno ancora accompagnando, con mia grande felicità) e mi sono stati accanto durante questo periodo, non tanto dal punto di vista lavorativo ovviamente, ma dal punto di vista personale, che è sicuramente, per me, ben più importante. Tra questi amici, sono contento di annoverare anche tanti colleghi, compagni di ufficio e non, che ho incontrato grazie al dottorato ma con cui si è creato un rapporto che va oltre la semplice attività lavorativa. Siete tanti e non voglio dimenticare nessuno, quindi non citerò tutti i vostri nomi; ma se leggete queste righe consideratevi inclusi ☺ e spero che sappiate già, e che vi abbia già comunicato in altre occasioni, quanto siete importanti per me.

GRAZIE A TUTTI!!!

Contents

1

Introduction

This dissertation deals with the management and querying of spatio-temporal information in databases. In particular, we focus on how temporal, spatial, and spatio-temporal data may be enriched, aggregated, and queried by using spatio-temporal granularities (i.e., partitions of a spatio-temporal domain). For this purpose, we will define two frameworks for the representation of spatial and spatio-temporal granularities and we will show how they can be used for enriching the representation and the query of spatio-temporal databases. These frameworks will focus on some of the most important issues in the management of spatio-temporal databases based on granularities, namely the possibility to represent and query data associated with different granularities (i.e., representing data at different levels). We will follow an approach similar to those proposed by Bettini et al. for defining the framework for temporal granularities [23, 27, 215], but we will extend it taking into account the peculiarities of spatial and spatio-temporal data and dimensions.

This introductory chapter is structured as follow. In next section we will briefly introduce the context of this dissertation, i.e., spatio-temporal databases. Moreover, we will show why the management of spatio-temporal information is important in real-world applications and how spatio-temporal granularities may help in the management of spatio-temporal databases. In Section 1.2 we will describe the original results presented in this dissertation about the management and querying of spatio-temporal data. Finally, in Section 1.3, we will explain how this dissertation is organized and we will summarize where our previous publications have been included in this dissertation.

1.1 Motivation

Human personal, commercial, and industrial activities have always been based on information. For example, merchants always needed to keep information on products, revenues, and expenses, while manufacturers always needed to keep data about warehouses, suppliers, and product components. For this reason, the management of information and data has always been necessary and crucial in human history. With the development of human activities and technologies the importance

of their management grew. Moreover, as it happened to sciences and technologies, that in last century advanced with increasing speed, also the amount of data that are produced by human activities and need to be collected and managed grew. In modern software and applications, the management of large persistent amount of data is crucial, and many users' and companies' activities depend on it. For this purpose, the role of *DataBase Management Systems* (DBMSs) in human activities in crucial, as well as the development of new database technologies answering to new requirements.

Currently, databases and DBMSs are fundamentals and pervade almost any kind of research and end-user applications. For example, DBMSs provide support to bank software for the management of bank transactions as well as to researchers in genetics for the storage of biological information (e.g., DNA sequences and related data). Thus, DBMSs may be capable to store and manage many different kind of data. For this reason, several extensions and technologies for DBMSs have been studied: e.g., biological extensions able to represent genetic data, multimedia data structures for representing audio and video information, spatio-temporal data structures for the management of data qualified with temporal and/or spatial locations. In particular, spatio-temporal data are collected and have to be managed in several different research and application fields, e.g.: epidemiology, biogeography, agriculture, geology, geophysics, urban planning, archaeology, and natural resource management. In this thesis we focus on this last kind of databases: spatio-temporal databases.

In [83] it has been estimated that 80% of the available datasets have a spatial component, and are often related to some temporal aspects. Thus, the management of information provided with temporal and spatial qualifications, i.e., associated with some temporal or spatial dimensions, is very important, and also spatio-temporal information are becoming important since spatial and temporal information are inherently interrelated.

At the end of eighties, Langran identified the need to describe spatial changes over time and studied the design of temporal GIS [133]. Ten years after, Abraham and Roddick reviewed the research on spatio-temporal databases [2] highlighting several research directions and contributions. In the same period, Hornsby and Egenhofer [117] presented an approach to represent and model geographic entities in time. Moreover, in recent years many new proposals about spatial and temporal data to perform statistical analysis, high-level reasoning, and aggregation have been presented [10, 21, 95, 118, 203]. In particular, temporal and spatial information are useful to enable new types of reasoning, aggregation, and management processes. In other words, time and space become meta-data used to reason about traditional data.

The growth of applications using spatio-temporal data requires the development of structures, algorithms, and tools helping in the integrated management of classical, temporal, spatial, and spatio-temporal data. Consider a simple example: the study of the spread of avian influenza since the last years of the nineties up to now. Epidemiologists gather information about influenza cases in the world. Among this information there are geographic locations and time. Locations in time and space help epidemiologists to better understand how the disease evolves in the world and what risk factors (e.g., bird migrations) are linked with the disease.

The temporal and spatial qualification of data is achieved by associating these data with locations on a temporal and/or spatial domain. For example representing the contacts of psychiatric patients with psychiatric services, we may say that a patient contacted a psychiatrist on May 6, 2011: we are associating the patient's contact with a position on the temporal domain. Similarly we can say that a given patient lives in the Verona municipality: we are associating the patient with a position in the space domain.

Usually, in software applications we do not need an infinite precision in temporal and spatial positions: we define and use a minimum level of detail, i.e., the finest partition of the domain we are interested in. The parts of this finest partition (e.g., seconds) are indivisible and, thus, points inside a part are indistinguishable. Other partitions of the same domain can be thus defined starting from the finest one. We call *granularities* these partitions and *granules* the parts of the granularities [124]. Temporal and spatial granularities can represent also the usual temporal and spatial units of measure, e.g., seconds, days, meters, square meters. In these units of measure, the temporal and spatial domains are partitioned in regular and equal-sized granules. However, granularities may represent any partition of the temporal and spatial domains. For example, spatial granularities may represent the subdivision of the Earth surface in continents, countries, or regions with different land usages.

Spatio-temporal granularities allow one to qualify data with a spatio-temporal location. Once data have been qualified by associating them with granules, granularities can support also the retrieving of qualified spatio-temporal data. Data can be aggregated according to granules they belong to, allowing one to conduct studies and to draw up statistics on aggregated data. Moreover, granularities can be used for restricting data to be retrieved with respect to temporal, spatial, and spatio-temporal constraints. For this purposes, a spatio-temporal query language have to be defined for supporting spatio-temporal granularities.

It is clear that many different granularities can be defined and used, eventually at the same time, to qualify data. Thus, a basic issue in the management of spatio-temporal data through granularities is the possibility to represent data associated with different granularities. For example, in a database representing psychiatric patients' data, patients' contacts may be represented at day level, while the patients' profession may be represented at month level. Systems capable to represent and deal with data associated with different granularities are called *multigranular systems*.

Multigranular systems must be able to represent, manage, and compare data associated with different granularities. In particular, the ability (1) to compare data associated with different granularities, and (2) to transform data associated with a granularity to equivalent data associated to a different granularity, is based on the knowledge of the system of the relationships between the involved granularities. For example, car accidents in regions can be obtained just summing car accidents in municipalities that partition regions. However, this is possible only because regions are equal to the union of municipalities (i.e., municipalities group into regions). When this is not true, i.e., other relationships hold or we know that some relationships do not hold, data may be calculated, queried, and aggregated in a different way (e.g., by using a sum operation weighted with respect to the per-

centage of extent shared by the granules). Thus, different calculations are adopted in different cases according to relationships holding between involved granularities. Knowledge about relationships between involved granularities allows one to know how data associated to these granularities can be compared.

On the other hand, as we mentioned before, new granularities can be defined from the finest one or, in general, from already defined granularities. This need requires the definition of a set of operations over granularities. For example, a new granularity may be calculated from another one by grouping together their granules according to a given rule.

All previous issues have been addressed in the temporal context by several proposals dealing with notions related to temporal granularities [23,54,66,154,220]. In general, temporal granularities represent any partition of a time domain in sets of points that can be used for qualifying classical data. For example, `Days` is a temporal granularity partitioning the time line in sets of time points (i.e., granules), each one representing the set of instants belonging to a day. The notion of temporal granularity has been developed by Bettini et al. in the last years of nineties [23,27]. Bettini et al.'s framework for granularity has been completed with a calendar algebra [154] for dealing with granularities. Calendar algebra defines relationships and operations for managing temporal granularities. Moreover, several alternative approaches to the algebraic one have been proposed. They mainly propose string-based [220], automata-based [65, 66] and logical formalisms [54, 87, 88, 148] for representing temporal granularities. Temporal granularities have been used in clinical context for representing and reasoning on temporal aspects of clinical guidelines [55], for modeling and managing clinical information [59–61, 128, 186], and for querying clinical databases [52, 58].

On the other hand, in the spatial context, the research community has not reached yet a widely accepted definition of granularity, even though a sound notion of spatial granularity may be useful to manage spatially qualified information. At the best of our knowledge, the notion of spatial granularity has not been deeply investigated till now. Only few papers in literature deal with spatial granularity [36,181,193,197,214]. Each one proposes a different definition and someone uses this term to represent other notions, e.g., spatial resolution or precision. Furthermore, none of these proposals define a complete framework including properties, relationships, and operations.

The same lack can be found in the spatio-temporal context. At the best of our knowledge, there are only few proposals about spatio-temporal granularity in literature [34,37,214], and some of these use term "multi-granularity" to represent the different notion of "multi-resolution" [29, 81].

Since there are many applications dealing with spatial and spatio-temporal data, these lacks and their filling can affect the development of new applications and approaches in several research fields.

This thesis aims to fill these gaps in the definition of spatial and spatio-temporal granularities and to show how they can be used in retrieving spatio-temporal data. For this last purpose, we need to apply our approach to a real spatio-temporal database.

There are many examples of applications dealing with spatio-temporal data. Actually, most applications related to spatial information and granularities can

be extended considering also time. As a matter of fact, it is possible to consider temporal qualification when a spatial or physical survey is repeated more times and to use it to make spatio-temporal reasoning. In particular, we will focus on spatial and spatio-temporal epidemiology.

Spatial epidemiology is a growing research area inside the medical research field [165]. Epidemiology studies the factors affecting the health and illness of populations (e.g., psychiatric, cardiovascular, respiratory, or genetic diseases), and serves as the foundation and logic of interventions made in the interest of public health and preventive medicine. Spatial epidemiology extends traditional ecological studies with the description and analysis of geographic variations in diseases with respect to several social risk factors (e.g., demography, environment, behavior, socioeconomy, genetics, infectious diseases) [4, 17, 75]. In particular, in epidemiology spatial and temporal information plays an important role, for example, in ecological or aggregate studies where the subjects of observations are not individuals but groups of people [4, 212]. In this context, spatial and temporal granularities are used in multiple-group and time-series studies [4, 151], respectively, i.e., granularities are used for spatio-temporally aggregating clinical data (e.g., disease surveys) and for representing environmental properties or factors (e.g., air pollution and traffic density) that may be related to diseases evolution. In particular, spatial data can describe, for example, landscape constraints, spatial associations of risk factors and disease, and origins of disease and outbreaks. In other words, geography represents the spatial context and character in which health risks occur.

However, as we mentioned before, epidemiology is not tied only to space, but also to time. Health status and risk factors vary also across time [1, 70]. Changes in data over time are useful to understand the temporal evolution of risks and diseases. This approach allows one to study where and how risks and diseases start and to predict their future evolution. For this purpose, it is useful to study temporal trends of data provided with spatial qualification, i.e., trends of spatio-temporal data.

While spatial analyses in epidemiology have been already studied, the use of spatio-temporal models has been only recently investigated [1] and in particular, at the best of our knowledge, there are no formal proposals for supporting spatio-temporal databases in medical and epidemiological contexts.

Moreover, note that spatial and spatio-temporal epidemiology have to deal with the issues related to multi-granularity we introduced before. Health risks are often mapped to relatively arbitrary administrative areas (the level at which population and disease data are available), but risks can be sensitive to changes in the considered granularities. It is known as the "modifiable area unit problem" [157]. It is a fundamental problem in spatial analysis since there are numerous ways (i.e., granularities) to partition space and time and data are usually presented at a particular granularity, but solutions depend on the chosen granularity. For these reasons, investigators need to use and cross-check several different granularities, determining the most appropriate geographic and temporal ones.

1.2 Our contribution

Taking into account all the considerations mentioned in the previous section about spatio-temporal granularities and the related issues, the specific goals of the thesis are:

1. to formally define the notion of spatial and spatio-temporal granularities in such a way to address related issues: the ability to compare data associated with different granularity and the capability to define new granularities from already defined ones;
2. to study how to represent the defined frameworks in relational DBMSs in order to allow one to use them for enriching existing spatio-temporal databases;
3. to show how to qualify data in relational DBMSs by using temporal, spatial, and spatio-temporal granularities;
4. to show how spatio-temporal data can be queried by exploiting temporal, spatial, and spatio-temporal granularities;
5. to apply the proposed approach to a real spatio-temporal database, showing how it can be enriched with granularities and how data stored in it can be retrieved with respect to spatio-temporal requirements based on granularities;
6. to study an inference system for relationships between granularities for avoiding the execution of potentially computational heavy algorithms for the evaluation of relationships.

In order to exemplify the notions and approaches we will propose, we introduce a spatio-temporal clinical database for the management of the Verona Community-based Psychiatric Service (CPS). However, we note that the proposed notions can be applied for the management of any spatio-temporal database. The introduced database (PCR) stores information about psychiatric patients and their contacts with the psychiatric services. The database contains both temporal and spatial data. For example, patients' personal data may change over time thus they are temporally qualified as well as the contacts between patients and the services. Moreover, patients' residence and domicile are spatially qualified as well as the contact location. The database is currently used by the CPS for both administrative and research activities and contains data collected in last 30 years. Thus it provides a good basis for applying our approaches and notions.

We will propose a theoretical framework for spatial granularities, following the same approach used by Bettini et al. for defining temporal granularities [23,27,215]. In the used approach, granularities are meta-data that, when associated to classical data, help to manage and query data. In other words, granularities can be used for qualifying and enriching other spatio-temporal data in order to exploit their spatio-temporal information. The framework we will propose includes a formal definition of spatial granularity that allow us to represent both the geometries (i.e., the granules) defining the partition of the spatial domain and relationships between the granules. Indeed, conversely to temporal granularities where granules are ordered by using the usual implicit order defined on time points, spatial points and granules have not an implicit and unique order. On spatial points and granules many different orders can be defined and these may be represented in the granularities. We note that the explicit representation of relationships between granules has not been considered in previous proposals about spatial granularities.

Conversely to previous proposals in literature, we complete the framework for spatial granularities defining also a set of relationships and operations on them. Relationships allow us to know how granules in different granularities are related (e.g., they intersect or are contained each other), permitting also to know how data associated to these different granularities may be compared. We define a more complete set of relationships with respect to previous proposals in literature, where at most two relationships have been defined (see Section 4.1). Operations on spatial granularities allow one to create new granularities from already defined ones, for example by merging together some granules of a given granularity. Conversely to previous proposals, where operations are not considered, we include in the framework for spatial granularities also several operations and will show how they can be used for defining granularities useful for the management of the PCR.

Spatial data can be represented by using two different, but related, models: the vector model and the raster model. The choice between raster data model and vector data depends on the kind of information we have to represent. In general, raster data are more suited to environmental applications while vector data are more suited to human activity [187]. Raster systems model complex spatial patterns with limited attributes, such as land-use patterns very well, while the vector data model is better for more clearly dened space with complex attributes, such as census data. Both are used in several application, for this reason we defined our frameworks for spatial and spatio-temporal granularities for both models. The definitions use two different formalisms and thus are quite different, but they represent the same idea and concepts.

We define a framework for spatio-temporal granularities by using the same approach we use for spatial granularities. Spatio-temporal granularities represent the evolution over time of spatial granularities. For this reason, they merge together the notions of temporal granularity proposed by Bettini et al. [23,27,154] and our notion of spatial granularity. Our definition of spatio-temporal granularity overcomes some limitations of previous proposals by associating a spatial granularity to each time point at which it is valid. Moreover, similarly to the framework for spatial granularity, our framework for spatio-temporal granularities includes the definition of relationships and operations over them. Previous proposals in literature did not include them. The framework and all related notions have been defined on both the vector and the raster models.

Based on the framework for temporal granularities proposed by Bettini et al. [23,27,154] and our frameworks for spatial and spatio-temporal granularities, in this dissertation we propose an inference system for inferring the relationships that hold between temporal, spatial, and spatio-temporal granularities. With respect to previous proposals that deal only with relationships between temporal, spatial, and spatio-temporal data, our inference system deals with relationships among granularities. The use of the inference system allows to avoid the execution of potentially computational heavy algorithms for evaluating the relationships. The proposed inference system comprises a set or rules, each one with a premise and a conclusion. Each rule has a premise constituted by one or more relationships that have to hold in order to apply the considered rule. When all relationships in the premise hold, the rule allows the system to infer the validity of the relationship in the conclusion of the rule. The system infers only relationships that definitely

hold, i.e., relationships that hold in any model (i.e., in any set of granularities) satisfying all relationships in the starting set of relationships. We will introduce rules in the inference system and their semantics. Moreover, we will prove that the inference system is sound and complete.

An important improvement of this dissertation with respect to previous proposals about granularities is the application of proposed approaches and notions to a real relational spatio-temporal database, i.e., PCR. For this reason, we design a database for granularities. The database has been designed for containing granularities (i.e., their definitions and granules) and all the related data (e.g., operations used for creating them and relationships between them). This database can be integrated to any existing spatio-temporal database and allows one to enrich and qualify spatio-temporal data with granularities. We will show how this may be done by extending and enriching the PCR database.

Moreover, we will propose a spatio-temporal query language capable to exploit granularities for retrieving spatio-temporal data from a database enriched with granularities. At the best of our knowledge, no spatio-temporal query languages dealing with granularities have been proposed. The query language we propose here allows one to retrieve data by specifying spatio-temporal selection, join, and grouping conditions based on granularities. We introduce also four different semantics for the management of temporal, spatial, and spatio-temporal dimensions. These semantics specify how the system has to manage tuples associated to the given dimensions for the evaluation of the query. In this way, automatic support to the management of temporal, spatial, and spatio-temporal dimensions is added and queries becomes easier to write and understand. We will show that the new constructs proposed in our spatio-temporal query language can be translated into classical SQL [122] statements. The proposed query language will be used to express spatio-temporal queries on the PCR database and to retrieve data from it with respect to spatio-temporal constraints based on granularities. Several examples will be presented both for illustrating the query language capabilities and for showing how clinical spatio-temporal data qualified with granularities can be exploited.

1.3 Overview of the thesis

In order to face the issues we introduced in the previous section, this dissertation has been structured as follows. In Chapter 2 we recall basics notions about temporal (Section 2.1), spatial (Section 2.2), and spatio-temporal databases (Section 2.3). In particular, for each of these contexts, we introduce notions about the representation of context-relevant information and data types in databases and we discuss main proposals in literature about ad-hoc query languages. Moreover, considering the temporal context, we present proposals dealing with the definition and representation of temporal granularities.

In Chapter 3 we introduce a clinical motivating scenario demonstrating the importance of spatio-temporal granularities in epidemiological studies. This scenario will be used through all the dissertation for providing examples of proposed notions. We focus on a psychiatric case register storing information about patients'

contacts with psychiatric services. On this database, spatio-temporal epidemiological studies are being conducted by researcher. For this reason in Section 3.1 we introduce background notions about spatio-temporal epidemiology while in Section 3.2 we present the psychiatric case register and the context in which it has been developed and it is used.

In Chapters 4, 5, 6, and 7 we present the contributions of this thesis. In Chapter 4 we present our framework for spatial granularities after a discussion about main related work. For answering to main requirements in the management of spatio-temporal granularities and data, the framework includes the definition of spatial granularity, of some related notions, and of relationships and operations over spatial granularities. As we mentioned, relationships allow one to reason about, compare and transform data qualified with spatial granularities, while operations allow one to define new granularities from already defined ones. The framework has been defined for two different spatial data models that we introduce in Section 2.2.3: the vector model and the raster model.

Following a similar structure, in Chapter 5 we present our framework for spatio-temporal granularities by merging the framework for temporal granularities proposed by Bettini et al. [23, 27, 154] with our framework for spatial granularities. The framework for spatio-temporal granularities includes the definition of spatio-temporal granularity and of relationships and operations over them. The framework has been studied for both the vector model and the raster model.

The theoretical part of the dissertation continues in Chapter 6 where we show how the evaluation of relationships between granularities, that may be very useful but also computational heavy, can be avoided in some cases by using an inference system for relationships. We present an inference system (including its semantics) that, starting from a given starting set of relationships assumed to hold, infers all other relationships that definitely hold. The inference system is constituted by a set of inference rules. We prove the soundness and completeness of the inference system with respect to the semantics that we present.

Once the frameworks have been defined, in Chapter 7 we deal with their practical usage in spatio-temporal databases. Thus, in first place we deal with the representation and the use of proposed notions of granularities in spatio-temporal databases. For this purpose, in Section 7.2 we conceptually and logically design a database for representing information about temporal, spatial, and spatio-temporal granularities. This database can be used for integrating granularities with an already existing spatio-temporal database; in this way spatio-temporal data in the original database can be qualified and aggregated by using granularities. After that, in Section 7.4 we exemplify this integration with a real spatio-temporal database. We do that by showing how spatio-temporal granularities can be used for enriching the psychiatric case register database we introduced in Chapter 3 and that we design in Section 7.4.1. We show both how the psychiatric database has been extended for including also granularities and how it may be queried exploiting spatio-temporal granularities. To do that, in Section 7.3 we extend the temporal query language proposed by Combi et al. [57] with new constructs for dealing with temporal, spatial, and spatio-temporal granularities.

Note that, each one of the previous three chapters starts with a section where main work related to the issues faced in the chapter itself are discussed.

Finally, in Chapter 8 we summarize the content of this thesis and we briefly highlight future research directions we would like to explore.

Some parts of our previous published or submitted proposals have been used as a basis for some chapters of this dissertation:

[18] Alberto Belussi, Carlo Combi, and Gabriele Pozzani. Towards a formal framework for spatio-temporal granularities. In *Proceedings of the 15th International Symposium on Temporal Representation and Reasoning, TIME 2008*, pages 49–53, Montréal, Canada, June 2008. IEEE Computer Society.
- Section 4.2
- Section 5.2

[19] Alberto Belussi, Carlo Combi, and Gabriele Pozzani. Formal and conceptual modeling of spatio-temporal granularities. In Bipin C. Desai, Domenico Saccà, and Sergio Greco, editors, *Proceedings of the International Database Engineering and Applications Symposium, IDEAS 2009*, pages 275–283, Cetraro, Calabria, Italy, September 2009. ACM.
- Section 4.2
- Section 5.2
- Section 7.2

[169] Gabriele Pozzani and Esteban Zimányi. Defining spatio-temporal granularities for raster data. In *Proceedings of the 27th International Information Systems Conference (BNCOD 2010)*, Dundee, Scotland, June 2010. Springer.
- Section 4.3
- Section 5.3

[62] Carlo Combi and Gabriele Pozzani. An inference system for relationships between spatio-temporal granularities. Technical Report 80/2010, Department of Computer Science, University of Verona, Italy, September 2010. http://www.di.univr.it/report.
- Chapter 6

[63] Carlo Combi and Gabriele Pozzani. An inference system for relationships between spatio-temporal granularities. Submitted to the *12th International Symposium on Spatial and Temporal Databases, SSTD 2011*, Minneapolis, MN, USA.
- Chapter 6

[20] Alberto Belussi, Carlo Combi, Gabriele Pozzani, and Francesco Amaddeo. Dealing with multigranular spatio-temporal databases to manage psychiatric epidemiology data. Submitted to the *Journal of Biomedical Informatics*.
- Section 7.4

2

Background

Many software applications need to manage data and information in order to provide their services. In some cases these data are managed internally by the application itself, while in many other cases applications delegate this task to some other software specifically designed for that. These software tools are called *Database Management Systems* (DBMSs) and have been introduced since 1970 [114]. A DBMS provides applications with the access to data freeing applications from the onerous details in the care and feeding of their data [114]. Main functionalities of DBMSs comprise storage, modification, and extraction of information from a large, shared, and persistent collection of data by warranting reliability and privacy in an efficient way. DBMSs may be used by home applications as well by critical applications (e.g., bank software or WEB hosting providers) and they can manage small data collections with just few entries as well as huge collections with millions of entries and terabytes of data (e.g., biological data including genetic data).

Any data collection managed by a DBMS is called *database*; some other terms (e.g., catalog) are used by different commercial DBMSs. Data are structured and organized in databases with respect to a data model. In literature several data models have been proposed, including, flat model, hierarchical model, network model, relational model, object-relational model, and object-oriented model [76].

In the flat model (also called flat file model) data are not structured and are stored in a single file usually containing a record per line. In each line, data fields are separated by a single delimiting character. Flat databases are used for example in Unix-like operating systems for storing users' information.

In the hierarchical model [206] (first introduced in the IBM's IMS DBMS [144]) data are organized in a tree-like structure in which the parent-child relationship can exist between information. Data are partitioned in entity types that group records with a specific meaning. Each record in an entity can refer to other records, eventually in different entities, but can be referred by only one record. That is, each record has exactly one parent but may have several children. Removing the restriction on the number of parents for each record in the hierarchical model, we obtain the network model [14]. Thus, in this case, the database schema, viewed as a graph, is not restricted to be a tree.

The relational model [50] is currently the most used database model and we will focus on it in the next section. In recent years the object-oriented paradigm

has been developed and applied both in programming languages and databases. In the database case, the aim is to introduce in DBMSs the type system used also in application programs. The relational model has been extended by introducing object data types and their features (e.g., encapsulation and polymorphism). These DBMSs are known as Object-Relational DBMSs (ORDBMSs). ORDBMSs allow users to define their own types and methods that apply to them, while in relational DBMS (RDBMS) data types are restricted to a fixed and limited set. Currently, many important commercial DBMSs support the object-relational model, for example, PostgreSQL [168], Oracle database [159], Microsoft SQL Server [147], IBM's DB2 [120]. Despite this consideration and that in the rest of the thesis we will often refer to these DBMSs, since we will focus only on relational features of these DBMSs, without regard to their object-oriented characteristics, in the next section we will deepen the relational model. An example of pure relational DBMS, i.e., an RDBMS not supporting objects, is MySQL [158]. Besides ORDBMSs, also Object-Oriented DBMSs (OODBMSs) have been developed based on the object model. OODBMSs are completely based on objects and use them as building block; moreover an object-oriented programming language is used as database language. OODBMSs also provide support for the persistence of objects. The object-oriented data model and ad-hoc query languages based on it have been standardized in [38] by the Object Data Management Group (ODMG) [155]. The standard data model specifies how classes, eventually with attributes, methods, and relationships between them, have to be defined. The standard comprises two languages. The Object Definition Language (ODL) provides support for the specification of object schemata, including attributes, relationships, and method signatures. The Object Query Language (OQL) is a declarative query language based on the Select-From-Where syntax also used in SQL.

In last years relational, object-relational, and object-oriented DBMSs have been extended for adding support for temporal and spatial data. In the rest of the chapter we will provide basic notions about these temporal, spatial, and spatio-temporal databases. In particular, in Section 2.1 we introduce specific notions about the management of temporal aspects of data in DBMSs, including their representation and querying. Moreover, we introduce the concept of temporal granularity including its definitions proposed in literature based on different formalisms and the calendar algebra for dealing with time granularities. In Section 2.2 we introduce issues related to the management of spatial data into databases. We will introduce notions about the representation of spatial data and their querying, and we will describe two different models for the representation of spatial data: the vector model and the raster model. Finally, following a similar structure, in Section 2.3 we will discuss the representation and management of spatio-temporal information in databases.

2.1 Temporal databases

Temporal databases (also known as historical databases) are those databases that take in account and supports temporal aspects of data. The goal of temporal databases is to manage time-varying data in a database context. In particular, one

or more temporal dimensions (i.e., temporal data with different meanings) can be associated to data or tuples in a database, enabling in this way new analysis, management, and querying features. For example, a valid time period can be associated to the residence address of a customer in order to trace when customer move to a different address and to retrieve the right address with respect to the time period we are interested to. Thus, the first feature that can be exploited adding a temporal attribute to data is the representation of both their current and past states.

As said, temporal databases extend traditional databases adding support for time. This extension, i.e., the management of temporal aspects, is a cross-referring topic and affects a variety of research areas in the database context. Temporal database research faces issues related to the representation of time and temporal dimensions, conceptual, logical, and physical design of temporal databases, semantics aspects of time, and management and querying of time evolving data. Concepts and issues faced in temporal database researches have several applications also in other information system technologies and research fields, e.g., temporal workflow and business management systems [103], temporal data mining [3], and temporal data warehouse [140]. Here only aspects relevant for this thesis will be discussed.

2.1.1 Representation of time and temporal dimensions

In the database context time is usually represented as a totally ordered set of time points in a temporal domain. This can be depicted as the usual directed time axis.

Based on this representation of time, in literature two main models for representing temporal dimensions and events have been proposed: the instant-based model and the interval-based model. In the first one tuples and data are associated with a single time point indicating usually that the considered data are valid in that particular time instant. For example, a bank transfer is associated with a timestamp representing the time when the transaction has been performed. On the other hand, in the latter model data are temporally qualified associating them with a time interval (or time period). Time intervals are usually represented as a pair `(start,end)` describing the fact that the interval starts at `start` and finishes at `end`. For example, notices on a University WEB site may be associated with a time period representing the fact that they are valid and published on the WEB site only during the considered period.

Several semantics can be associated to time instants or periods that qualify database objects. Usually the most important temporal dimensions represented in temporal databases are *valid time* and *transaction time* [124]. The valid time associated to an information represents the time instants or intervals when the information is true in the modeled reality. Conversely, transaction times represent the time (usually an interval) during which objects are current and can be retrieved in the database. Thus, usually, a transaction time associated with an object starts when that object is stored in the database and finishes when it is logically deleted. Due to this consideration, transaction times are generated (e.g., by using transaction commit times) and supplied by the system. Depending on the application, in some cases only one of these two temporal dimensions can be used while in some other cases both can be represented. A database in which both dimensions are represented is called a *bitemporal database*.

In literature other temporal dimensions have been proposed. In [45], Kim and Chakravarthy introduced *event time* in order to represent the delay between the time when a fact become valid and the time when a decision has been taken or an event happened determining the considered fact. When event time and valid time coincide event time can be omitted and represented by using the valid time. Considering the relationship between starting event time and the starting valid time, Kim and Chakravarthy classified events in the following way:

- on-time events: the start of event and valid times are the same;
- retroactive events: valid time starts before the event time;
- proactive events: valid time starts after the event time.

Combi and Montanari [56] observed that in some cases one has to model also the time when the information system or user become aware of a fact, and then they can take a decision or perform an action. Thus, authors introduce the *availability time*.

Since all these temporal dimensions have a well known semantics and meaning, a system that allows a user to represent one or more of the previous temporal dimensions knows exactly how to manage them and can thus supply ad-hoc support for them.

Conversely, users can represent also temporal dimensions with semantics different from the previous ones. All these "generic" temporal dimensions are called *user-defined time* and have no special support from the system.

All previous temporal dimensions can be added to a relation schema. A relation that includes one system supported valid time is called *valid time relation*. On the other hand, a relation with one system supported transaction time is called *transaction time relation*. In both cases, as just mentioned before, temporal dimensions can be incorporated in relations by adding a time point attribute (for representing point events) or a time interval (for representing interval events). Relations incorporating both valid and transaction time are called *bitemporal relations*.

All the most important and used commercial database management systems (DBMSs) (e.g., PostgreSQL [168], MySQL [158], Oracle [159], SQL Server [147], IBM DB2 [120]) allow users to add temporal attributes in order to represent user-defined temporal dimensions. They define temporal data types for representing dates, times, timestamps (time and date), with or without time zone, and durations (unfortunately called intervals in DBMSs), i.e., an amount of time with known length but with no specific starting and ending instants. All the mentioned DBMSs provide functions and operators to manage, convert, and compare temporal data types.

Moreover, some of previous DBMSs include also, in some sense, support for transaction time (called with different names) in order to provide versioning and roll-back of data changes. For example, Oracle Database includes support for transaction time in order to access prior states of a database and to roll back to a previous state discarding all changes made after that time. Further, Oracle Workspace Manager introduced support for valid-time and bitemporal relations, extending traditional databases to historical databases [161].

2.1.2 Temporal query languages

As said before, temporal dimensions can be incorporated in relations by adding temporal attributes. These attributes can be then queried or used to manage and retrieve other stored data. Previously mentioned commercial DBMSs use SQL [122] as query language.

SQL (Structured Query Language) is the ISO standard language for relational databases. SQL provides commands for schema creation and modification, data access control, data insertion, querying, updating and deletion. In particular, queries are performed with the well-known `SELECT` statement. A `SELECT` statement retrieves data from one or more tables that comply constraints described by the user in the command itself. A `SELECT` query is divided in different components called *clauses*. The basic clauses are:

- `SELECT`: it specifies the list of columns and expressions to be included in the final result;
- `FROM`: it specifies the list of tables from which data is to be retrieved;
- `WHERE`: it specifies the condition that retrieved data must comply with;
- `GROUP BY`: it is used for grouping in just one row all rows with common values on given columns and, usually, for performing aggregate functions on their values. Groups to be included in the result can be filtered by using the `HAVING` clause, that allows the user to specify conditions that groups must satisfy;
- `ORDER BY`: it specifies columns to be used for sorting retrieved data.

SQL provides also predicates, comparison operators, and functions for managing data and that can be used also in `SELECT` queries for specifying the retrieving condition. However, SQL has been studied to work on traditional "atemporal" databases and it does not provide support to write temporal queries and manage historical or versioned data. In particular, SQL treats temporal attributes like the other ones just providing comparison operators and modifying functions.

To get over these limitations of SQL with respect to temporal databases, in literature several temporal query languages have been proposed [32, 68, 152, 174, 180,194,201]. In general, these languages take account of the semantics of proposed temporal dimensions (e.g., valid time and transaction time).

In 1995, Snodgrass et al. proposed TSQL2 [194], a temporal extension to the SQL-92 standard query language. Despite many researches proved the usefulness, effectiveness and efficiency of TSQL2, the project for incorporating some TSQL2 capabilities in the ISO SQL standard has been canceled in 2001. However, some of its features have been implemented in Oracle DBMSs [159].

The draft proposed for adding temporal support in SQL standard, called SQL/Temporal, includes both valid (i.e., the time instants or intervals when an information is true in the modeled reality) and transaction times (i.e., the time interval during which data are current and can be retrieved in the database) [124] from TSQL2 and two semantics for temporal queries: sequenced and non-sequenced semantics. These semantics allow a user to specify how the DBMS has to (automatically) manage temporal information (e.g., the valid time) during the evaluation of a query. According to the first semantics, a SQL query is evaluated on a given temporal dimension instant by instant, that is the query engine evaluates the query for each time instant in the time domain selecting only those tuples whose value for

the considered temporal dimension includes or is equal to the considered instant. Conversely, in the latter semantics, a temporal relation is considered as a whole, the query is evaluated only one time considering all tuples in the relations, without regard to their value for the given temporal dimension. In this case, the user has, eventually, to manage the temporal dimension. These semantics have been used also in [32] to define ATSQL, an SQL-based temporal language, where semantics (called modifiers) can be used in order to specify how queries have to be evaluated.

In [57] Combi et al. proposed the temporal query language T4SQL. It extends the SQL language [122] adding the support to temporal attributes and queries. It is able to manage valid time, transaction time, availability time (i.e., the time when the database system or user become aware of a fact), and event time (i.e., the time when a decision has been taken or an event happened determining the considered fact) [124]. Moreover, T4SQL allows a user to specify in a query a semantics for each temporal dimension considered in the query. The four proposed semantics extend the two semantics proposed in the SQL/Temporal draft:

- SEQUENCED(d): it forces an instant by instant evaluation of the query with respect to the temporal dimension d (e.g., valid time). For each time point in the domain of d, the DBMS evaluates the query by selecting only those tuples where d contains the considered time point. This semantics allows one to perform historical analyses. This semantics corresponds to the homonymous semantics in SQL/Temporal.
- CURRENT(d): specifying this semantics the DBMS evaluates the query only on those tuples where d is equal or contains the current date.
- NEXT(d): this semantics considers only pairs of tuples related to the same entity and that are consecutive with respect to the temporal ordering.
- ATEMPORAL(d): it disables any support from the system, thus d is considered as a classic attribute managed by the user. This semantics corresponds to the non-sequenced semantics in SQL/Temporal.

For example, the following query retrieves, for each hospital patient, the symptoms whose valid time overlaps the valid time of the prescribed therapy according to the current state of the database.

```
SEMANTICS SEQUENCED ON VALID, CURRENT ON TRANSACTION
SELECT Symptom, PatId
FROM PatSymptom AS ps, PatTherapy AS pt
WHERE ps.PatId = pt.PatId
```

Both previous languages include support for temporal joins and temporal grouping. The first one is an extension of natural join in which join condition include one or more temporal dimensions. In particular, tuples are merged if their values for a given temporal dimension overlap. The value of the temporal dimension in the resulting tuples is the intersection of the input values. In temporal grouping, tuples can be grouped with respect to one or more temporal dimension. On grouped tuples (temporal) aggregate functions can be applied.

2.1.3 Temporal granularities

A particular class of temporal data used for temporally qualifying other database information comprise temporal granularities. The notion of temporal granular-

ity has been developed since the last years of 1990's. Based on several previous proposals, Bettini et al. developed in [23, 27, 215] the formalization for temporal granularity now widely accepted by the temporal research community.

Informally, a temporal granularity represents a partition of a time domain. Each element of this partition (i.e., the granularity) is called *granule*. When, describing a fact, we can associate these granules to data in order to provide them with a temporal qualification at the suitable granularity. In other words, a temporal granularity represents a temporal unit of measure.

To give the definition of temporal granularity it is first of all necessary to define how we represent a time domain. A *time domain* is a pair $(T, \leq)$ where T is a non-empty set of time instants and $\leq$ is a total order over T. The domain represents the usual time line: it is the set of primitive entities used to interpret the other notions. A time domain is *bounded* if it has upper and lower bounds with respect to its order relationship, otherwise is called *unbounded*. Formally, a time domain T is bounded if exist instants $t_1, t_2 \in T$ such that $t_1 \leq t \leq t_2$ for all instants $t \in T$. Moreover, a time domain can be either *dense* or *discrete*. Their definitions are based on the mathematical definition of dense and discrete sets [113]. A dense domain allows one to represent arbitrarily finer granularities while discrete domains are usually used when the system has to represent a fixed smallest granularity (e.g., `seconds`). Examples of discrete time domain are $(\mathbb{N}, \leq)$ and $(\mathbb{Z}, \leq)$, while $(\mathbb{R}, \leq)$ is an example of dense time domain.

Given a time domain T, a *granularity* is a mapping G from an index set I to the power set of T such that:

1. if $i < j$ and $G(i)$ and $G(j)$ are non-empty, then each element of $G(i)$ is less than all elements of $G(j)$;
2. if $i < k < j$ and $G(i)$ and $G(j)$ are non-empty, then $G(k)$ is non-empty.

The first condition states that granules (i.e., the sets of instants corresponding to indexes) do not overlap one each other and that the index order and the time domain order are the same. Instead, the second condition states that non-empty granules are contiguous. The index set is a subset of integers; thus, a granularity defines a countable set of granules, each one identified by its index. A granularity is *bounded* if there exist two indexes $k_1, k_2 \in I$ such that $G(i) = \emptyset$ for all $i < k_1$ and $k_2 < i$. Often, granules are not referred by their indexes but using labels, i.e., textual representations. For this purpose it can be defined also a label mapping that associates to each label the corresponding granule.

Usual granularities are `seconds`, `minutes`, `days`, `years`. Hence, if we consider, for example, the granularity `years`, the granule `years`(2008) corresponds to the time instants belonging to year 2008, while with the label "November 2008" we refer to instants in the 11th month of year 2008. Labels like this have been introduced [154] for facilitating users. As a matter of fact, human users are used to relative representations of time instead of representations based on indices (e.g., integers), and labels allow one this kind of representation.

Thus, a granularity is made up of some non-empty granules. Granules represent sets of time instants perceived and used as indivisible entities. A granule can represent either a single instant, a time interval (i.e., a set of contiguous instants), or a set of non-contiguous instants.

Other notions are defined about a temporal granularity:

- the *origin* of granularity G is a special granule designated as the initial granule, e.g., $G(0)$;
- the *image* of a granularity is the union of granules in the granularity;
- the *extent* of a granularity is the smallest interval of the time domain that contains the image of the granularity. Formally, it is the set $\{t \in T | \exists a, b \in Im, a \leq t \leq b\}$ where T is the time domain and Im is the image of the granularity.

Several relationships have been defined between temporal granularities (with the same time domain). These relations allow us to build hierarchies of granularities and address some issues related to the conversion of information from a granularity to a related one. This ability is an important research theme about temporal information systems and temporal reasoning [27, 154].

The most important and used relations among time granularities are:

GroupsInto: a granularity G groups into a granularity H, denoted $G \trianglelefteq H$, if for each index j there exists a (possibly infinite) subset S of the integers such that $H(j) = \bigcup_{i \in S} G(i)$.

FinerThan: a granularity G is finer than a granularity H, denoted $G \preceq H$, if for each index i, there exists an index j such that $G(i) \subseteq H(j)$. If $G \preceq H$ then H is coarser than G ($H \succeq G$).

Sub-granularity: a granularity G is a sub-granularity of H, denoted $G \sqsubseteq H$, if for each index i, there exists an index j such that $G(i) = H(j)$.

ShiftEquivalent: two granularities G and H are shift equivalent, denoted $G \underline{\leftrightarrow} H$, if there exists an integer k such that $G(i) = H(i + k)$ for all i in the index set. Note that $G \underline{\leftrightarrow} H$ if and only if $G \sqsubseteq H$ and $H \sqsubseteq G$.

Partitions: a granularity G partitions a granularity H if $G \trianglelefteq H$ and $G \preceq H$.

GroupsPeriodicallyInto: a granularity G groups periodically into a granularity H if:

1. $G \trianglelefteq H$;
2. there exist $n, m \in Z^+$, where n is less than the number of non-empty granules of H, such that for all $i \in Z$, if $H(i) = \bigcup_{r=0}^{k} G(j_r)$ and $H(i + n) \neq \emptyset$ then $H(i + n) = \bigcup_{r=0}^{k} G(j_r + m)$.

Using these relationships it is possible to define two other useful notions: bottom granularity and calendar. Given a granularity relation *g-rel* and a set of granularities over the same domain, a granularity G in the set is said to be a *bottom granularity* with respect to *g-rel* if for each granularity H in the set, we have G *g-rel* H. Moreover, we call *calendar* a set of granularities having the same domain and including a bottom granularity with respect to $\trianglelefteq$ (groups into). An usual example of calendar is constituted by the set {`Minutes`, `Hours`, `Days`, `Months`, `Years`}.

Using these definitions and notations, Ning et al. [154] completed the framework for temporal granularity defining some operations useful to build new granularities from already existing ones. In particular they use an algebraic approach called *calendar algebra*.

The calendar algebra consists of operations allowing one to manage and build temporal granularities. These operations can be classified in two classes: grouping-oriented and granule-oriented operations. The first ones combine the granules of a given granularity to form the granules of a new granularity, while the last ones construct a new granularity choosing some granules from a given one.

In literature, several alternative approaches to the algebraic one have been proposed. They use string-based [220], automata-based [65, 66] and logical formalisms [54, 87, 88, 148].

The logical-based framework overcomes the limited reasoning methods and expressiveness of the algebraic one. In fact this approach provides an extensively investigated reasoning method also based on some automatic tools as theorem provers and model checkers. Hence, this approach is most used in the context of verification where decision procedures are unavoidable to validate the granularity system. In [149, 150] several results about decidability of temporal structures are studied.

In the logical framework, temporal granularities, and their interconnections, are represented by using mathematical structures called *layered structures*. A layered structure is an eventually infinite set of related differently-grained temporal domains. It represents the relevant time domains and the relations (similar to the ones defined in the algebraic approach) between time points belonging to different domains, constructing a hierarchy. Several operations allow one to move horizontally within a given temporal domain and vertically across different domains.

Over a layered structure, logical formulas (both classical and temporal) can be formulated specifying suitable properties. A formula may involve different granularities mixing several operators. Thus, some algorithms are provided to solve the satisfiability problem (i.e., to verify whether a given formula is consistent) and the model checking problem (i.e., to check if a formula is satisfied by a given model).

Another proposal using a logical-based framework was presented by Combi et al. [54]. They represent a granularity G by using a *linear time structure* labeled with proposition symbols taken from $\{P_G, Q_G, H_{P_G}, H_{Q_G}\}$. These symbols are used to delimit start and end of granules and start and end of gaps inside granules, respectively. A linear time structure is $(\mathbb{N}, <, V)$ where $(\mathbb{N}, <)$ is the time domain and V associates to each time instant a set of proposition symbols. For example, the structure such that $V(i) = \{P_G\}$ if and only if i is even and $V(i) = \{Q_G\}$ if and only if i is odd represents the granularity whose ith granule is composed by time instants $2i$ and $2i + 1$. Combi et al. study also relationships between granularities and expressiveness.

The approach based on strings has been developed by Wijsen [220]. He restricts the attention to infinite discrete periodic granularities, i.e., granularities whose domain is isomorphic to $\mathbb{Z}$ and granules are repeated periodically. Using this approach granularities are expressed in terms of periodic strings over an alphabet of three symbols, namely, ■ (filler), □ (gap), and ≀ (separator), which are respectively used to denote time points covered by some granule, to denote time points not covered by any granule, and to delimit granules. This representation is called *granspec*. Granules are constructed from fillers and gaps, and are delimited by separators. Then, Wijsen represents a granularity as an ordered pair where the

first element is the offset (i.e., the initial non-periodic part of the granularity) and the second one is the repeating pattern (composed by one or more granules). The main disadvantage of this formalism is that it is not compact: in fact, the representation of a granularity can be very long if the granularity has a long prefix or period (e.g., the Gregorian Calendar). This approach was further developed by Dal Lago et al. [65]. They define an automata-theoretic counterpart of the string-based model defined by Wijsen. They propose Single-String Automata (SSA), a variant of deterministic Büchi automata [202] accepting a single infinite string to represent temporal granularities. They show that SSA provides an efficient solution to the fundamental problems of equivalence and classification about granularities. Finally, they show how expressions of calendar algebra can be mapped into SSA.

In [66] Dal Lago et al. face two kinds of optimization problems about automata-based representations of time granularities, namely, computing the smallest representation and the most tractable representation (i.e., the one in which granule conversion is more efficient).

Using these different formal specifications for temporal granularity several issues are faced in literature:

- with regard to temporal databases, it has been studied how to represent these formalizations in databases and to associate data with temporal information expressed using granularities. An important question is about how to manage, view, and query data dealing with multiple granularities providing a precise semantics and guaranteeing consistency [215, 216];
- with regard to data mining, it has been studied how to manage a huge amount of temporal data, recording and querying it. In particular, it is crucial to develop algorithms to derive implicit information and to predict the future trend and behavior of data we are monitoring. The latter activity requires an analysis of the frequency of certain events, the discovery of their regularity, and the identification of sets of events that are linked by particular temporal relationships. We note that also these algorithms must be able to deal with multiple granularities [3, 26, 64, 141, 175, 218, 219, 221];
- with regard to problem solving, several real-world problems (e.g., scheduling, planning and diagnosis) can be formulated as temporal constraint satisfaction problems, eventually involving multiple granularities. In this kind of problems, variables are used to represent event occurrences and constraints are used to represent their granular temporal relationships [24, 25, 153, 166, 185].

From the above research issues, we observe that an important topic regards the management of multiple granularities. In fact, anyone in everyday life implicitly uses several different temporal granularities (and, implicitly, converts information between different granularities); hence, any application handling temporal data must be able to treat multigranularities. In particular a problem to consider is the granule conversion. This operation allows the user to view temporal information in terms of different granularities and to integrate temporal data expressed with different granularities (e.g., data coming from different sources). We will see in the next sections that this is a common problem also in the spatial and spatio-temporal context.

2.2 Spatial databases

Temporal databases provide users with the support for managing temporal and temporally qualified data. Similarly, spatial databases (also called geometric or geographical databases) are those databases in which geographical data can be represented and managed. Spatial databases are database system extended with additional functions for handling spatial data [187]. Main examples of spatial data comprise locations, roads, and regions on the earth surface. Note that, in spatial databases, geometrical information is connected to non-spatial data. Thus, similarly to temporal dimensions, spatial data are used for qualifying and enriching classical data.

As it happens for temporal databases, the representation of spatial data types and data adds new features to databases and introduces new issues about database-related topics, e.g., conceptual, logical, and physical modeling of geographical databases, querying of spatial data, management of uncertain spatial information, indexing of spatial data. Here, we discuss only few of these topics.

An important distinction is between spatial DBMSs and Geographical Information Systems (GISs). Spatial DBMSs aim, as any other database system, at a permanent and persistent representation of spatial data. Moreover, they provide functions to define, manipulate, and retrieve spatial data. On the other hand GISs are software systems that aim at the management, transformation, and analysis of spatial data stored in several ways, including spatial databases and files. Thus, a GIS software system does not provide directly structures and functions to store geographical data, and can be one of the possible applications based on spatial databases, where data are really stored. In this thesis, we focus on spatial databases, but many concepts, notions, and features can be considered and extended also to GIS software.

2.2.1 Representation of spatial data

The first feature of spatial databases is the definition and representation of spatial data types. Spatial data are usually represented as geometries (also called *features*) that should be interpreted on an implicitly or explicitly represented space domain (e.g., an Euclidean space).

Spatial data may be used either to represent the spatial position of any object in a space domain, or to represent the space itself, i.e., "properties" of points in the space. Both these representation requirements can be satisfied allowing one to define single geometrical objects and collections of objects [105].

This idea has been implemented in the specifications for spatial data proposed by the Open Geospatial Consortium (OGC) [156]. OGC is a non-profit international voluntary organization aiming at the development of standards for geospatial and location based services. One of the proposed standards includes the definition and specification of spatial data types.

Based on the previous idea and on the OGC standard specification, spatial data types allow one to represent several kinds of spatial data that can be classified in the following way:

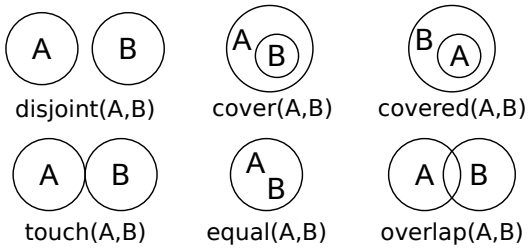


Fig. 2.1: Examples of the six relationships defined in the Four-Intersection Model

- simple: 0-dimensional data (points), 1-dimensional data (curves or lines), 2-dimensional data (polygons or regions)
- collections: sets of simple spatial data, i.e., sets of points, curves, or polygons.

Simple features may be used for representing single objects. Points represent the geometrical position in the space of objects but not their extent. For example, the location on a map of domiciles of hospital patients can be modeled as points. Lines can represent generic curves in space but are usually modeled as polylines, i.e., sequences of line segments. Roads, railways, and rivers are typical examples of objects that can be represented by using lines. Regions represent the extent, possibly with holes and disconnected pieces, of objects in space. For example, a country, the extent of a building, a natural park may be modeled with regions. On the other hand, collections allow one to represent sets of any kind of simple spatial objects. For example, a collection of regions can be used to represent all countries in Europe, or all buildings in a University campus. All these notions can be defined on any multi-dimensional space and have been deepen for two- and three-dimensional (where regions become volumes) spaces. Here, we focus on bidimensional spaces.

Networks are a special case of collections of lines. A network can be viewed as a graph in which edges are represented by a set of line geometries connecting nodes represented as point objects. Railways, highways, urban transports, telephone systems, power supply lines are examples of networks. Partitions are special cases of collection of regions. A partition is a set of regions required to be disjoint (possibly, regions can share their boundary points). Partitions can be used to represent several kinds of maps: for example, land use maps, municipalities, soil type maps. Moreover, we will see in Chapter 4 that partitions are the basis for the definition of spatial granularities.

Points, lines, and regions are the basis for many spatial algebras proposed in literature [104, 109, 139, 182, 228] in which also operations and relationships over spatial objects have been investigated.

Proposed relationships include three main classes of relationships: topological relationships, directional relationships, and metric relationships.

Topological relationships are the most deeply investigated ones and have been proposed in several frameworks. In order to define topological relationships, the set of points belonging to any spatial object can be partitioned in two subsets [183]: boundary points and interior points. A point belongs to the boundary of the object if all its neighborhoods contain both interior and exterior points. Interior points are

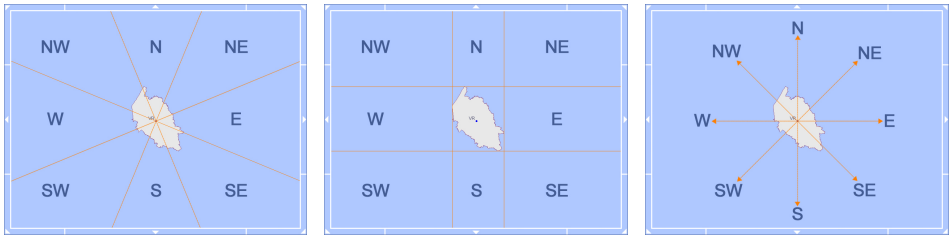
those points that have at least one neighborhood containing only points belonging to the object itself. Exterior points are those points that are neither interior nor in the boundary, i.e., all points that have at least one neighborhood containing only points that are not in the considered object. For each pair of spatial objects, topological relationships consider their sets of interior and boundary points and how they are related, e.g., if they overlap each other or they are disjoint. This leads to define 16 combinations [72] but only six of them are valid relationships (see Fig. 2.1): `disjoint`, `covered`, `touch`, `equal`, `cover`, and `overlap`. This model is called also Four-Intersection Model because between two spatial regions, there are four intersections to consider. In [72] Egenhofer studied these relationships only between regions without holes and disconnected pieces. Egenhofer extended this work [71] also to consider points and lines. In [73] the author has further extended his first work about regions in order to consider also intersections with the exterior points sets. This model is also called Nine-Intersection Model. This last model has been extended by Clementini et al. [49], defining the Dimensionally Extended Nine-Intersection Model with 256 possible relationships. In this model also the dimension of the intersections (i.e., if intersections are empty, 0-, 1-, or 2-dimensional) is taken into account. Clementini et al.'s work has been integrated in the standard specification proposed by the OGC. However, they proved that the five relationships `touch`, `covered`, `cross`, `overlap`, and `disjoint`, together with three function to access boundary points of objects, are enough to represent all the other relationships.

A similar set of topological relationships is defined in the Region Connection Calculus (RCC) [51, 171]. The RCC framework describes possible qualitative relationships between two spatial objects considering how their are connected. In the basic formulation, the framework includes eight relationships, as, for example, externally connected (objects share only a part of their boundaries), equal, partially overlapping (objects share a part of their boundary and interior points), and tangential proper part (one object is contained in the other and the intersection of their boundaries are not empty). The RCC framework has been further extended to consider regions with a single hole in [208].

Distance relationships [233] allow one to represent constraints and relationships about the distance between spatial objects, e.g., distance between objects is greater than 1 km.

Direction relationships [28, 48, 90, 92, 98, 137, 191, 232] describe where an object is placed with respect to another one. For example, one can state that Italy is south of Germany. The main idea for defining this kind of relationships is to partition the space around a reference object and relate other objects with this one observing in which part of the space they are. There exist many ways to partition a space. Based on a psychological investigation by Franklin et al. about how people perceive cardinal directions in space, Goyal [98] proposed three methods for partitioning space:

- cone-based model: a particular point of the reference object is taken (e.g., the geometric centroid) and the space is partitioned in four or eight cones around this point (see Fig. 2.2a). If four cones are used, the `North`, `East`, `South`, and `West` directions are defined. If eight cones are used, also `NorthEast`, `NorthWest`, `SouthEast`, and `SouthWest` directions are considered.



(a) Cone-based example (b) Rectangle-based example (c) Projection-based example

Fig. 2.2: Goyal's three ways for partitioning space around a reference object.

- Rectangle-based model: the bounding box of the reference object is taken, its four sides are used to partition the space into nine parts (see Fig. 2.2b): the bounding box itself, the strips of space `West`, `East`, `North`, and `South` of the bounding box and the four quadrants `NorthWest`, `NorthEast`, `SouthWest`, and `SouthEast` of the bounding box. Note that this method takes into account the extent of the reference object.
- Projection-based model: similarly to the cone based model, a particular point of the reference object is taken and from it a directed semi-axis starts in the direction of each cardinal point we are interested in to represent (see Fig. 2.2c). The relationship with a second spatial object is computed projecting it on the axis.

Once the space has been partitioned with respect to a reference object, the relationship of other objects with it is computed observing in which part of the partitioned space they are. In the case of points no problems exist, since they are for sure just in one part of the space. In case we have to relate lines or regions with the reference object, some issues arise. For example, a region can overlap more than one cone in the cone-based model: thus, we have to decide in which direction this region is with respect to the reference object. Several proposals investigated the problem of calculating direction relationships in the case of lines and regions [28, 98, 232]. Main proposed solutions consider a second spatial object be in a direction if

1. one its special point, e.g., the geometric centroid, or
2. at least one its point, or
3. all its points

are in the part of the space around the reference object representing that direction.

Of course, each of these solutions has different properties. In the first case, we do not consider the extent of the object. This leads, in some cases, to strange situations. If we choose the second option, an object may be in more directions since some its points are in one direction while other ones are in another direction. On the other hand, in the third option, there are cases in which an object is not related to the reference one (e.g., in case B overlaps two cones around A, but all

its points are required in a cone for considering it in that direction, B cannot be related to A).

The definition of direction relationships to be used depends on the application we are considering and on the properties we require or wish.

Frameworks for spatial objects comprise also operations on spatial data [104, 109,139]. Proposed operations include functions for calculating the length of lines, perimeter and area of regions, min or max distance between two objects, the bounding box or the convex hull, and centroid. In [109], Güting and Schneider proposed three kind of operations:

- operations returning simple spatial objects; for example, intersection between two lines or two regions, or the contour of a region.
- Operations returning a numerical value, e.g., the length of a line, the perimeter of a region, or the distance between two objects.
- Operators over sets of objects, for example the geometric union of a set of objects.

All these operations have been implemented in the proposed spatial query languages.

2.2.2 Spatial query languages

Several query languages, mainly based on SQL [122], have been proposed [74, 93, 134, 156]. Currently, many commercial DBMSs that implement spatial functionalities, offer spatial data types, relationships, and operations following the approach proposed by OGC in [156] and standardized in ISO/IEC 13249-3 SQL/MM Part 3 [123]. ISO/IEC 13249 SQL/MM is a standard extending SQL with multimedia and application-specific features. In particular, its third part defines how to store, retrieve, and process spatial data in SQL. Spatial data types and functions for converting, comparing, and analyzing spatial data are defined. Some examples of DBMSs implementing this standard include PostgreSQL/PostGIS [172], Oracle Database [160], IBM's DB2 [120], MySQL [158], Microsoft SQL Server [147].

Spatial query languages implement functions for testing the validity of relationships between spatial data, and computing operations over spatial objects; by using these functions they allow one also to perform spatial selection and spatial join. The first one is actually a traditional selection operation based on a spatial predicate (i.e., the testing of spatial relationships or properties over spatial objects). For example, using the SQL/MM standard, the following query retrieves bank branches whose area has an extent of at most 10 km^2. The selection is based on the `ST_Area` function that computes the area of the geometry on which it is applied, i.e., the spatial attribute `area`.

```
SELECT b1.branch_id
FROM branches AS b1
WHERE b1.area.ST_Area('KILOMETRES')<=10
```

The latter is a join operation which join condition is based on a spatial predicate. For example, using the SQL/MM standard, the following query retrieves the identifiers for each two bank branches that have an non-empty overlap in the zones

and also the overlapping area, encoded in a text representation. Note that the join condition contains a spatial predicate `ST_Overlaps()` that tests whether the geometry on which it is applied (i.e., `b1.area`) overlaps or not with the geometry passed as parameter (i.e., `b2.area`). Moreover, the query returns, besides the `id`s of the retrieved branches, the geometry representing the intersection of the areas of the two joined branches. The intersection is computed by using the `ST_Intersection` function and converted in a readable textual form by using `ST_AsText()`.

```
SELECT b1.branch_id, b2.branch_id,
       b1.area.ST_Intersection(b2.area).ST_AsText()
FROM branches AS b1 JOIN branches AS b2
     ON (b1.area.ST_Overlaps(b2.area))
WHERE b1.branch_id < b2.branch_id
```

Since spatial functions may be very heavy from a computational point of view, spatial selection and spatial join are supported in all commercial spatial DBMSs by spatial indexing techniques [126] and special join algorithms [224].

2.2.3 Vector vs raster model

In the previous section we assumed the use of a vector model for representing features. The vector approach uses geometries such as points, lines or polygons to represent objects [94]. For example, a hotel can be represented as a point, a road can be represented as a polyline, and a province as one or more polygons. Vector features can be made to respect spatial integrity through the application of topology rules such as polygons not being allowed to overlap. Vector data can also be used to represent continuously varying phenomena. Moreover, the scale does not affect the representation and the storage requirement of vector data.

However, spatial data can be represented also in an alternative way by using raster maps. A raster map consists of a grid (or matrix) of cells, each one storing a value. Each cell represents an area whose size changes depending from the resolution of the map. Greater is the resolution, smaller is the area corresponding to any single cell. Cells are arranged in rows and columns where rows represent the x-axis of a Cartesian plane and columns the y-axis [138]. Stored values can represent magnitude, distance, or relationship of the cell on a continuous surface. Values can also represent categorical data such as soil type or land-use class.

In the *raster* model, the construction of a raster map starts from the partitioning of the space domain we are interested to survey in many areas (usually they have all the same extent and shape, e.g., a square) that completely cover the domain (see Fig. 2.3). The size of areas depends on the resolution (i.e., the accuracy needed) of the map: it may range from centimeters to kilometers. Inside each area, the physical dimension of interest is measured and its value is associated to the area. The partitioning of the space domain allows one to obtain a discrete space from a continuous one. In fact, areas can be uniquely and totally numbered starting from a specific point, the origin of the raster map. Usually, areas are numbered defining a Cartesian coordinate system: in this way, each area corresponds to a unique pair of integers. Thus, a raster map can be represented by a matrix whose components are called *cells*. Knowing the position in the space domain of

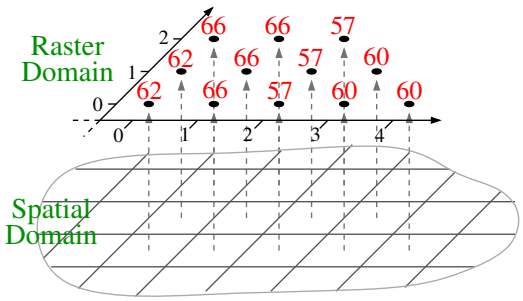


Fig. 2.3: Structure of a raster map

the origin of the map and its resolution (i.e., the size of areas), it is possible to know what area corresponds to each cell and vice versa.

Raster data are generally divided into two categories: thematic data and image data. Values in thematic raster data represent some measured quantity or classication of particular phenomena such as elevation, pollution concentration or population. For example, in a land cover map the value 5 may represent forest, and the value 7 may represent water. The values of cells in an image represent reflected or emitted light or energy such as that of a satellite image or a scanned photograph.

Values stored in raster map can be integer or floating-point. Usually integer values are used for representing categorical data while floating-point are used for representing continuous properties, e.g., elevation or slope. These values, if interpreted properly, allow one to display raster maps as colored or gray-scale images.

Raster maps can represent features. A point is represented by the smallest unit of a raster map, a cell. Since usually points are those objects that have no extent (i.e., they are 0-dimensional) while cells have one, the smaller is the area corresponding to cell, the better is the representation of point features. A line is represented as a series of connected cells. Again, the smaller is the area corresponding to cell, the better is the representation of lines features. Polygons are represented by sets of connected cells that best portrays its shape. However, boundaries of a polygon are now represented by a series of cells, leading to the well-known problem called "jaggies", an effect that resembles stair steps [138].

A given real-world situation can be represented by using either a raster or a vector model. The choice between raster and vector data depends on the application field. In general, raster data are more suitable for environmental applications, involving continuous spaces, while vector data are more suited to human activity [138]. Due to the nature of the data storage technique, raster data allow one easier and more efficient implementations of some spatial analysis techniques, e.g., quantitative and overlay analysis. Moreover, remote sensors (e.g., satellites and cameras) produce a raster-based output. We can also consider that the raster model is more suitable for representing spatial information with limited attributes (e.g., land-use patterns), while the vector data model is better for spaces with complex attributes (e.g., census data). Due to these considerations, raster maps play an important role in some areas of health geography, in which raster thematic

maps can represent demographic, economic, social, and environmental dimensions, and satellite imagery in which surveys are always stored in a raster format. Considering data storage, vector models store only geometries while raster models store also pixels representing empty background. Thus, usually, vector data require less storage space than raster data also considering the usage of compression techniques for raster images (e.g., Run-Length Encoding [96]). Note that it is always possible to convert raster data into a vector model (vectorization) and vice versa (rasterization). However, techniques and algorithms for performing these two conversions are not important for this dissertation and thus they are omitted.

2.3 Spatio-temporal databases

Spatio-temporal databases provide notions, data structures and support for representing moving objects, i.e., spatial data changing over the time. Moving objects can be abstracted in two subtypes: moving points, for which only the position is time dependent, and moving regions, for which also extent and shape change over the time [108]. Moving object databases aim for two different goals: the representation and query of a set of currently moving objects, and the representation of the histories of movements (trajectories). In the first case the name *tracking database* is also used, while the second case is also called *trajectory database* [107].

The usual approach to spatio-temporal databases is to take spatial definitions and structures and to extend them adding temporal components. Thus, notions about temporal and spatial databases exposed in previous sections constitute a basis for spatio-temporal databases. Moreover, due this approach to spatio-temporal notions, spatio-temporal databases offer to users also support to spatial and temporal data.

Research on spatio-temporal databases began in the late 1990s with two different projects. The first one [189, 222, 223] developed a model for keeping trace in a database of a set of time dependent locations. This model is based on the observation that is not necessary to directly store the position, as it requires many updates, but it is enough to store a motion vector representing the expected position over time. In this way an update is necessary only when real position moves away from the excepted one.

The second project, the European project CHOROCHRONOS [132], aimed at integrating concepts from temporal and spatial databases. They started defining a framework for modeling discrete changes over time of various kinds of geometries (e.g., points, lines, regions). In [77, 86, 108, 136] the model has been extended to allow continuous changes. Besides data types for moving points and regions, the model includes also a complete set of operations.

Along with modeling problems, spatio-temporal databases introduced also issues related to new query languages, indexing structures, uncertainty. The querying of spatio-temporal information requires new features in order to be able to use new operators and functions.

2.3.1 Spatio-temporal data and data types

As previously mentioned, spatio-temporal data types allow one to represent continuously or discretely changing geometries. Among spatio-temporal data types, the most important ones are moving points and moving regions. Moving points represent the position of spatial objects, e.g., cars, people, airplanes, or animals. On the other hand, moving regions represent shape and extent of changing spatial objects, e.g., hurricanes, pollution clouds, and spread of diseases.

In literature several frameworks for modeling spatio-temporal objects have been proposed [77, 131, 133, 189, 213, 225]. The two main proposals are those introduced by Erwig et al. [77] and by Sistla et al. [189]. These proposals focus on two different kinds of spatio-temporal databases we briefly introduced before, trajectory databases and tracking databases, respectively.

In [77] Erwig et al. proposed a framework for spatio-temporal data type definitions and operations. In particular, they define moving points and regions as mappings that associate to each time instant the point or region, respectively, valid on it. Moreover, they define a set of operations between them. Proposed operations include, for example, `trajectory` that returns the curve representing the trajectory of a moving point, and `traversed` that computes the region traversed in any time by a moving region. Since some properties of moving objects change over time, authors define similarly also the notions of moving real and moving bool for representing a numerical or Boolean value, respectively, whose value changes over time. For example, the distance between two moving points is represented by a moving real.

The model proposed by Erwig et al. is most suitable for representing and querying trajectories. In [189] Sistla et al. proposed the MOST (Moving Object Spatio-Temporal) model, that is most suitable for querying the current position of moving objects. Authors start from the observation that in order to know the current position of moving objects, if we store directly their positions, we have to update data frequently in order to maintain them. This is not possible for rapidly moving objects or large sets of objects. Thus, they propose to store a motion vector, i.e., they store a position and a function representing the expected movement from the stored position. Of course, after these information have been stored, the object continues to move and its real position can divert from the expected one that can be calculated from the stored position and the motion function. However, this difference between expected and real position cannot exceed a given threshold. Otherwise, an updated position and motion function have to be stored replacing the previous ones. In this way, an update is needed only when the stored position does not reflect anymore the real position. On the other hand, this solution leads to a certain amount of uncertainty in the position since it is just an expected position that can be different from the real one of an amount limited by the chosen threshold. The smaller the threshold is, the smaller the uncertainty is and the greater the number of updates is.

Other proposals focus on the representation and implementation of spatio-temporal data in databases [33,82,125,131]. Some of these proposals have been also implemented in prototypes [31, 125, 173]; however, the classical approach to pair temporal and spatial information for representing spatio-temporal data is used.

In these cases, only some functions for managing and retrieving spatio-temporal information are introduced.

Besides these efforts for implementing spatio-temporal databases, we also note that no commercial DBMS provides direct support for spatio-temporal structures and data. As we mentioned in the previous sections, they provide support for temporal and spatial data. Thus, the current approach used in real DBMSs is to represent spatio-temporal data as pairs composed by temporal and spatial information. Thus, no ad-hoc operations and relationships over spatio-temporal information are currently available in DBMSs.

2.3.2 Spatio-temporal query languages

After TSQL2 [194] and SQL/MM Part 3 [123], there have not been efforts for standardizing the representation of spatio-temporal information in databases. However, based on proposals for representing spatio-temporal information, several spatio-temporal query languages have been proposed [47, 79, 108, 119, 189, 209].

In [47, 79, 108, 209] extensions of SQL [122] have been proposed. These extensions allow one to use the operations and operators proposed in the corresponding representation models. In this way, they allow the user to perform spatio-temporal selection and join. For example the following query, proposed by Erwig and Schneider and based on their model [79] we described in the previous section, retrieves the trajectory of flight with ID UA207 from 7AM until 9AM.

```
SELECT trajectory(route(7.00..9.00))
FROM flights
WHERE id="UA207"
```

where `route` is a moving point (see the previous section) recording the position of a flight over time and `id` identifies a flight. The `trajectory()` function returns a polyline representing the trajectory of a moving point over time. The notation `route(7.00..9.00)` restricts the spatio-temporal object `route` to the time interval starting at 7.00 and finishing at 9.00.

A slightly different approach has been proposed in [189] by Sistla et al.. After introducing the MOST model for representing spatio-temporal data (we have briefly described it in the previous section), they introduce also a query language based on it for performing future queries. They do not extend SQL, they consider cumbersome. Instead, they adapt the FTL (Future Temporal Logic) language they had introduced in [188] for defining temporal triggers in active databases [22]. According to authors, FTL language is more natural and intuitive for expressing future queries on moving objects and it has been studied for being implemented on top of an existing DBMS. FTL queries are specified in the `Retrieve-Where` form, where the `Retrieve` clause specifies which data have to be returned and the `Where` clause specifies the FTL formula representing the condition that the retrieved data must comply with. Besides classic logical operators (e.g., negation and conjunction), FTL allows one to formulate temporal conditions by using the `NextTime` and `Until` modal temporal operators. Temporal conditions can use spatial relations in order to perform spatio-temporal selection.

For example, the following query retrieves the pairs of objects o and n such that the distance between them remains smaller than 5 miles until they both enter the polygon P.

```
RETRIEVE o,n
WHERE dist(o,n)≤5 UNTIL (inside(o,P) ∧ inside(n,P))
```

Similarly to what happens for spatio-temporal modeling, no commercial DBMS offers ad-hoc support for spatio-temporal querying. Thus spatio-temporal querying is based on the usage of temporal and spatial operators in the (usually SQL-based) query languages provided by the DBMS.

3

A motivating scenario: a psychiatric case register database

There are many examples of applications dealing with spatio-temporal data (i.e., data changing both in time and space). Actually, most applications related to spatial information can be extended considering also time. As a matter of fact, it is possible to consider temporal qualification when a spatial or physical survey is repeated more times and spatio-temporal reasoning has to be performed.

Key applications dealing with spatio-temporal data include those providing location-based services [89, 210]. These services keep information about the users' geographical position and movement and inform them about service points they are approaching, e.g., hotels, gas stations, hospitals, and shopping centers. In another class of applications, there is the need to maintain information about the current position of several moving objects and their movement direction. Applications dealing with such kind of issues include road, railway and air traffic control, logistics companies, taxi companies, public transport systems, parcel delivery companies, and tracing of animals movement [106]. There are also several military and security applications, e.g., tracing of friend and enemy armies, position and movement control of criminals and defendants on probation. All these applications need probably of tracking or trajectory databases storing data about moving objects.

Apart from moving objects, spatio-temporal data include also spatial data that evolve over the time. The difference with moving objects here is that these data are not moving, they just change their shape or extent (and thus, in some sense also their position) discretely. Usually, these data can be represented as a spatial object (a point, line, or region depending on the type of the information) with an associated temporal dimension, e.g., the valid time. Such kind of spatio-temporal data include data coming from [46, 97, 111]:

- epidemiology
- biogeography
- agriculture
- geology
- geophysics
- urban planning
- archaeology
- natural resource management

The main goal of these applications is to analyze data according to both classical and spatio-temporal criteria. For this purpose, temporal GIS [231] (i.e., GIS enhanced with temporal capabilities) can be used. Possible analyses that can be performed on such kind of spatio-temporal data include statistical analysis, trend and pattern analysis, and other data mining problems.

In particular, in this thesis we focus on health data coming from the Verona psychiatric case register. We will show how notions and approaches proposed in the thesis can be used for enriching these data and for performing spatio-temporal epidemiological studies. In this chapter, we introduce this motivating scenario. In the next section we introduce spatio-temporal epidemiology, i.e., the context in which the psychiatric case register is used as basis for conducting studies. We describe what the spatio-temporal epidemiology is and what kind of studies it allows to researchers to perform on spatio-temporal health data. Next, in Section 3.2, we describe the Verona psychiatric case register: how it is organized and managed, the data stored in it, and what kind of studies are being conducted on it.

3.1 Spatio-temporal epidemiology

Epidemiology is an important science that studies factors affecting the health and illness of populations (e.g., psychiatric diseases, cardiovascular diseases, respiratory diseases, genetic diseases), and serves as the foundation and logic of interventions made in the interest of public health and preventive medicine [167]. While several studies have been conducted also before 20th century, it has been during the second half of the 20th century that epidemiology started to be developed introducing principles for the design and evaluation of epidemiological studies. Since then, many epidemiological studies have been conducted, also changing in some cases the public way of life. We remember here only the studies about cardiovascular diseases that are still now topical and continue to be developed. Another important example of epidemiological study is the one on the Salk vaccine [200], the largest formal human experiment ever conducted. This study provided the first practical basis for the prevention of paralytic poliomyelitis. Other very discussed and controversial epidemiological studies have been conducted on the effects of tobacco use [207]. These researches have been probably the first studies that attracted the public attention also thanks to the news media work. In last years other epidemiological researches have attracted the public attention: for example, those related to avian influenza [41], severe acute respiratory syndrome (SARS) [43], vaccination and autism [211], passive smoking [44], and acquired immune deficiency syndrome (AIDS) [42].

In first studies the main problem was the disagreement about basic conceptual and methodological points that led in some cases to profound differences in the interpretation of data. Such problem suggested that the methodological foundations of epidemiology had not yet been established, and that it remained young in conceptual terms. Thus in last part of the 20th century the understanding and synthesis of epidemiological concepts has been improved. Despite the surge of epidemiological activity in the late 20th century, epidemiology remains in an early stage of development [163].

In epidemiology several kind of studies can be conducted. A first classification divides studies in experimental and non-experimental. In general an experiment is a set of observations, conducted under controlled circumstances, in which the scientist manipulates the conditions to ascertain what effect, if any, such manipulation has on the observations [177]. In particular, epidemiological experiments are limited by definition to topics for which the exposure condition can be manipulated. Epidemiological experiments include clinical trials, field trials, and community intervention trials. On the other hand, when it is not possible to design (e.g., for ethical problems) experiments, non-experimental (or observational) studies are conducted. Four types of non-experimental studies exist:

- cohort studies in which all subjects in a source population are classified according to their exposure status and followed over time to ascertain disease incidence;
- case-control studies in which cases arising from a source population and a sample of the source population are classified according to their exposure history;
- cross-sectional studies, including prevalence studies, in which one ascertains exposure and disease status as of a particular time;
- ecological studies in which the units of observation are groups of people.

Here we focus on *ecological studies* (the reason will be clear after we introduce our proposed notions and we show how they can be used in Section 7.4) that differ from the other types of non-experimental studies because while in these the subjects of studies are individuals, in ecological studies the unit of observation is a group of people. For this reason they are called also *aggregate studies*. This kind of studies can be conducted, for example, on schools, cities, factories, or nations. The important thing is the availability of measures of exposure and disease distributions in each group. Usual goals of ecological studies include the detection of associations between exposure distributions and disease occurrence [4, 178]. Ecological studies suffer from unavailability of data necessary for adequate control of the results of the analysis [99], but, on the other hand, they are preferred in some cases to individual studies also when it is difficult to measure relevant exposures or doses at the individual level for large numbers of subjects.

An important part of ecological epidemiology is spatial epidemiology. *Spatial epidemiology* extends traditional ecological studies that use explanations of the distribution of diseases in different places to better understand the etiology of diseases [4, 212]. In particular, spatial epidemiology aims at the description and analysis of geographic variations in diseases with respect to several risk factors (e.g., demography, environment, behavior, socioeconomic, genetics, infectious diseases) [4, 17, 75].

Spatial epidemiology is part of a long tradition of geographical analyses dating back to the 1800s, when maps of disease rates in different countries began to emerge to characterize the spread and possible causes of outbreaks of infectious diseases, such as yellow fever and cholera [195, 212].

In epidemiology spatial data represent an important aspect. Spatial information can be used for describing the following information.

- Locations in surveillance data. Surveillance is the continuous analysis, interpretation, and feedback of systematically collected data aiming at observing

and predicting progression of diseases by observing trends in time, place, and persons.
- Spatial determinants of transmission, i.e., any spatial factor that can affect the transmission of infectious agents.
- Landscape constraints, i.e., any environmental property (e.g., land cover, soil type, watershed).
- Spatial association of risk factors and disease, i.e., the spatial correlations between diseases and (spatial) risk factors affecting their progression.
- origins of disease and outbreaks, i.e., the spatial evolution of diseases and their spread.

This information may play a crucial role in helping to explain variability in risk because health status and risk factors vary across space. Geography represents the spatial context and character in which health risks occur. The use of space in epidemiology is often implicit: also in classical (i.e., non-spatial) epidemiology studies the space plays an important role. The importance of spatial data in epidemiology is also proved by the large amount of tools and projects developed in recent years to integrate GIS (Geographic Information Systems) and public health: HealthMapper [229], SIGEpi (Sistemas de Información Geográfica en Salud) [142], GeoDA [11, 196], RIF (Rapid Inquiry Facility) [13, 192] further developed in EUROHEIS project [121] and now sponsored by the Centers for Disease Control and Prevention (CDC) [39].

Examples of spatial epidemiological studies include exploratory multiple-group studies [4, 151], in which the rates of disease among different regions during the same period are compared. The goal in this case is to find spatial patterns, if any exist, that might be used for formulating etiologic hypotheses (i.e., hypotheses about the causes of diseases). For example, a study by Mason et al. [143] mapped the cancer mortality rates in the United States by county for the period 1950-1969 and it found a difference in geographic patterns by sex for oral cancer: among men, the mortality rates were greatest in the urban Northeast; but among women, the rates were greatest in the Southeast.

Moreover, spatial data play an important role also in other classes of epidemiological studies, e.g., social epidemiology [127] and environmental epidemiology [116]. Social epidemiology looks for relations between social factors (that may change across different regions) and diseases in populations. Environmental epidemiology searches for relations between environmental factors (i.e., factors that are exogenous to and nonessential for the normal functioning of human beings, for example, physical, chemical, biologic agents, social, political, cultural, and architectural factors) and diseases in populations.

Epidemiological studies are not tied only to space, but also to time. Health status and risk factors vary also across time [1, 70]. Changes in data over time are useful to understand the temporal evolution of risks and diseases. In a pure exploratory time-trend or time-series study [4, 151] the disease rates over time of a geographically defined population (i.e., the population is not classified in different groups with respect to space) are compared. This approach allows one to study where and how risks and diseases start, discover time trends, and, possibly, to predict their future evolution. Among exploratory time-trend analyses, an important kind of studies used by epidemiologists is the age–period–cohort analysis [151]. In

this kind of studies, a collection of past data from a large population during a period of at least 20 years is analyzed in order to estimate the separate effects of three time-related variables on the rate of disease: patients' age, period (calendar time) when disease cases have been surveyed, and birth cohort (patients' year of birth). Analysis results can be represented through graphs depicting age-specific rates on the y axis while period or cohort represent the x coordinates of the graphs. Again, the goal is to find time trends and time-related etiologic hypotheses. For example, in [135] an age–period–cohort analysis of melanoma mortality among white men in the United States between 1951 and 1975 has been conducted. This study showed that persons born in more recent years experienced a higher rate than did persons born earlier.

Of course, multiple-group spatial-based studies and time-trend studies can be mixed in order, for example, to find, describe, or predict time trends in the disease rate for populations in multiple spatial regions. This kind of trends can be called spatio-temporal trends. We note that while spatial analysis has been already exploited in epidemiological studies, space-time models have been only recently and partially investigated [1].

Multiple-group and time-trend epidemiological studies are based on the grouping of data with respect to different temporal and spatial granularities. For example, in age–period–cohort analyses data may be grouped according to patients' year of birth, while in multiple-group studies patients may be partitioned according to their domicile. However, epidemiologists may use different granularities also in the same study, e.g., in an age–period–cohort study birth dates may be grouped with respect to years while the periods when diseases have been surveyed may be grouped according to semesters. Thus, also in epidemiology researchers have to deal with issues related to multi-granularity (see Chapter 1).

As a matter of fact, in epidemiological studies health risks are often mapped to relatively arbitrary administrative areas (the level at which population and disease data are available), but risks can be sensitive to changes in the scale of output. It is known as the "modifiable area unit problem" [157]. This is a fundamental problem in spatial analysis due to the following remarks:

- there are numerous ways (i.e., granularities) to partition space and time;
- data are usually presented at one particular granularity;
- granularities can be combined or split to form new granularities;
- solutions depend on the chosen granularity.

Hence, grouping data by using different spatial and temporal granularities or aggregating data to different areal arrangements will lead to variations in the results, which may affect the interpretation of findings. This problem arise also in exploratory multiple-group studies [151] that we have introduced before. For example, spatial regions with few observed cases show greater variability in the estimated rate than region with many observed cases. Moreover, investigators need to use several spatio-temporal granularities because each granularity is more effective on some data rather than on other ones [112]. For example, some allergies rates may be analyzed aggregating data with respect to *Seasons* while psychiatric diseases may be grouped according to *Days*.

Because of all these remarks, investigators need to use and cross-check several different granularities, determining the most appropriate geographic and temporal boundaries. Since boundaries data used for health data can change over time, two related issues come out [17].

The first one regards the problem of inconsistent geography, e.g., a spatial granule may change its name and the system must trace this event to manage properly spatially qualified data that refer to this granule. The second one requires to pass information from a granularity to another one, e.g., to report disease at national or regional scale studying variations in disease occurrence rates at a local (small-area) scale [17,75]. Both problems are important issues related to multigranularity (spatial and temporal) and need to be faced in any proposal about spatio-temporal granularity.

Considering methodologies used in epidemiology, we can see that spatial epidemiology consists mainly of searching and studying disease clusters and patterns. For this purpose investigators use statistical (spatial and temporal) analysis to make ecological inferences. An ecological inference is when conclusions about associations among variables at the individual level are made from observations at an aggregate level [184]. So, even though the final analysis is done at the individual level, cluster analysis is made using aggregated data. In this phase, a temporal-GIS can summarize data and measure variables that affect the individual. In other words, a temporal-GIS is useful for processing data to determine the role of factors that occur at or can be measured for an area or neighborhood.

A cluster is an unusual aggregation, real or perceived, of health events that are grouped together in time and space [40]. The main goal of epidemiologists is more looking for patterns (spatial, temporal or both) in data than searching for specific associations between agent and disease [40]. For this purpose, epidemiologists study disease rates and patterns for a geographic area or time period and compare them with the ones of other areas (adjacent or similar areas), times (previous or subsequent periods), or with the expected ones.

From the above discussion, it is clear the strong link between epidemiology, spatio-temporal granularities, and spatio-temporal data.

3.2 The case of study: the psychiatric database PCR

In Chapter 7 we will show how the proposed framework for spatio-temporal granularities may be used for enriching and querying a spatio-temporal database. In particular, we will show several examples of queries on spatio-temporal clinical data recorded in the Verona Psychiatric Case Register (PCR). Verona is a city in North-East of Italy with about 260,000 inhabitants. The Verona Health District comprises the Verona municipality and other 35 municipalities around the city, for about 460,000 inhabitants. The District is also partitioned in four catchment areas, each one composed by some municipalities. Moreover, biggest municipalities (including Verona itself) are divided in subzones, corresponding to quarters. Each catchment area is served by a Community-based Psychiatric Service (CPS). CPSs aim at providing responses to psychiatric patients' practical as well as psychological and social needs, while trying to alleviate and control their symptoms.

Special emphasis is given to integrating different interventions, such as medication, family support, and social work. For this reason permanent staff includes psychiatrists, clinical psychologists, social workers, health visitors, community nurses, ward nurses and counselors. Moreover, CPS structures include

- a psychiatric ward;
- an outpatient department providing psychiatric consultations and individual and family therapy;
- a consultation liaison office that maintains psychiatric integration with other hospital-based medical activities and ensures continuing contact with psychiatric patients when they are hospitalized for medical reasons;
- a 24-hour accident and emergency room;
- a night and week-end emergency room;
- a 24-hour staffed hostel, a group home, and two apartments, offering different levels of supervision.

CPS provides also home visits. These ones can be both in response to emergency calls and, for chronic patients, planned in advance for offering regular, long-term support. Each patient must refer to the CPS leading the area where he lives.

CPS is well integrated, and allows easy and informal access to patients. It is a public service run by the National Health Service. Thus, payment is not required, except for a fee for out-patient visits.

The Verona Psychiatric Case Register (PCR) is an information system collecting information about patients' accesses to CPS since 1979 [9]. At the first contact with the psychiatric service, socio-demographic information, past psychiatric and medical history, and clinical data are routinely collected for patients aged 14 years and over. These data may be updated at successive contacts when required. Each patients' contact with CPS structures is recorded in PCR. Recorded contacts with psychiatrists, psychologists, social workers and psychiatric nurses include:

- attendances at the out-patient clinic;
- domiciliary visits;
- telephone calls;
- day cares provided at the day hospital units;
- all admissions to the acute psychiatric ward and private clinics.

On the other hand, psychiatrists and psychologists in private practice and general practitioners do not report to the PCR.

To each patient a diagnosis is assigned according to ICD-10 categories and then coded into 12 standard diagnostic groups. The International Statistical Classification of Diseases and Related Health Problems 10th Revision (ICD-10) is a coding of diseases and signs, symptoms, abnormal findings, complaints, social circumstances and external causes of injury or diseases, as classified by the World Health Organization (WHO) in 1990 [230]. The diagnosis may be updated at successive contacts if necessary. For all patients who have been registered in the PCR, death and migrations (across or outside the four catchment areas) are recorded by means of annual checks with demographic records of the municipalities of the catchment areas. Data on about 28,700 patients and more than 1,500,000 psychiatric contacts have been recorded in PCR up to now.

Besides patients' personal data (e.g., birth information, health insurance card number, sex, nationality, contacts), patients' medical record, and contact information (including duration, involved operators, motivation, contact kind and conclusions), PCR information system records also education, employment, professional, cohabitants, and marital status of patients.

PCR is used for administrative, clinical, and research purposes. Administrative uses include the analysis of incidence and prevalence rates, number of patients seen, and number of visits made over different time periods for reporting them to local, regional and national level health administrations. Moreover, PCR is used as a basis for calculating direct costs for different groups of patients [7] and for monitoring the effects of changes in resources, organization, and needs.

Clinical purposes include the monitoring of patients who have been in contact with the service for organizing contacts at regular intervals and the provision to clinicians of reports about admissions and contacts for individual patients in a given time period.

Besides their clinical work, psychiatrists and psychologists, along with other staff members, take an active role in research and teaching. Research activities include studying and analysis of the Verona Psychiatric Case Register (PCR). In research activities, PCR is used for:

- epidemiological studies about the utilization of services: for example in [8] Amaddeo et al. investigate the relationship between the lunar cycle and the frequency of patients' contact with CPS;
- studies about the correlation between geographical factors (e.g., position and distance of services) and service utilization [235];
- discovering services-related, area-based, and socio-demographic factors influencing accesses to health services [164, 198, 199];
- studies on mortality among psychiatric patients [102];
- comparisons with other case-register areas [176].

Some of these research studies are based on temporal and spatial analysis of the PCR data. Temporal information are contained in several parts of PCR. For example, patients' contacts are temporally qualified with the date in which they occurred, while all personal information about patients (e.g., marital status, employment, and diagnosis) have an associated valid time period. As we mentioned in the previous section, these temporal data can be used for conducting time-trend or time-series studies. For example, epidemiologists may be interested in to know the number of contacts in different time periods with respect to different factors (e.g., patients' age, diagnosis, and year of birth). Similar multiple-group spatial studies can be conducted by considering spatially qualified data stored in PCR. These data include patients' domicile addresses, that can be depicted in a spot map [167] (i.e., a map showing the geographic location of people according to a specic attribute) like the one in Fig. 3.1. This information allows us to locate patients in several spatial groups, i.e., granularities, (e.g., municipalities, census tracts, catchment areas) and to analyze PCR data with respect to these groups.

A first phase of such kind of studies allowed epidemiologists to develop the atlas of Verona psychiatric services [234]. This is a WEB-based interactive atlas based on data contained in PCR that allow one to report, displayed on a map,

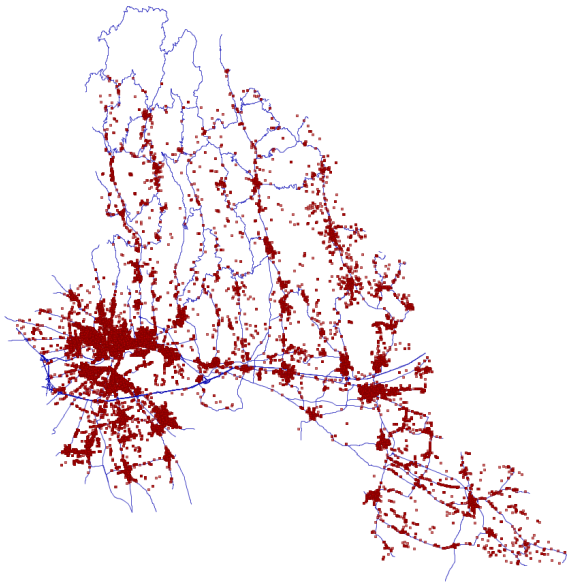


Fig. 3.1: Patients' domicile contained in PCR.

the spatial distribution of several indicators. Indicators include socio-demographic factors (e.g., population density), employment factors (e.g., employment distribution among various industry sectors), composition of family units, health factors (e.g., number of voluntary hospitalizations, average length of hospitalizations), and diagnosis type. These factors can be calculated and displayed with respect to different spatial aggregation (i.e., granularities): municipalities, census tracts, and catchment areas.

Currently, research directions include the development of epidemiological studies aiming at discovering temporal, spatial, and spatio-temporal trends and patterns in factors calculated from PCR data. Factors include those already taken in consideration in the atlas we mentioned. On the other hand, the goal is to further enrich analyses linking PCR with other spatio-temporal data that can be used for qualifying patients and contacts data and factors. These new spatio-temporal information include, for example, pollution data (e.g., air pollution with respect to PM10, ozone, and carbon monoxide and night and day noise pollution). Some of these data are only spatially qualified, i.e., they represent the average pollution level in space regions in a fixed time period, while others are spatio-temporally qualified and represent the evolution over time (e.g., the daily evolution) of pollutant levels in several spatial regions. Pollution data allow us to partition (i.e., to define new spatio-temporal granularities) Verona's region with respect to pollution levels. For example, we defined five ranges for day noise pollution. These ranges represent low, lower-middle, middle, upper-middle, and high levels of noise pollution. Thus, municipalities can be grouped based on the average level that has been surveyed, obtaining a spatial partition similar to that depicted in Fig. 3.2, where pollution levels have been represented with different colors. Granularities like this

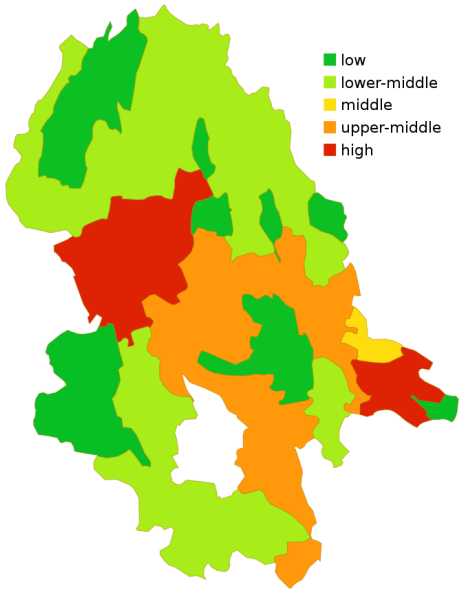


Fig. 3.2: An example of partitioning of municipalities with respect to pollution levels.

one allow us to spatio-temporally qualify, aggregate, and analyze PCR data, e.g., with respect to patients' domicile (Fig. 3.1). For example, we may retrieve the number of contacts for patients living in municipalities with high noise pollution. Of course, these spatial aggregations can be mixed with other usual constraints on data. For example, we may retrieve the number of contacts for patients grouped with respect to their current diagnosis and living in municipalities with high noise pollution. Similar requests can be formulated also based on temporal constraints. For example, epidemiologists may be interested in to retrieve the number of contacts in each month grouped according to the patients' diagnosis.

Since different pollution data refer to different spatial partitions (e.g., noise pollution is based on municipalities while PM10 pollution is based on regions defined around the position of environmental stations that survey pollution levels), PCR data can be analyzed with respect to different granularities, as required by principles for epidemiological studies, as previously outlined. Thus, for example, a database query may retrieve some data aggregated with respect to PM10 concentration levels and other data aggregated with respect to catchment areas or municipalities.

Finally, as mentioned in the previous section, in epidemiology the spatial multiple-group and time-series approaches can be mixed for discovering spatio-temporal patterns and trends. Such studies can be conducted on PCR by retrieving data based on both spatial and temporal constraints. For example, epidemiologists may retrieve the number of contacts for patients grouped by diagnosis and noise pollution level in each season. Note that some PCR information are temporally qualified; thus in previous queries we have to consider also the evolution over time

of these data and retrieve them correctly. For example, patients' domicile address may change over time. In this case, when we retrieve the number of contacts for patients living in highly acoustic polluted areas, we have to consider the domiciles valid at the time instant when contacts occurred.

In next chapters, we will propose a formal definition of spatial and spatio-temporal granularities in order to manage partitions like those defined by pollution data. In Chapter 7 we will see how granularities can be applied for querying PCR data and to accomplish requests, like those previously mentioned, that could be useful in epidemiological studies based on PCR. We will show that proposed notions allow one to aggregate data in several (temporal, spatial, and spatio-temporal) granularities in the same query.

4

A framework for spatial granularity

Conversely to temporal granularities [27], spatial granularity is not a widely accepted and well-known notion. In literature the term "spatial granularity" has been used for several different meanings. Some proposals use this term to represent "multi-resolution" (i.e., the representation of spatial data at different levels of detail with different resolutions) and other notions [69, 84, 85, 203]. On the other hand, main proposals about temporal granularities and some proposals about spatial and spatio-temporal granularities are based on the idea that granularities can be considered as meta-data through which spatio-temporal data can be managed and queried. By using this approach, granularities can be used for qualifying and enriching other spatio-temporal data in order to exploit their spatio-temporal information.

If we consider spatial granularities in the same way we consider the temporal granularities, we may think that a spatial granularity represents a partition (generally, non total as it happens for the temporal one) of a spatial domain, i.e., a division of a space domain in non-overlapping regions called granules. Particular cases of spatial granularities are the common units of measure (e.g., m^2 and km^2) where the Earth surface is divided in regular and all equal granules. Other typical examples of spatial granularities include administrative subdivisions (e.g., quarters, municipalities, provinces, and countries), that partition (a part of) the Earth surface in granules different from each other and, eventually, with holes inside or between the granules. For example, let us consider the spatial granularity representing countries in the European Union. Each granule represents a country and has a label (i.e., a name) that identifies it. Between granules (i.e., countries) there are holes (e.g., the sea), and there are disconnected regions (e.g., Corsica and Sardinia islands).

Considering spatial data as meta-data, spatial granularities can be used for spatially qualifying classical data, i.e., for adding to classical data a spatial location. Spatial qualification based on granularities can be achieved in two ways:

- adding to classical data a space location that can be subsequently used for locate data in granules;
- adding to classical data a reference to one or more spatial granules, meaning that the qualified information in located inside or is about the related granules.

For example, we can associate data (e.g., population, birthrates, and Prime Minister names) with a spatial qualification at country level by using the granularity representing countries in European Union.

An important difference between temporal and spatial granularities is the domain dimension. Usually, time is represented by using one dimension (i.e., the directed time line); on the other hand, geographical spaces are represented by using more dimensions. The multidimensional nature of space creates new problems and possibilities in the management of granularities. For example, while among time instants and granules (in the definition proposed by Bettini et al. [27]) there is always exactly one total order, between spatial points and granules generally this order does not exist and several different relationships may be defined (e.g., direction relations, metric relations, and orders based on space filling curves [179]). This difference leads, conversely to what happen for temporal granularities, to the need for explicitly representing relationships between spatial granules.

In this chapter we present our frameworks for spatial granularities. The proposed frameworks deal also with previous considerations. In Section 2.2.3, we introduced the two models for representing spatial data: the vector and the raster model. We propose a framework for spatial granularities for both these models. Thus, the chapter is organized as follows. In the next section, we present related work about definitions of spatial granularities on both vector and raster models. In Section 4.2 we introduce our framework for spatial granularities based on vector data. As we have already mentioned in previous chapters, the framework includes formal definitions of spatial granularity and related concepts. Moreover, we define operations useful for creating new granularities from already defined ones, and relationships between granularities that can be used during reasoning phases. Following the same structure, in Section 4.3 we introduce our framework for spatial granularities based on raster maps.

4.1 Related work

A definition of spatial partition is presented in [78]: Erwig et al. define a partition as a function from the space point ($\mathbb{R}^2$) to the set of labels and define several operations needed to modify and create them. Moreover they demonstrate that the set of all partitions is closed with respect to these operations. This definition of partition focuses on points and not on regions, which are just subordinate entities defined as sets of points with the same labels. Besides, they do not use an "index" or a symbolic representation for partitions.

This lack is partially solved in [145], where McKenney and Schneider, using the definitions presented in [78], show how to represent a partition using a graph whose nodes represent intersection points between regions and whose edges represent the boundaries of the regions. Moreover they associate labels no longer with points but with spatial regions identified by cycles in the graph. However, they limit the use of graphs to the representation of a partition without to consider relations between regions of the partition: in fact, these relations are neither investigated nor represented.

Using an ontology-based approach, in [193] Smith and Brogaard present a theory to incorporate granularities and mereotopology. For them a granularity, which they call "granular partition", is a way to classify the reality into classes. A granularity is then a cognitive artifact and it is similar to apply a grid onto a certain part of the reality.

They represent a granularity as a labeled system cells in which each cell is projected by the user onto a particular fact of the world, e.g., a municipality. However, they do not focus only on spatial granularities but also on all possible taxonomies and classifications of the reality. Then, they do not define exactly what is a spatial granularity but what is, in general, a partition and access it using cells.

In [214], Wang and Liu define a spatial granularity as a mapping from an index set to a spatial domain, similarly to the approach used in the temporal context. Their granules represent non-overlapping regions and are totally ordered using a generic index set isomorphic to $\mathbb{N}$. They also define the finer-than relationship between granularities following the temporal one. However, they polarize their attention on modeling spatio-temporal granularities: they model only those aspects of a spatial granularity that are closely related to the definition of a spatio-temporal granularity. They do not study neither possible relationships between granules of a spatial granularity nor relationships between different granularities.

A similar approach is used by Khatry et al. [130]: the authors propose an annotation-based conceptual model for designing spatio-temporal databases. Objects in a conceptual model can be annotated for providing them with spatio-temporal qualifications based on spatio-temporal granularities. For this purpose, they extend Worboys's proposals [226, 227] and define a spatial granularity as a mapping that associates to each index in an index set a set of points in the space domain (i.e., a granule). Granules in a spatial granularity cannot overlap. Moreover, they define *finer-than* and *groups-into* relationships between spatial granularities extending the ones defined on temporal granularities by Bettini et al. [23, 24]. They use these relationships for organizing granularities in lattices. Finally, following again the Bettini et al.'s approach, they define some notions related to spatial granularities: *anchor* and *image*.

In [203], Timko et al. use a space model based on the partitioning of roads into a sequence of atomic line segments, termed space granules. Timko et al. do not define a complete notion of spatial granularity, but they use granules to qualify cars' position on roads and, then, aggregate cars position through a data structure they propose, Sequenced Spatio-Temporal Tree.

In [36] Camossi et al. discuss some issues related to the definition of spatial granularities already present in literature and oriented to spatial databases. Then, they introduce a formalization of spatial granularities that solves these issues.

They define spatial granularities following the standard temporal definition [34, 35, 37], i.e., as a mapping from an index set to a spatial domain. In particular, they formalize the notion of granularity and one possible relationship between granularities, finer-than, whose definition follows the temporal one. They give an object-oriented formalization extending the ODMG (Object Data Management Group) model [38, 155] with spatial granularities. They define new basic and multigranular object-oriented types able to represent temporal and spatial information. After that, they defined two spatial geometric conversion operators

implementing generalization and refinement operations for geometrical and quantitative attributes. These operations allow one to transfer information from a finer (coarser) granularity to a coarser (finer) one. They use granularity lattices and show how to handle data conversion between different granularities related by a relationship using generalization and refinement operators.

Camossi et al. use a multiresolution-based approach to spatial granularities. Hence, they represent a spatial granularity associating geometric data (e.g., polylines representing regions), describing its extent, to a unit of measure (e.g., *cm*, *m*, *km* but also *mq*, *kmq*), describing the resolution level of the data. Moreover they allow one to represent also granularities that do not represent regions (i.e., partitions of a space domain) but also one-dimensional data (e.g., rivers, railways, and roads).

However, as Wang et al., Camossi et al. focus on spatio-temporal granularities, so they do not explore and analyze completely the notion of spatial granularity.

Camossi et al. implement and extend the framework presented by Stell and Worboys in [197]. Stell and Worboys define a formal approach to multigranularity, multi-resolution, and vague representation of spatial data. Authors define a granularity as a particular semantic and geometric level of detail. Their basic notion is the "stratified map space", a structure composed by a granularity lattice and, for each granularity, a map space, i.e., a set of maps. A map denotes a finite collection of data; an example may be a set of objects having classical and geometric attributes (e.g., data conveyed by a classical paper map). Maps are grouped together in map spaces, i.e., a set of all possible maps described using some fixed objects (i.e., maps covering the same extent). Inside any map space, maps are partially ordered by precision (i.e., resolution); however, precisions are limited by the level of detail represented by the granularity referred by the map space. Hence, if m_1 and m_2 are maps belonging to the same map space, we say that $m_1 < m_2$ if m_1 is a more precise version of m_2. A granularity represents a level of detail: since levels are related with respect to their precision, it is possible to define a granularity lattice. Finally, Stell and Worboys define the notion of stratified map space, their main idea. A stratified map space represents an extent at different levels of detail by using a granularity lattice G. For each granularity $g \in G$, a map space $\texttt{Maps}(G)$ is defined containing all maps representing the extent at the level of detail represented by g. Moreover, they define two functions able to transfer information between maps expressed in different granularities: `Gen`, that transfers from a higher to a lower level of precision, and `Lift` that transfers from a lower to a higher level of precision.

Another different definition and approach to spatial granularities is introduced by Schmidtke and Woo [181]. Authors use a mereotopological approach where granularities are considered as a multi-resolution representation of a space domain. Their proposal is based on the notion of "grain-size", i.e. the relevance of grains. The relevance of grains induces a total order over grains and it is a way to decide if a grain is important or not and then if it should be represented or not in a given resolution.

Schmidtke and Woo give an axiomatic definition of granularity, grain, and their relationships. Thus, they provide logical formulas stating the properties of grains and locations. For example, the sizes of locations are totally ordered by the *leq*

relation, with the usual properties, and any part of a location is smaller or with the same size of the location itself. Based on this notion of "size", authors define what means that a grain region is contained in a context region. This relation between grain locations and context locations denotes the membership of a grain to another one. Properties defined by authors state when a grain can be defined inside another one. For example, if a grain g_1 is located in the context c_1 and the grain g_2 is located in the context g_2, then g_1 is smaller than g_2 if and only if c_1 is smaller than c_2. In other words, grains are ordered in the same way as their respective context locations and vice versa.

Finally, Schmidtke and Woo define two relationships on granularities. A spatial granularity c_1 is finer than c_2 if and only if there is a grain of c_2 that is larger than c_1, while they are comparable if c_1 is not smaller than any grain of c_2 and c_2 is not smaller than any grain of c_1.

The main lack of their proposal is that it does not represent explicitly the link between the high-level representation of a granularity and the spatial level (i.e., the geometric data). Moreover, their formalism does not allow an easy representation of other spatial relations.

Finally, in [204] Tomlin and Berry introduce the map algebra, a set of operations for dealing with raster maps. They use the same definition of raster map we introduced in Section 2.2.3, where a map is divided in locations identified by pairs of Cartesian coordinates and are associated to a measured physical characteristic. Based on this definition, operations in the map algebra allow one to create new maps based on already defined ones. Several operations have been defined. These operations can be classified in three categories: local, focal, and zonal operations [67]. Local operations compute the value on each location as a function of the value of the same location in other maps given as parameters. These operations include for example arithmetic operations with another map. Focal operations compute the value for each location in the resulting map based on the value of neighboring locations in given maps. For example, focal operations include the calculation, for each location, of the average of the values in the neighboring locations and its assignment to the considered focal location. Zonal operations compute the value for each location based on the values from one existing layer (the value layer) which are associated with that location's zone (i.e., the locations around the considered resulting location with equal values) on another existing layer (the zone layer). For example, zonal operations allow one to compute for each zone in the zone layer the max value in the value layer.

4.2 Vector-based spatial granularities

In this section, we introduce our framework for spatial granularities based on vector data, including properties of granularities, relationships between granularities and operations for creating new granularities from already defined ones.

As we said above, we follow the same approach to spatial granularities used by Bettini et al. for defining temporal granularities [23, 27, 215]. In our definition, each granularity has two levels: the spatial domain, where the spatial regions are defined, and the index set used to access and manage the granules.

However, there are many deep differences between temporal and spatial granularities, and we have to consider these differences when studying a framework for spatial granularity. In particular, differences are about granules and relations between them.

Considering granules, we note that many useful temporal granularities are generally a "regular partition" of the time line and there is some kind of periodicity for granules extents (e.g., all days are equal and they are repeated periodically). Conversely, a spatial granularity usually is not regular and its granules may have any possible shape without any kind of repetition or periodicity.

Considering relations between granules, the time line, and also the time granules, is totally ordered by the "successor" relation (we always know whether an instant precedes another one or vice versa) and thus the index set may be any infinite discrete ordered set (e.g. $\mathbb{N}$). Since the successor relation is unique and well-known, it is not necessary to explicitly represent it in each temporal granularity definition. Conversely, in a spatial domain we may find many possible orders between points and granules (e.g., several definition of direction-based relations), thus we must use a flexible data structure able to represent them explicitly. For this reason we have chosen to adopt graphs as index set for spatial granularities, as we will discuss in Section 4.2.2. Another consequence of these differences is that some notions defined in the framework for temporal granularity have no counterparts in the spatial one.

4.2.1 Spatial domain and multidigraph

First, we have to define two layers a spatial granularity: the spatial domain and the index set.

We may have two kinds of spatial domain: a continuous one and a discrete one.

Definition 4.1 (Spatial domain). *A* continuous *spatial domain is a connected subset of* $\mathbb{R}^2$ *(or* $\mathbb{R}^3$*) without holes. Thus, it is an infinite set of points.*

A discrete *spatial domain is a regular subdivision (grid of cells) of a chosen space (e.g.* $\mathbb{R}^2$ *or* $\mathbb{R}^3$*). Therefore, it is a numerable set of cells, called* chorons*.*

A relation over points (or chorons) inducing a ordering on them is associated to each spatial domain.

Chorons are the counterpart of the temporal chronons (i.e., an indivisible span of time) and using them we may focus on discrete domains, as it has been done for temporal databases through chronons [124].

Example 4.2. Given the spatial domain $SD = \mathbb{R}^2$, we may associate to it the direction relation $North$ such that we have $North(P, Q)$ only if the latitude of P is lower or equal to the one of Q.

As we will see in the next section, the index set used in spatial granularities is defined as a multidigraph, thus we give hereby the definition of the notion of multidigraph.

Definition 4.3 (Multidigraph). *A labeled multidigraph MG is a 8-uple* $<V, MA, \Sigma_V, \Sigma_A, s, t, l_V, l_A>$ *representing a labeled directed graph with multiple labeled edges and such that:*

- V *is the set of nodes;*
- $MA = (A, m)$ *is the multiset of edges. The multiset is composed of the set of edges* $A \subseteq V \times V$ *and the function* $m : A \to \mathbb{N}$ *that for each edge in* A *gives its multiplicity.*
- Σ_V *is the finite alphabet of node labels;*
- Σ_A *is the finite alphabet of edge labels;*
- $s : A \to V$ *is a function indicating the source node of an edge;*
- $t : A \to V$ *is a function indicating the target node of an edge;*
- $l_V : V \to \Sigma_V$ *is the labeling function for the nodes, it is a bijection;*
- $l_A : A \to \mathcal{P}(\Sigma_A)$ *is the labeling function for the edges.* l_A *must associate to each edge as many labels as its multiplicity, then we impose that, for each edge* $e \in A$, $|\, l_A(e) \,| = m(e)$.

4.2.2 Spatial granularities

The definition of granularity is given incrementally, thus only at the end of this section the presentation of all details of the definition will be completed. First we give the definition of granule.

We remember that [183]: given a point set $S \subseteq SD$: a point $x \in S$ is an interior point if a neighborhood of x, N, exists (i.e., the set of points around x within a non-null radius) such that $N \subseteq S$. x is an exterior point if a neighborhood of x, N, exists such that $N \cap S = \emptyset$. A point x is a boundary point of S if every neighborhood of x contains at least one interior point of S and at least one exterior point of S. A point x is a closure point if it is either an interior or boundary point. The set of all interior points of S is denoted with S°; the set of exterior points of S is denoted with S^{-}; the boundary of S is denoted with ∂S and the set of closure point of S is denoted with $\overline{S}$. A set S is said to be closed if $SD \setminus S$ is open and S is open if and only if, for each $x \in S$, there exists a neighborhood O of x such that $O \subset S$. Finally a closed subset S of DS is said to be regular if $S = \overline{(S^{\circ})}$.

Given these definitions of closed and regular sets, we can define a granule as follow.

Definition 4.4 (Granule). *Given a spatial domain SD, a granule is any closed and regular subset of SD. We call* $SG_{SD} \subseteq \mathcal{P}(SD)$ *the set of all possible granules of SD and* SR_{SD} *the set of all possible spatial relations among granules of SD. Each relation R is represented as a mapping, that given a granule* $g \in SG_{SD}$ *returns the set of granules related in R to g.*

Note that, with this definition, we allow granules to have holes and unconnected regions.

Definition 4.5 (Spatial granularity). *A spatial granularity* $\mathcal{G}$ *is a 4-uple* $<SD, MG, D_A, G>$ *where:*

1. *SD is a spatial domain;*

2. *MG is a multidigraph;*
3. *$D_A : MG.\Sigma_A \to SR_{SD}$ is a mapping from the edge labels to the relations between the granules that are represented in the granularity;*
4. *$G : MG.V \to SG_{SD}$ is a mapping from nodes of the multidigraph to spatial granules of the granularity such that for each node $v \in MG.V$, $G(v) \neq \emptyset$ and any pair of granules must have disjoint interiors.*

For any pair (v_1, v_2) of nodes of $MG.V$ and any label $l \in MG.\Sigma_A$, given $g_1 = G(v_1)$, $g_2 = G(v_2)$, and $R = D_A(l)$, in MG there must be an edge labeled with l between v_1 and v_2 if and only if the two granules g_1 and g_2 are related by the relation R, i.e. $g_2 \in R(g_1)$.

The multidigraph *MG* is a high level representation of the spatial granularity. The graph allows one to manage and query spatial information more efficiently than by the direct use of granule geometries. Each node represents a different granule and, since edges may be labeled with more labels, each edge represent relations between two granules.

The last part of the definition of spatial granularity introduces a constraint ensuring that the multidigraph reflects the spatial information. For the same reason, each edge label corresponds to a spatial relation in SR_{SD} defined between granules. The spatial meaning of the edge labels is defined by the total function $D_A : MG.\Sigma_A \to SR_{SD}$ from the set of edge labels to the spatial relations over the granules of GS_{DS}. For example, if $N \in \Sigma_A$ might be that $D_A(N)$ is the mapping defining the usual spatial relation "North". Note that two different granularities may have distinct labels for the same spatial relation.

Two granularities with the same spatial domain and the same mapping between nodes and granules are different if their functions D_A are different: indeed, a different definition of edge labels could produce different edges and a different multidigraph.

In the following, with G we indicate, when it is not ambiguous, both the mapping from the nodes to the granules and the whole granularity, thus we use G for $\mathcal{G}$. The components (e.g. graph, spatial domain) of a granularity G are denoted by using the dot notation, e.g., $G.MG$, $G.SD$, $G.MG.l_A$. Moreover, when we say that there exists an edge between v_1 and v_2 labeled with l, written $(v_1, v_2)_l$, we express the fact that there exists the edge $(v_1, v_2) \in A$ and $l \in l_A(v_1, v_2)$. Finally, since the mapping G from the nodes to the granules is a bijection, we can indicate without ambiguity a node or a granule with its label. For the same reason, when we talk about a spatial relation over the granules we use its corresponding edge label. Thus, we use R for both the relation between granules defined on the space domain and its label. Moreover, since it is not possible to confuse the components of a multidigraph in a granularity with other components of the granularity itself, we omit MG from the dot notation of the graph components in a granularity G. Thus, we use, for example, $G.V$ for $G.MG.V$ or $G.l_A$ for $G.MG.l_A$.

We impose two restrictions on any granularity $\mathcal{G}$. First, $\mathcal{G}$ cannot contain empty granules, i.e., for each node $v \in G.V$, $G(v) \neq \emptyset$. Second, granules of $\mathcal{G}$ must have disjoint interiors. Due to geometrical issues depending on the definition of the spatial domain, we allow granules to have common boundaries. Formally this restriction stand for the spatial domain which we use. If the spatial domain is a

regular subdivision, then we impose that for each pair of nodes $v_1,\ v_2 \in G.V$, $G(v_1) \cap G(v_2) = \emptyset$, so we must define the chorons in such way that they must be disjoint. A possible solution, similar to the one used in the temporal context, is to define each choron closed in the upper and left sides and opened in the remaining two sides. Otherwise, if the domain is a continuous space domain (e.g., $\mathbb{R}^2$), we impose that each pair of granules have disjoint interior, i.e., $(G(v_1))^\circ \cap (G(v_2))^\circ = \emptyset$. In such cases, since the granules are closed and the boundary belongs to the granule, two contiguous granules share the points on their boundary.

Summarizing, the construction of a granularity is composed of the following steps:

1. we identify the spatial domain and the associated relation over its points or chorons. In many applications the domain is determined by the representation of the spatial data we are interested in;
2. we identify the relations we want to represent in the granularity (e.g. the eight direction relations based on cones and centroids [90]). For each relation we must pick a label acting for it in the graph;
3. we define the total function D_A mapping each edge label to the spatial relation it represents;
4. we determine the granules and for each of them we create a node in V and a label in Σ_V;
5. we define the function G that maps each node in the graph with its correspondent granule;
6. we construct edges between nodes in the graph by using the following rule: for each pair of nodes v_1 and v_2 in V and each edge label $l \in \Sigma_A$, an edge $(v_1, v_2)_l$ exists in MG if and only if $G(v_2) \in D_A(l)(G(v_1))$.

Example 4.6. Fig. 4.1 depicts an example of a spatial granularity for the Verona Community-based Psychiatric Service catchment areas (see Section 3.2) with the usual eight direction relations. For clarity here only some edges (i.e., relations) between the granules are depicted. In the figure we see the two layers composing a spatial granularity: the spatial domain represented by the geometrical information about catchment areas, and the multidigraph representing the spatial information at higher level. In the figure, dashed lines represent the mapping from nodes to granules, while continuous lines represent edges between nodes. We can see that between two granules several relations may exist (e.g., the *NorthWest* area is West, Northwest, and North of the *CentreEast* area).

Considering a granularity G, we define two functions. Given a node v, the mapping $step_R : G.V \to \mathcal{P}(G.V)$ returns the set of nodes reachable from v by edges labeled with R, i.e.

$$step_R(v) = \{w \in G.V \mid \exists (v, w) \in G.A \wedge R \in l_A(v, w)\}$$

where (v, w) is the edge from v to w, and $R \in l_A(v, w)$ means that the label R is among the labels associated to the (v, w) edge.

The second function is $next_R$; it is defined considering the construction of an optimized graph obtained deleting the edges derivable from other edges by

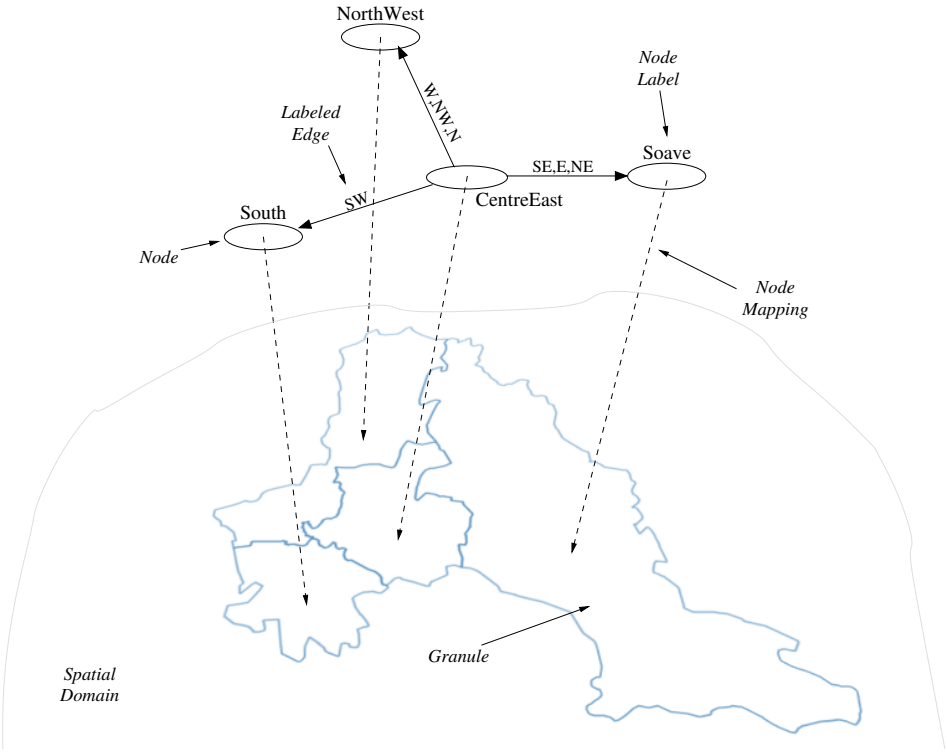


Fig. 4.1: An example of spatial granularity: the CPS catchment areas

the transitive property of the spatial relation. For each transitive relation R, the mapping $next_R : G.V \rightarrow \mathcal{P}(G.V)$, given a node v, returns the set of nodes v' reachable from v with edges labeled with R in the graph and such that there is not another edge path labeled with R from v to v'. Then, we could define an optimized graph, containing only edges considered by $next_R$.

Moreover, we define the image $G.I$ of a granularity G as the subset of $G.SD$ covered by the whole granularity. In other words, $G.I$ is the union of all granules of G, i.e. $G.I = \bigcup_{v \in G.V} G(v)$. Instead, the extent of a granularity G is the smallest convex region of $G.SD$ containing $G.I$.

A granularity G is said to be:

- *externally continuous* if the image $G.I$ is a connected region;
- *internally continuous* if each granule of G is a connected region;
- *continuous* if it is externally and internally continuous;
- *completely continuous* if G is continuous and it has no holes, that is $G.I$ is topologically equivalent to a disc;
- *total* if $G.I = SD$, i.e., the granularity covers entirely the spatial domain;
- *extent uniform* if all granules of G have the same area (or the same volume for granules in $\mathbb{R}^3$);
- *shape uniform* if all its granules have the same shape;
- *uniform* if G is uniform with respect to both the extension and shape.

4.2.3 Relationships between granularities

In this section we define the possible relations that may exist between two given spatial granularities.

Considering the spatial definition of granules, we are able to define several relationships between granularities. Some of these relationships have a weak version and a strong one. The weak version considers only how granules of the two given granularities are related, while the strong version considers also the existing spatial relations between granules.

GroupsInto(G, H): we say that a granularity G *groups weakly* into a granularity H (**GroupsInto**(G, H)) if for each node $w \in H.V$ there exists a subset $S_w \subseteq G.V$ such that $H(w) = \bigcup_{v \in S_w} G(v)$. Such definition is less restrictive of its analogous temporal definition because here there are no constraints on the relation between the granules in the subsets S_w. Conversely, in temporal framework, because of the definition of temporal granularity, granules in each group must be consecutive.

Example 4.7. The granularity representing census cells groups into the granularity representing CPS catchment areas (see the motivating scenario introduced in Section 3.2).

Moreover, we say that a granularity G *groups strongly* into a granularity H with respect to a spatial relation R, written **GroupsInto**$_R(G, H)$, if G groups weakly into H and edges in G between granules belonging to different granules of H and representing R are preserved also in H. Formally:

- G groups weakly into a granularity H;
- there exists $L_1 \in G.\Sigma_A$ such that $G.D_A(L_1) = R$ and exists $L_2 \in H.\Sigma_A$ such that
  1. $H.D_A(L_2) = R$, i.e., L_2 is the label used in H for the R relation;
  2. for each edge, labeled with L_1, $(v_1, v_2)_{L_1}$ in $G.MG$ if v_1 and v_2 belong to two different groups (i.e. they are contained in two different granules) of $H.MG$ then in $H.MG$ must exists the edge $(w_1, w_2)_{L_2}$ where w_1 and w_2 are respectively the nodes such that $v_1 \in S_{w_1}$ and $v_2 \in S_{w_2}$.

FinerThan(G, H): a granularity G is *weakly finer than* a granularity H if for each node $v \in G.V$ there exists a node $w_v \in H.V$ such that $G(v) \subseteq H(w_v)$. Moreover, we say that G is *strongly finer than* another granularity H w.r.t. a spatial relation R (**FinerThan**$_R(G, H)$), if G is weakly finer than H and edges in G between granules belonging to different granules of H and representing R are preserved in H. Formally:

- G is weakly finer than H;
- exists $L_1 \in G.\Sigma_A$ such that $G.D_A(L_1) = R$ and exists $L_2 \in H.\Sigma_A$ such that
  1. $H.D_A(L_2) = R$, i.e., L_2 is the label used in H for the R relation;
  2. for each edge $(v_1, v_2)_{L_1}$ in $G.MG$ if v_1 and v_2 are contained in two different granules of $H.MG$ then in $H.MG$ must exists the edge $(w_1, w_2)_{L_2}$ where w_1 and w_2 are respectively the nodes such that $G(v_1) \subseteq H(w_1)$ and $G(v_2) \subseteq H(w_2)$.

Example 4.8. The granularity for subzones in PCR database (see Section 3.2)) is finer than the granularity representing municipalities.

SubGranularity(G, H): G is a *weak subgranularity* of H, if for each node $v \in G.V$ there exists a node $w_v \in H.V$ such that $G(v) = H(w_v)$.
G is a *strong subgranularity* of H w.r.t. the spatial relation R, written $\mathbf{SubGranularity}_R(G, H)$, if G is a weak subgranularity of H and the edges in G representing R between granules belonging also to H are preserved in H. Formally:

- G is a weak subgranularity of H;
- exists $L_1 \in G.\Sigma_A$ such that $G.D_A(L_1) = R$ and exists $L_2 \in H.\Sigma_A$ such that
  1. $H.D_A(L_2) = R$, i.e., L_2 is the label used in H for the R relation;
  2. for each edge $(v_1, v_2)_{L_1}$ in $G.MG$ exists the edge $(w_{v_1}, w_{v_2})_{L_2}$ where w_{v_1} and w_{v_2} are respectively the nodes such that $G(v_1) = H(w_{v_1})$ and $G(v_2) = H(w_{v_2})$.

Example 4.9. The granularity representing municipalities considered in the Verona PCR (see Section 3.2) is a subgranularity (i.e., a subset) of the granularity representing all municipalities in the province of Verona.

Partition(G, H): a granularity G *partitions* another granularity H if G groups into H and G is finer than H.
We say that G *partitions strongly* H w.r.t. R, if G groups into H and G is strongly finer than H w.r.t. R. Moreover, we say that G is an m-partition of H, $\mathbf{Partition}_m(G, H)$, if G groups into H, G is weakly finer than H and each granule in H contains exactly m granules of G.

Example 4.10. Let us consider common administrative subdivisions. *Municipalities* partitions *Provinces* that, in turn, partitions *Regions*.

CoveredBy(G, H): a granularity G is *covered by* another granularity H, if the image of G is contained in the image of H, i.e., $G.I \subseteq H.I$. We note that if G is a subgranularity of H, then G is also covered by H.

Example 4.11. The granularity representing CPS catchment areas is covered by the granularity representing municipalities considered in the Verona PCR.

Disjoint(G, H): two granularities G and H are *disjoint* if their images do not intersect each other, i.e., $G.I \cap H.I = \emptyset$.

Overlap(G, H): a granularity G *overlaps* H, if its image overlaps the image of H but they do not cover each other.

Using the above relationships we can define also the following notions.

Bottom Granularity: let *g-rel* be a granularity relation (e.g., **Subgranularity**) and S be a set of granularities with the same spatial domain and representing the same relations. A granularity $G \in S$ is a bottom granularity with respect to *g-rel* if for each granularity $H \in S$, *g-rel*(G, H).

Granularity Lattice: a set of spatial granularities with the same spatial domain and representing the same relations is a lattice w.r.t. a granularity relation *g-rel*, if for each pair of granularities in the set there exists a least upper bound and a greatest lower bound with respect to *g-rel* (only partial order relations, i.e. reflexive, antisymmetric, and transitive relations, can define a lattice).

Coverage: a coverage is a set of granularities having the same domain and a bottom granularity with respect to **GroupsInto** relation. For example, the usual hierarchical administrative divisions used by nations (`Municipalities`, `Provinces`, `Regions`) is a coverage.

4.2.4 Operations over granularities

In this section we present some useful operations in order to manage and create granularities using the already defined ones. We identify three classes of operations:

- grouping-oriented operations;
- granule-oriented operations;
- set operations.

Grouping-oriented operations

The *grouping-oriented operations* generate granules of a new granularity combining together some granules of one or two given granularities.

Grouping(G, P, L): we define an user-driven grouping operation. Let G be a granularity, $P = \{Q_1, Q_2, \ldots, Q_n\}$ be a set of pairwise disjoint node sets containing nodes of G, and $L = \{l_1, l_2, \ldots, l_n\}$ be a label set such that $|P| = |L|$. **Grouping**(G, P, L) produces the granularity G' with the same spatial domain of G, the same edge labels of G, $G'.D_A = G.D_A$ and where each granule g' of G' is the spatial union of the granules of the granules in $Q_i \in P$. Since nodes have been changed, in G' edges must be recomputed.

Example 4.12. Considering the municipalities in the Verona PCR, the **Grouping** operation allows epidemiologists to calculate the spatial granularity `DayNoise` representing municipalities with the same level of day noise pollution. Thus, in this case the starting granularity is `Municipalities` (see Fig. 4.2), each set in P contains the label of the municipalities with the same noise level, and L contains the new labels: "low", "moderate", "intermediate", "high", and "veryHigh". The resulting granularity, `DayNoise`, will contain five granules, each one representing municipalities with the same noise pollution level. The `DayNoise` granularity is depicted in Fig. 4.3, where granules are identified by different colors. □

Combining(G_1, G_2): this operation creates a new granularity G', from two given granularities G_1 e G_2, combining into one granule of G' all granules of G_2 covered by a single granule of G_1. In other words, it groups together granules of G_2 according to the partition of DS induced by G_1. Formally, **Combine**(G_1, G_2) is the granularity G' created with the following rule. For each node k in $G_1.V$ we define $s(k) = \{h \in G_2.V \mid G_2(h) \subseteq G_1(k)\}$ then G' has a node k' for each

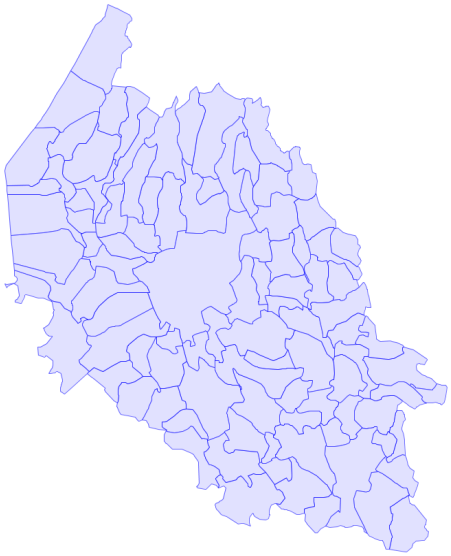


Fig. 4.2: The granularity representing municipalities in the province of Verona.

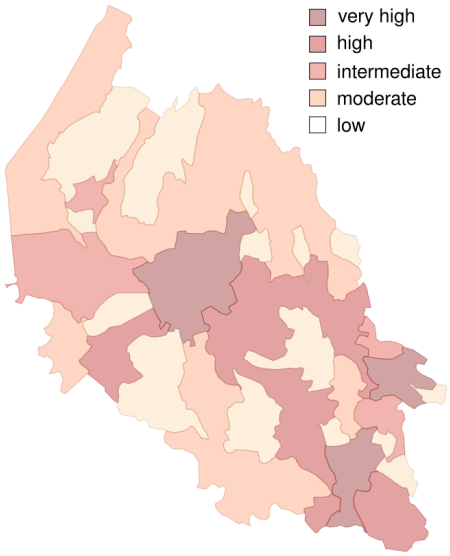


Fig. 4.3: The granularity representing municipalities grouped with respect to day noise pollution.

non-empty set $s(k)$ and: $G'(k') = \bigcup_{j \in s(k)} G_2(j)$. G' is defined with the same spatial domain, edge labels and relation definitions of G_1.

Granule-oriented operations

Granule-oriented operations build new granularities selecting only some granules of the input granularities.

Subset: we define two variants of **Subset**. The first version, **Subset**(G, S), given a granularity G and a subset S of granules of G, creates G' defined exactly as G but only with the specified granules.
We define also a second version of **Subset**. Given a granularity G, a node $v \in G.V$ and an edge label R, **Subset**(v, G, R) generates a new granularity G' with the same components of G but only with the nodes reachable from v with edges labeled with R.

Example 4.13. Let us consider the `DayNoise` granularity we calculated before in this section by using the *Grouping* operation and partitioning the municipalities in the province of Verona with respect to day noise pollution. **Subset**(`DayNoise`, $\{high, veryHigh\}$) returns a new granularity, `HighDayNoise` (see Fig. 4.4), containing only the granules of `DayNoise` that represent areas with high or very high day noise pollution. □

SelectContain(G_1, G_2): it creates a new granularity G' selecting only granules of G_1 containing at least one granule of G_2. We calculate the new node set as

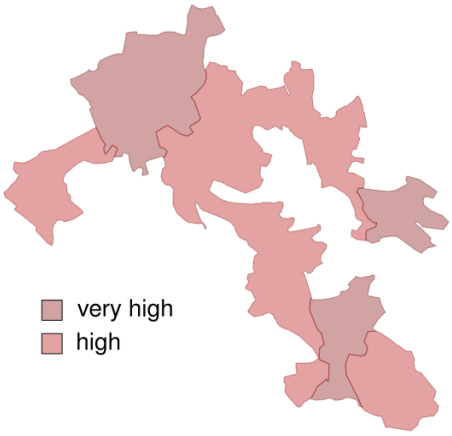


Fig. 4.4: The granularity representing areas with high and very high noise pollution.

Fig. 4.5: The granularity representing municipalities that are inside areas with high day noise pollution.

$$G'.V = \{j \in G_1.V \mid \exists i \in G_2.V : G_2(i) \subseteq G_1(j)\}$$

then, for each node k in $G'.V$, $G'(k) = G_1(k)$. The other components of G' are the same of G_1.

SelectInside(G_1, G_2): it create G' selecting only the granules of G_1 included in one granule of G_2. We calculate the new node set as

$$G'.V = \{j \in G_1.V \mid \exists i \in G_2.V : G_1(j) \subseteq G_2(i)\}$$

then for each node k in $G'.V$, $G'(k) = G_1(k)$. The other components of G' are the same of G_1.

Example 4.14. Epidemiologists may create a new granularity for representing only those municipalities included in areas with high or very high day noise pollution. The **SelectInside**(`Municipalities`, `HighDayNoise`) operation creates this new granularity, `MunHighDayNoise` (see Fig. 4.5). □

SelectIntersect(G_1, G_2): $G' =$ **SelectIntersect**(G_1, G_2) is generated selecting only the granules of G_1 intersecting at least one granule of G_2.

$$G'.V = \{j \in G_1.V \mid \exists i \in G_2.V : G_1(j) \cap G_2(i) \neq \emptyset\}$$

then, for each node k in $G'.V$, $G'(k) = G_1(k)$. The other components of G' are the same of G_1.

Set operations

Set operations extend the usual set operators to spatial granularities.

Fig. 4.6: The granularity representing areas with low noise pollution.

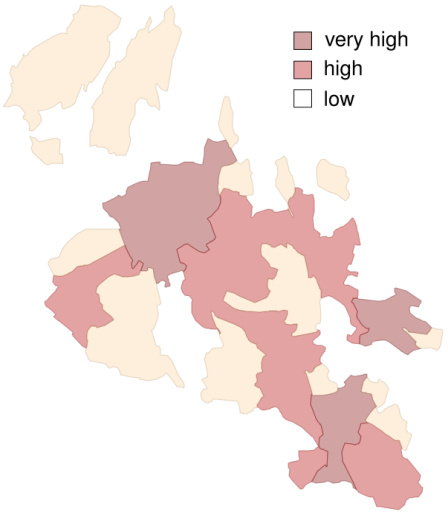


Fig. 4.7: The granularity representing both municipalities with low and high day noise pollution.

Union: $G_1 \cup G_2$ is a new granularity G' with all the granules of G_1 and with the granules, or their parts, of G_2 that do not intersect the granules of G_1. To calculate the union of G_1 and G_2, we start with G' equals to G_1 in all its components. Next, for each node v in $G_2.V$, if $G_2(v) \setminus G_1.I \neq \emptyset$ then we add to G' a new node, labeled with $G_2.l_V(v)$ (eventually, we have to change these labels, e.g., adding them a prefix, in order to avoid duplicated labels with those used by G_1), associated to the spatial granule $G_2(v) \setminus G_1.I$. Note that if a node $v \in G_2.V$ is completely covered by the image of G_1 (i.e., $G_2(v) \setminus G_1.I = \emptyset$) then we do not store the node in G' because its corresponding granule would be empty. In G' we represent spatial relations contained either in G_1 and G_2. We notice that this definition of the union operator is not commutative as the corresponding set operation.

Example 4.15. Let `LowDayNoise` (see Fig. 4.6) and `HighDayNoise` (see Fig. 4.4) the granularities representing the municipalities with low and high or very high noise pollution levels, respectively. Thus,

$$\texttt{LowHighDayNoise} = \texttt{LowDayNoise} \cup \texttt{HighDayNoise}$$

is the granularity representing both municipalities with low and high day noise pollution (see Fig. 4.7). □

Intersect: intersecting two granularities G and H we maintain only the intersection of their granules. Thus, $G' = G_1 \cap G_2$ is a new granularity having the intersection of $G_1.DS$ and $G_2.DS$ as spatial domain. Moreover, G' has a node for each non-empty intersection between one granule of G_1 and one granule of G_2. The new nodes in G' are labeled with the concatenation of the labels of

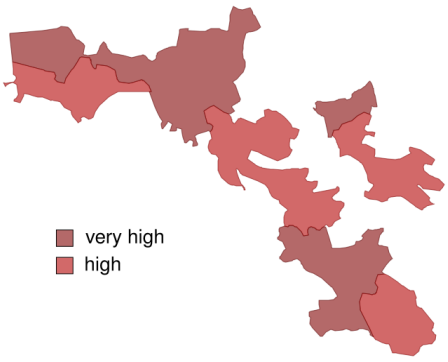


Fig. 4.8: The granularity representing areas with high night noise pollution.

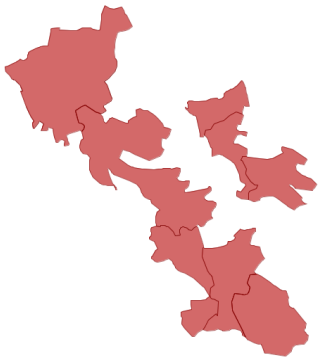


Fig. 4.9: The granularity representing areas with high noise pollution both in the daytime and in the night.

Fig. 4.10: The granularity representing areas with high noise pollution in the daytime but not in the night.

the two nodes whose intersection defines the node. In G' we represent spatial relations that are represented either in G_1 and G_2.

Example 4.16. Let `HighDayNoise` (see Fig. 4.4) and `HighNightNoise` (see Fig. 4.8) be the granularities representing areas with high day noise pollution and high night noise pollution, respectively.

$$\texttt{HighNoise} = \texttt{HighDayNoise} \cap \texttt{HighNightNoise}$$

is the granularity (depicted in Fig. 4.9) representing areas with high noise pollution both in the daytime and in the night. □

Difference: the granules of $G' = G_1 \setminus G_2$ are the same of G_1 minus the area covered by some granule of G_2. We start with $G' = G_1$, then for each $v \in V'$ we modify $G'(v)$ removing the points contained also in $G_2.I$; if we obtain an empty set we remove the node from G'. Relations represented in G' are the same of G.

Example 4.17. Let `HighDayNoise` and `HighNightNoise` be the granularities representing areas with high day noise pollution and high night noise pollution,

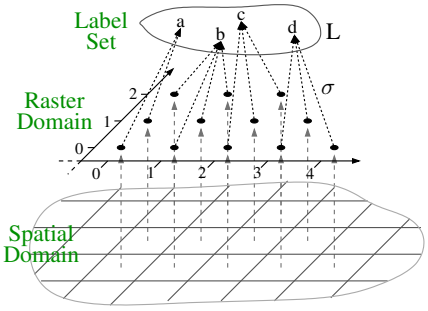


Fig. 4.11: Structure of a raster spatial granularity

respectively. `HighOnlyDayNoise` = `HighDayNoise` ⧵ `HighNightNoise` is the granularity representing areas with high noise pollution in the daytime but not in the night. The `HighOnlyDayNoise` granularity is depicted in Fig. 4.10 with the same scale of previous pictures. □

4.3 Raster-based spatial granularities

In this section, we define a notion of spatial granularity for the raster data model. Moreover, based on this notion, we redefine relationships and operations already defined in the previous section for vector-based spatial granularities. As we discussed in Section 2.2.3, the raster model is model, alternative to the vector one, for representing spatial information. Raster maps are more suitable for environmental applications, involving continuous spaces. They are used for examples in all those applications where data are measured by remote sensors (e.g., satellites and cameras). Usual raster data include raster thematic maps representing demographic, economic, social, and environmental dimensions, and satellite imagery in which surveys are always stored in a raster format.

4.3.1 Spatial granularities

In the *raster* model, maps are represented by partitioning the space domain in equal-sized square areas (see Fig. 4.11). The size of areas depends on the resolution (i.e., the accuracy needed) of the map: it may range from centimeters to kilometers. Inside each area, the physical characteristic of interest is measured and a corresponding value is associated to the area. Areas can be uniquely and totally numbered starting from a specific point, the origin of the raster map. Usually, areas are numbered defining a Cartesian coordinate system: this way, each area corresponds to a unique pair of integers. Thus, a raster map can be represented by a matrix whose components are called *cells*. Knowing the position in the space domain of the origin of the map and its resolution (i.e., the size of areas), it is possible to know what area corresponds to each cell. Hereafter, we suppose that all maps have the same coordinates system and resolution. Each cell stores the value of the corresponding area. We do not consider the construction of raster maps, but only the mapping associating to each cell the corresponding label.

A spatial granularity for raster maps represents the partitioning of cells, and then of areas in the space domain, accordingly to their associated values, called *labels*. Hence, we formally define a spatial granularity σ as a total function from two-dimensional coordinates in $\mathbb{Z}^2$ to a label set L, $\sigma : \mathbb{Z}^2 \to L$. In this way, given a cell $c \in \mathbb{Z}^2$, $\sigma(c)$ represents the label associated to c. The same model has been used also by Erwig et al. [78]. Note that the label set can be of any type, e.g., integer numbers, pairs, or strings. To ensure that σ is a total function, we assume that each label set L contains the special label $\perp$ (called *undefined*) and that cells whose areas are not covered by σ are all labeled with $\perp$. We define the image of a spatial granularity as the set of all cells with a non-undefined label, i.e., $\mathbf{Image}(\sigma) = \{c \in \mathbb{Z}^2 \mid \sigma(c) \neq \perp\}$.

Granules composing a granularity are the sets of all cells with the same label. Since each cell belongs exactly to one granule and areas corresponding to cells are disjoint by construction, also granules are disjoint without imposing any further constraint. Given a granularity σ, the granule with label $l \in L$, denoted with $\gamma_\sigma(l)$, is then represented by all cells labeled with l, $\gamma_\sigma(l) = \{c \in \mathbb{Z}^2 \mid \sigma(c) = l\}$. Moreover, given a cell $c \in \mathbb{Z}^2$, we denote with $\gamma_\sigma(c) = \gamma_\sigma(\sigma(c))$ the granule to which c belongs. We define the set of all granules composing σ as $\Gamma_\sigma = \{g \in \mathcal{P}(\mathbb{Z}^2) \mid g = \gamma_\sigma(l) \wedge l \in \mathbf{range}(\sigma)\}$ where $\mathbf{range}(\sigma)$ is the set of all labels actually used in σ.

Despite the different underlying structure, and thus the different formalism, granularities for raster and vector models represent the same notions and thus have similar requirements. Indeed, in both cases, granules are required to be disjoint. Hence, as it is possible to transform (eventually with some approximations) vector data in raster data (rasterization) and vice versa (vectorization) [94], it may be possible to transform a spatial granularity based on vector data into a granularity based on raster data. A basic approach for obtaining a vector representation of a raster map could be to define vector geometries defining the boundary of raster granules. Then, a spatial granularity can be defined on the obtained geometries, where the label of each granule could be the label of the original raster granule. The opposite process (i.e., the rasterization) is based on the usual partition of the space domain in cells and the substitution of each vector geometry with the set of cells intersecting it.

Fig. 4.12 depicts an example of raster spatial granularity representing land usage. For the sake of simplicity, we represented each label with a different color, explained beside. In this example, the granularity is made up of three granules representing areas covered by commercial, recreational, and residential buildings.

Note that the above definition of spatial granularity is also suitable for representing granularities that are regular subdivisions of the space, as in the example depicted in Fig. 4.13. For example, knowing that cells (squares with thinner borders) represent square areas in the space domain whose size is 10×10 meters, we would like to make granules (squares with thicker borders) representing sets of cells whose total extent in the space domain is 20×20 meters. In this case, cells are grouped together in bigger squares. The number of cells, on x-axis and y-axis, to be grouped depends by the ratio between the map resolution (10×10) and new granule size (20×20), in this case it is two. To each group (i.e., granule) must

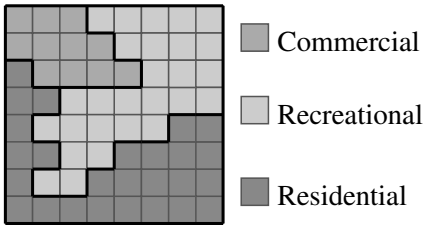


Fig. 4.12: Example of a raster spatial granularity

10	10	11	11	14	14	15	15
10	10	11	11	14	14	15	15
8	8	9	9	12	12	13	13
8	8	9	9	12	12	13	13
2	2	3	3	6	6	7	7
2	2	3	3	6	6	7	7
0	0	1	1	4	4	5	5
0	0	1	1	4	4	5	5

Fig. 4.13: A raster regular spatial granularity

be associated a different label. Labels can be calculated, for example, by using space-filling curves, e.g., Z-order (exemplified in the figure).

4.3.2 Relationships

Given two spatial granularities σ_1 and σ_2, we define the following *relationships* between them, whose meaning is equivalent to that of relationships between vector-based spatial granularities given in Section 4.2.3.

- The **GroupsInto** relationship requires that each granule in σ_2 is equal to the union of a set of granules in σ_1.
  $\mathbf{GroupsInto}(\sigma_1, \sigma_2) \triangleq \forall g \in \Gamma_{\sigma_2}.\exists G \subseteq \Gamma_{\sigma_1}.g = \bigcup_{h \in G} h$.
- The **FinerThan** relationship requires that each granule in σ_1 is contained in a granule of σ_2:
  $\mathbf{FinerThan}(\sigma_1, \sigma_2) \triangleq \forall g_1 \in \Gamma_{\sigma_1}.\exists g_2 \in \Gamma_{\sigma_2}.g_1 \subseteq g_2$.
- The **Partition** relationship corresponds to impose both the **GroupsInto** and the **FinerThan** relations:
  $\mathbf{Partition}(\sigma_1, \sigma_2) \triangleq \mathbf{GroupsInto}(\sigma_1, \sigma_2) \wedge \mathbf{FinerThan}(\sigma_1, \sigma_2)$.
- The **Subgranularity** relation requires that each granule in σ_1 has a correspondent equal granule in σ_2:
  $\mathbf{Subgranularity}(\sigma_1, \sigma_2) \triangleq \forall g_1 \in \Gamma_{\sigma_1}.\exists g_2 \in \Gamma_{\sigma_2}.g_1 = g_2$.
- The **CoveredBy** relationship requires that the image of σ_1 is covered (i.e., is a subset) of that of σ_2 :
  $\mathbf{CoveredBy}(\sigma_1, \sigma_2) \triangleq \mathbf{Image}(\sigma_1) \subseteq \mathbf{Image}(\sigma_2)$.
- The **Disjoint** relationship imposes that the images of σ_1 and σ_2 be disjoint:
  $\mathbf{Disjoint}(\sigma_1, \sigma_2) \triangleq \mathbf{Image}(\sigma_1) \cap \mathbf{Image}(\sigma_2) = \emptyset$.
- Finally, the **Overlap** relationship requires that the images of σ_1 and σ_2 have a non-empty intersection:
  $\mathbf{Overlap}(\sigma_1, \sigma_2) \triangleq \mathbf{Image}(\sigma_1) \cap \mathbf{Image}(\sigma_2) \neq \emptyset$.

The comparison of two different granularities with these relationships is meaningful only if they are based on the same coordinate system. The same thing is true for operations.

4.3.3 Operations

We can redefine also the operations given for vector-based spatial granularities. In the following definitions we assume that $\sigma : \mathbb{Z}^2 \to L$, $\sigma_1 : \mathbb{Z}^2 \to L_1$, and $\sigma_2 : \mathbb{Z}^2 \to L_2$ are raster-based spatial granularities. The meaning of each operation is the same as we explained in the previous section about vector-based spatial granularities.

- The **Grouping** operation creates σ' in which granules of σ are grouped together with respect to a partition of its labels. Formally, if $P = \{Q_1, \ldots, Q_n\}$ is a partition on the label set L such that $\bigcup_{i=1}^{n} Q_i = L \setminus \{\bot\}$, then **Grouping**$(\sigma, P)$ is defined as the mapping $\sigma' = \mathbb{Z}^2 \to \mathcal{P}(L)$ such that:
  $\forall c \in \mathbb{Z}^2.\sigma'(c) = Q_i$ iff $\sigma(c) \in Q_i$.
- The **Combine** operation groups together the granules of a given granularity σ_2 to form a new granule. Conversely to **Grouping**, in this case the groups are given by the partition of the space defined by the granularity σ_1. All granules of σ_2 completely contained in a granule of σ_1 are grouped together, and they receive the label of the granule in σ_1. Formally, $\sigma' =$ **Combine**$(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1$ is such that
  $$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \sigma_1(c) & \text{if } \gamma_{\sigma_2}(c) \subseteq \gamma_{\sigma_1}(c) \wedge \sigma_2(c) \neq \bot \\ \bot & \text{otherwise.} \end{cases}$$
- The **Subset** operation returns a new granularity σ_1 in which only cells having their label in a given label set I are maintained, while the other ones are set to $\bot$. Formally, if $I \subseteq L$ then $\sigma' =$ **Subset**$(\sigma, I) : \mathbb{Z}^2 \to I$ is defined as
  $$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \sigma(c) & \text{if } \sigma(c) \in I \\ \bot & \text{otherwise.} \end{cases}$$
- The **SelectContain** operation sets to $\bot$ all cells of σ_1 whose granules do not contain at least one granule of σ_2. Formally, $\sigma' =$ **SelectContain**$(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1$ is defined such that
  $$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \sigma_1(c) & \text{if } \exists l_2 \in L_2 \setminus \{\bot\}.\gamma_{\sigma_2}(l_2) \subseteq \gamma_{\sigma_1}(c) \\ \bot & \text{otherwise.} \end{cases}$$
- The **SelectInside** operation keeps only those granules of σ_1 that are contained (i.e., inside) in a granule of σ_2. Formally, $\sigma' =$ **SelectInside**$(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1$ is such that
  $$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \sigma_1(c) & \text{if } \gamma_{\sigma_1}(c) \subseteq \gamma_{\sigma_2}(c) \wedge \sigma_2(c) \neq \bot \\ \bot & \text{otherwise.} \end{cases}$$
- The **SelectIntersect** operation keeps only those granules of σ_1 that intersect at least one granule of σ_2. Formally, $\sigma' =$ **SelectIntersect**$(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1$ is defined as:
  $$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \sigma_1(c) & \text{if } \exists l_2 \in L_2 \setminus \{\bot\}.\gamma_{\sigma_2}(l_2) \cap \gamma_{\sigma_1}(c) \neq \emptyset \\ \bot & \text{otherwise.} \end{cases}$$
- The **Union** operation calculates the union of two granularities σ_1 and σ_2 by taking all granules of σ_1 and also those parts of the granules of σ_2 that do not intersect any granule of σ_1. Formally, $\sigma' =$ **Union**$(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1 \cup L_2$ is defined as the mapping such that:
  $$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \sigma_1(c) & \text{if } \sigma_1(c) \neq \bot \\ \sigma_2(c) & \text{otherwise.} \end{cases}$$
- We can define two versions of the intersection of two granularities, that we call inner intersection and outer intersection. The first one considers only cells

that have an associated label in both parameter granularities. It is defined as $\sigma' = \textbf{InnerIntersect}(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1 \times L_2$ such that

$$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \langle \sigma_1(c), \sigma_2(c) \rangle & \text{if } \sigma_1(c) \neq \perp \wedge \sigma_2(c) \neq \perp \\ \perp & \text{otherwise.} \end{cases}$$

where $\langle \sigma_1(c), \sigma_2(c) \rangle$ represent the concatenation of the two original labels: $\sigma_1(c)$ and $\sigma_2(c)$.

The second one considers also cells that have not an associated label in one of the two parameter granularities, i.e., $\sigma' = \textbf{OuterIntersect}(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1 \times L_2$ is such that

$$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \perp & \text{if } \sigma_1(c) = \perp \wedge \sigma_2(c) = \perp \\ \langle \sigma_1(c), \sigma_2(c) \rangle & \text{otherwise.} \end{cases}$$

- The **Difference** operation keeps the label of only those cells of σ_1 that are undefined in σ_2. Formally, $\sigma' = \textbf{Difference}(\sigma_1, \sigma_2) : \mathbb{Z}^2 \to L_1$ is such that:

$$\forall c \in \mathbb{Z}^2.\sigma'(c) = \begin{cases} \sigma_1(c) & \text{if } \sigma_2(c) = \perp \\ \perp & \text{otherwise.} \end{cases}$$

We can also adapt to our definition of spatial granularity the **Relabel** operation proposed by Erwig and Schneider [78]. This operation allows one to modify the labels of a granularity. To do that, it uses a total function r that associates to each label of the given granularity a new label. Then, if I is a label set and $r : L \to I$ is a relabeling total function, $\sigma' = \textbf{Relabel}(\sigma, r) : \mathbb{Z}^2 \to I$ is defined such that $\forall c \in \mathbb{Z}^2.\sigma'(c) = r(\sigma(c))$.

We note that the **Grouping** and **Subset** operations can be expressed by using the **Relabel** operation. Given a partition $P = \{Q_1, \ldots, Q_n\}$ of the label set of σ (say L), we can define the relabeling function $r_P : L \to \mathcal{P}(L)$ such that, for each label $l \in L$, $r_P(l) = Q_i$ if and only if $l \in Q_i$. Then, $\textbf{Grouping}(\sigma, P) \equiv \textbf{Relabel}(\sigma, r_P)$.

On the other hand, given a subset I of the label set of σ (say L) we can define the relabeling function $r_I : L \to I$ such that, for each label $l \in L$, $r_I(l) = l$ if $l \in I$, and $r_I(l) = \perp$ otherwise. Then, $\textbf{Subset}(\sigma, I) \equiv \textbf{Relabel}(\sigma, r_I)$.

5

A framework for spatio-temporal granularity

In the previous chapter we defined the notion of spatial granularity. Now we join together our notion of spatial granularity with that of temporal granularity [23] for obtaining a model for spatio-temporal granularities.

Informally a spatio-temporal granularity represents a spatial granularity that changes over time. Any change to the spatial aspect we are modeling (e.g., provinces) modifies the spatial granularity defining a new version of the same granularity. For example, we may consider Italian municipalities since year 1861 (when the Kingdom of Italy was proclaimed). Over these years, municipalities evolved: some municipalities have been suppressed, some new ones have been declared, some municipalities have been merged, some others have been split. Thus, the partition of Italian territory has been changed many time over the years. We can use a spatio-temporal granularity to represent this evolution. In other words, it is possible to imagine a spatio-temporal granularity as a slide show where each slide corresponds to a time instant and represents a spatial granularity.

Some proposals [37, 214] in the literature define a spatio-temporal granularity as a mapping that associates to each temporal granule the spatial granularity valid in that temporal granule. Thus, they associate a spatial granularity to each temporal granule and it remains constant up to the next temporal granule that will have eventually associated another spatial granularity. However, we think that this approach limits the expressiveness. We propose to associate a spatial granularity to each time instant and to use temporal granules for grouping together time instants, and thus spatial granularities. In our approach, we allow to the spatial information to change during a single temporal granule. This approach is more general than other proposed in literature. As a matter of facts, our approach can represent also the other proposed approaches in which just one spatial granularity is associated to each temporal granule.

In a spatio-temporal granularity several kinds of change may happen between two subsequent versions of a spatial granularity. Some granules could modify their position and extent, some granules could be deleted, some granules could be split, finally other granules could be merged together. Our spatial-temporal granularities keep track of these modifications maintaining also information about the history of single granules. Hence, it is possible, for example, to know that a given granule is derived at a certain instant from the merging of two other ones.

In this chapter we present our framework for spatio-temporal granularities. The framework has been instantiated both on vector data model and on raster data model. In both cases, the framework includes, besides the spatio-temporal granularity definition, the definitions of relationships and operations over granularities. The chapter is organized as follow. In Section 5.1 we discuss the main proposals in the literature about the definition of spatio-temporal granularities and related notions. In Section 5.2 we introduce our framework for spatio-temporal granularities based on vector data, including relationships and operations. Finally, in Section 5.3 we present our framework for spatio-temporal granularities based on the raster data model.

In the following, we distinguish spatio-temporal, temporal, and spatial granularities adding to their name the prefixes *st*, *t*, and *s*, respectively.

5.1 Related work

As said earlier, there are only few proposals about spatio-temporal granularity. Some of these use term "multi-granularity" to represent "multi-resolution" [29,81]. Since this work focuses on granularities, we do not describe proposals about multi-resolution systems.

In [214] Wang and Liu define a spatio-temporal granularity as a pair composed by a temporal and a spatial granularity. Formally, if SG is a spatial granularity and TG a temporal granularity then $STG = SG \otimes TG$ is a spatio-temporal granularity. $STG(i, j)$ represents a spatio-temporal granule composed by the ith spatial granule of SG and the jth temporal granule of TG. This means that at instants belonging to $TG(j)$ the spatial granule $SG(i)$ is valid (i.e., exists). We note that using this approach they just attach the valid time (i.e., the time period during which a fact is valid in the reality) to granules of a spatial granularity. Hence when a change in the spatial granularity SG happens they must define a new spatio-temporal granularity to represent the new data. On the other hand, this approach allows one to represent spatial granules with different valid times, i.e., it works on single spatial granules rather than on the whole spatial granularity. So, for example, we can represent a spatio-temporal granularity based on `Years` (to represent its temporal part) and on `NaturalParks` (to represent its spatial part) where `NaturalParks`*(Stelvio)* is valid in `Years`(2007) but not in `Years`(2015) while `NaturalParks`*(VeronaWalls)* is not valid in `Years`(2007) but it is in `Years`(2015) even if parks are represented in the same spatial granularity.

In [34,35,37] Camossi et al. present a similar approach about spatio-temporal granularities. Conversely to Wang et al., Camossi et al. attach valid time to spatial granularities, not to single granules. A spatio-temporal granularity is defined as a partial function that associates a spatial granularity to those temporal granules representing instants during which it is valid.

They represent spatio-temporal information extending ODMG standard [38, 155]. For this purpose they define two new parametric datatypes. Temporal information are defined using the type $\mathcal{T} = Temporal < G_t, \tau >$ where G_t is a temporal granularity (e.g., `Seconds`, `Days`, `Years`) and τ, called inner type, represents the data associated to G_t. If τ is a basic type then $\mathcal{T}$ represents

a temporal data type, otherwise τ may be a spatial data type and $\mathcal{T}$ represents spatio-temporal data. In this case, Camossi et al. define the parametric type $Spatial < G_s, \gamma >$ where G_s is a spatial granularity (e.g., m, km, mq, kmq) and γ describes the geometric extent of the spatial data. For example, $Temporal < \texttt{Decades}, Spatial < \texttt{km}, set < Region >>>$ represents a spatio-temporal granularity whose spatial extent is represented by a set of regions (each one representing a spatial granule) expressed by using kilometers and updated at most once a decade.

In [30] Bittner proposed the notion of stratified spatio-temporal map spaces, a spatio-temporal extension of stratified map spaces [197]. The author defines a spatio-temporal granularity as a pair composed by a temporal and a spatial granularity. A spatio-temporal location is defined as a pair made up of one temporal and one spatial granule. In a map, objects are associated to spatio-temporal locations containing them, thus objects are provided with a temporal and a spatial location. Based on this notion, Bittner introduces a set of four spatio-temporal relationships (i.e., same-time-same-place, different-time-same-place, same-time-same-place, different-time-different-place) between two spatio-temporal locations.

In [80] Erwig and Schneider extend their previous work about spatial partitions [78], defining the notion of spatio-temporal partition or temporal map. Spatio-temporal partitions represent spatial partitions evolving over time. In particular, similarly to our approach, a spatio-temporal partition associates a spatial partition to each time point. Authors propose also a set of operations over spatio-temporal partitions:

- overlay, i.e., intersection of two temporal maps;
- clipping, i.e., the restriction of a temporal map to a given spatial region;
- reclassification, i.e., the relabeling of granules in a temporal map;
- fusion, i.e., the merging of neighboring granules with the same label;
- domain, i.e., the calculation of all time points where a non-empty spatial partition is defined;
- temporal restriction, i.e., the restriction of a temporal map to some given temporal intervals;
- temporal selection, i.e., the checking for each time point whether a specified predicate holds or not;
- temporal aggregation, i.e., the combination into one label of all labels associated to a spatial point over time;
- temporal projection, i.e., the temporal aggregation applied to all points in a temporal map.

Considering raster spatio-temporal maps, the model for raster spatial partition proposed in [204] and described in Section 4.1 has been extended to the spatio-temporal case by Mennis et al. [146]. They extend the usual two-dimensional space adding time as third dimension. Thus, they represent spatio-temporal maps as mappings from a three-dimensional matrix (where a dimension represents time while the other two ones represent space) to a set of labels. Each entry in the three-dimensional matrix represents a spatio-temporal location. Labels represent the measured physical characteristic associated to spatio-temporal locations, i.e., a spatial location in a time point. Using this model, they extend also the map

algebra functions [204] (see Section 4.1) defining the three-dimensional extension of local, focal, and zonal operations [67]. Their definitions are the same of those for two-dimensional map algebra with the only difference that locations that now zones and neighborhoods have cubic forms.

5.2 Vector-based spatio-temporal granularities

In our proposal, a spatio-temporal granularity represents and traces the changes in the time of a particular spatial granularity. For example, if we consider the Italian provinces, they were not always the same, they changed several times during the last century. Thus, in the spatio-temporal granularity representing provinces, we can keep the information about all the history of Italian provinces.

Similarly to the framework for spatial granularities, also the framework for spatio-temporal granularities includes the definition of a spatio-temporal granularity, and the definitions of relationships and operations over spatio-temporal granularities. In this section, we present our framework for spatio-temporal granularities.

5.2.1 Spatio-temporal granularities

Formally, the notion of spatio-temporal granularity is based on the notion of spatial evolution that really associates to each point the valid spatial granularity.

Definition 5.1 (Spatial evolution). *Let T be a (possibly infinite) temporal domain and $GF = \{sG_k\}_k$ a set of spatial granularities with the same edge label set and representing the same spatial relations. A* spatial evolution *E is a mapping from T to GF such that:*

$$\exists t_1, t_2 \in T, t_1 \leq t_2 : \forall t \in [t_1, t_2] : \exists sG_k \in GF : E(t) = sG_k$$

i.e., given a temporal point t between a lower t_1 and upper bound t_2, $E(t)$ provides the spatial granularity valid at t.

Given a spatial node label j, $E(t)(j)$ represents the spatial granule j valid at point t.

Lower and upper bounds in the evolution definition allow one to represent finite evolutions. However, we can represent infinite evolutions when lower and upper bounds are infinite and, thus, $E(t)$ provides the spatial granularity for an infinite number of temporal instants . Hence, a spatio-temporal granularity is defined as follow.

Definition 5.2 (Spatio-temporal granularity). *Let tG be a temporal granularity and E a spatial evolution, both over domain T. A* spatio-temporal granularity *stG is a pair $\langle tG, E \rangle$. Moreover, $stG(i, j) = \{E(t)(j)\}_{t \in tG(i)}$ is the* spatio-temporal granule *representing the evolution of spatial granule j during the temporal granule i.*

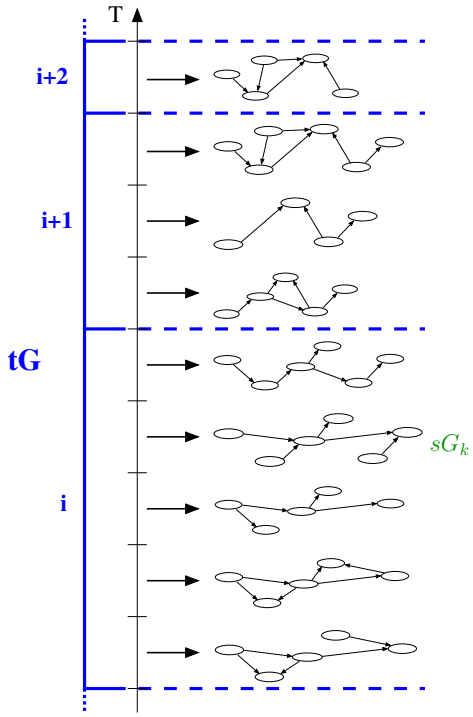


Fig. 5.1: The structure of a spatio-temporal granularity.

Conversely to other approaches in literature [34,214], in our definition of spatio-temporal granularity we do not associate a single spatial granularity to a time granule, but we associate to each granule a sequence of spatial granularities, one for each instant in the time granule. This approach overcomes limitations of other proposals in literature. Those proposals limit the comparison and the conversion of information only to spatio-temporal granules belonging to granularities based on temporal granularities related by $FinerThan$ or $GroupsInto$ relationships. Our approach overcomes this constraint. Maintaining always information at time point level, we can compare any couple of granularities without loss of information.

Fig. 5.1 shows the structure of a spatio-temporal granularity. The vertical axis represents the time line where a spatial granularity is associated to each time point. Time points are grouped with respect to the time granules of the time granularity tG on which the spatio-temporal granularity is based.

To maintain the history of single spatial granules in spatio-temporal granularities we introduce the mapping $nextTime$.

Definition 5.3 (nextTime). *Given a spatio-temporal granularity* $stG = \langle tG, E \rangle$, $nextTime_{stG} : T \times SG \to \mathcal{P}(SG)$ *(*SG *is the set of all possible spatial granules of the evolution of* stG*) is a mapping that, for each instant* t *in the image of* tG *and given a spatial granule* j *belonging to the spatial granularity valid at time* t *(i.e.,* $j \in E(t).V$*), returns the set of nodes* NT *representing the granule* j *at time* $t+1$*. Considering* NT*, there are three cases:*

1. $NT = \emptyset$: *the granule* j *does not exist any more;*
2. $NT = \{j'\}$: *if* $j' = j$ *then the granule still exists and it has not been split; if* $j' \neq j$ *then the granule has been renamed or merged with other granules;*
3. $NT = \{j_1, j_2, \ldots, j_n\}$: *the granule has been split in* n *new granules.*

We say that a merge of several granules $j_1, j_2, \ldots, j_n$ *at time* t *in one granule* j *at instant* $t+1$ *has happened if* $nextTime(t, j_k) = \{j\}$ *for each* $1 \leq k \leq n$.

5.2.2 Relationships between granularities

Similarly to temporal and spatial granularities, also for spatio-temporal granularities it is possible to define some relations and operations useful to manage and reason about them.

In particular, relations are useful in order to perform spatio-temporal reasoning (e.g., it is possible to convert information expressed by using a granularity into an equivalent information represented by another granularity).

Spatial definitions have been extended to the spatio-temporal context by adding a time domain. The idea is to check whether a given spatial relationship is valid between spatial granularities associated at some time points in two given spatio-temporal granularities. Thus, each spatio-temporal relationship wraps a spatial relationship and check whether this spatial relationship is valid between spatial granularities in the two given spatio-temporal granularities. Moreover, spatio-temporal relationships include also two temporal quantifications, one at the granule level and another at the time point level. Quantifications allow one to specify when spatial relationships must be valid during the spatial evolution in order to consider satisfied a spatio-temporal relation. The "always" ($\forall$) and "sometimes" ($\exists$) temporal quantifiers are used. The quantification at the granule level allows one to check whether the relationship is valid at time points in "all" temporal granules or only in "some" granules (i.e., at least one). The second quantifier, that always follows and is subordinate to the first one, allows one to check whether the relationship is valid at "all" or "some" (i.e., at least one) time points in considered temporal granules. Thus, there are four possible checks that can be specified by the user:

- $\forall\forall$ checks whether the relationships is valid at each time point in each time granule of the first given spatio-temporal granularity;
- $\forall\exists$ checks whether the relationships is valid in at least one time point in each time granule of the first given spatio-temporal granularity;
- $\exists\forall$ checks whether the relationships is valid at each time point in at least one time granule of the first given spatio-temporal granularity;
- $\exists\exists$ checks whether the relationships is valid in at least one time point in at least one time granule of the first given spatio-temporal granularity;

The process for checking a spatio-temporal relationship check the spatial relationship between spatial granularities associated to spatio-temporal granularities instant by instant. Then, it check if temporal quantifications are fulfilled.

Fig. 5.2 depicts the checking of the *Overlap* spatial relationship between two spatio-temporal granularities A and B. Depending on the specified temporal qualifiers the relationship is true or not:

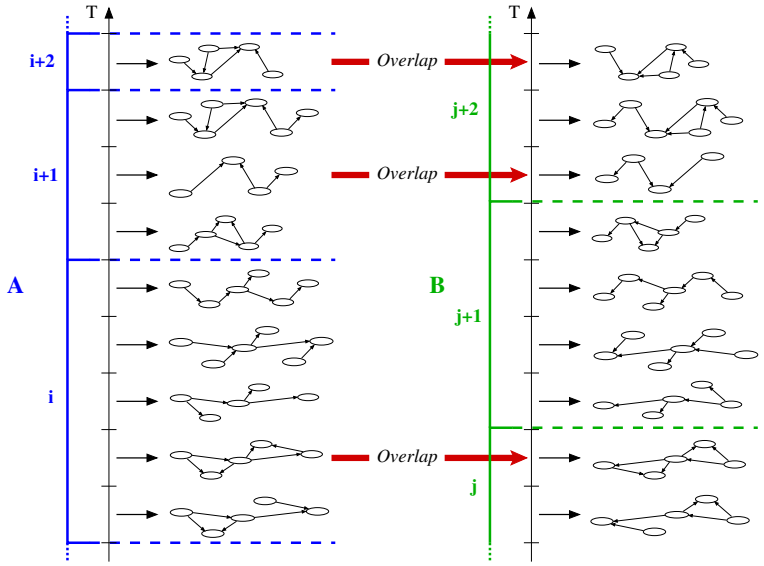


Fig. 5.2: Example of a spatio-temporal relationship.

- $\forall\forall$**Overlap**(A, B) is not valid since there are time points at which **Overlap** is not true;
- $\forall\exists$**Overlap**(A, B) is valid because for each time granule of A there exists a time point where **Overlap** holds;
- $\exists\forall$**Overlap**(A, B) does not hold since there is no time granule in A in which **Overlap** is true at all time points;
- $\exists\exists$**Overlap**(A, B) is true because there exists at least one time granule in which **Overlap** holds in at least one time instant (moreover, it logically follows from $\forall\exists Overlap(A, B)$ [62]).

This extension allows one to represent concepts similar to "spatial relation R is always valid" or "for each time granule exists a time point in which spatial relation S is valid". For example, we can state that municipalities always partition provinces, $\forall\forall$**Partition**(`stMunicipalities`, `stProvinces`).

To better understand, let us consider an example. Epidemiologists may want to know whether every day there exists time points during which the spatial granularity representing areas with high noise pollution intersects the granularity representing quarters in the Verona city center. This issue can be addressed by the relation $\forall\exists$**Overlap**(`stHighNoise`, `stCenterQuarters`) where `stHighNoise` is the spatio-temporal granularity based on days representing areas where high noise pollution have been measured and `stCenterQuarters` is the spatio-temporal granularity representing quarters in the city center of Verona.

We note that each relation can be formally expressed by a logical formula. For example

$$\forall\exists\textbf{Overlap}(A, B) \equiv \forall i \in tG.I \exists t \in tG(i).Overlap(A.E(t), B.E(t))$$

where we remember that $tG.I$ is the set of indexes in the tG temporal granularity and $A.E(t)$ and $B.E(t)$ are the spatial granularity valid at time t in A and B, respectively.

Spatio-temporal relationships can be extended also for comparing one spatio-temporal granularity with a spatial one. In this case, the spatial granularity given as parameter is considered be valid at each time point. Thus, it is compared instant by instant with spatial granularities associated to time points in the given spatio-temporal granularity. For example, let `stHighNoise` be the spatio-temporal granularity we introduced before and representing areas where high noise pollution has been measured. Moreover, let `sAreas` be the spatial granularity representing the CPS areas we introduced in Section 3.2. Thus, $\exists\forall$**Disjoint**(`stHighNoise`, `sAreas`) checks whether it exists a day where all CPS areas do not overlap any area with high noise pollution. Symmetrically, the spatial granularity can be provided as first parameter to the spatio-temporal granularity. In this case, the meaning is the same of the previous case, but the temporal quantifier at granule level is applied to the temporal granularity on which the second parameter (i.e., the spatio-temporal granularity) is based. Thus, for example, $\forall\exists$**FinerThan**(`sAreas`, `stHighNoise`) checks whether in each day (i.e., the temporal granularity of `stHighNoise`) it exists a time point at which CPS areas are included in high noise pollution areas.

5.2.3 Operations over granularities

Let us now consider some operations useful to manage spatio-temporal granularities. Since spatio-temporal granularities essentially record spatial granularities, it is possible to extend spatial operations to spatio-temporal granularities adding time to them. For this purpose, we apply the *lifting* of operations proposed by Güting et al. [108] for extending spatial operations to spatio-temporal moving objects. By using this approach, each spatio-temporal operation (whose name is equal to the correspondent spatial one prefixed by *st*) applies, instant by instant, the correspondent spatial operation to all spatial granularities associated to the spatio-temporal granularity given as parameter.

Thus, let $<Oper>$ be an unary operation over spatial granularities requiring a set of (possibly empty) parameters Par. We define the corresponding operation on spatio-temporal granularities, $st<Oper>$, as follow: $stG' = st<Oper>(stG, Par)$ is such that for each instant t, $stG'.E(t) = <Oper>(stG.E(t), Par)$. In other words, $st<Oper>$ applies the original spatial operation $<Oper>$ to each spatial granularity recorded in the spatio-temporal one.

For example, **stSubset**(`stNoise`, $\{high\}$) returns the `stHighNoise` spatio-temporal granularity we used before for exemplifying spatio-temporal relationships. It is obtained selecting, instant by instant, only the granule with the *high* label from the spatio-temporal granularity representing the evolution over time of noise pollution level. Fig. 5.3 depicts a qualitative sketch of this operation.

On the other hand, considering a binary operation $<Oper>$ over spatial granularities (with a possibly empty set of parameters Par), we define two spatio-temporal extensions. The first one allows one to apply an operation to a spatio-temporal granularity and a spatial one. It is $stG' = st<Oper>(stG, sG, Par)$ such that for each instant t, $stG'.E(t) = <Oper>(E(t), sG, Par)$. In other words, the

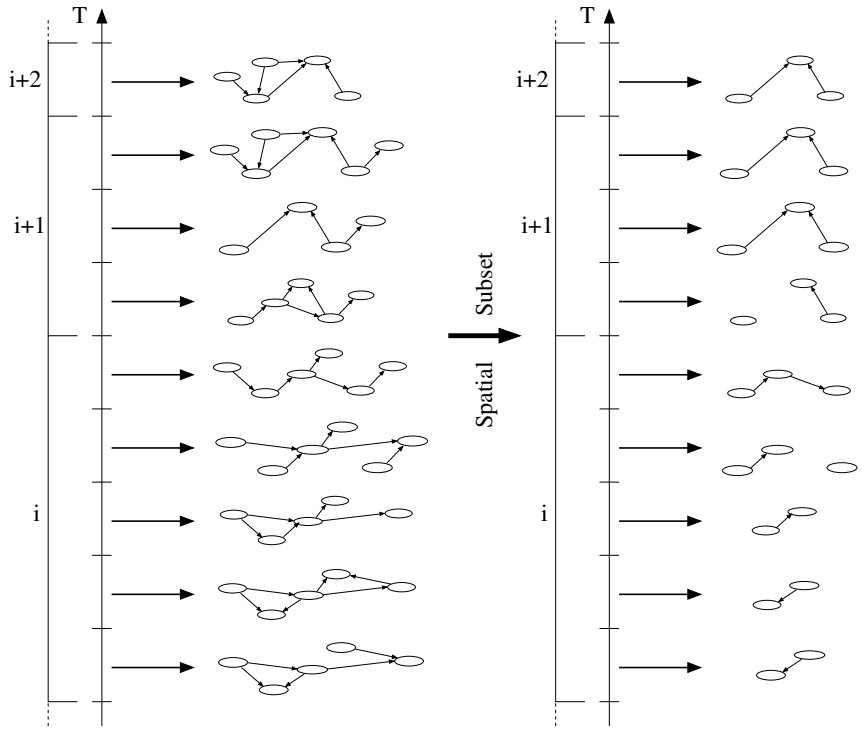


Fig. 5.3: An example of spatio-temporal operations

corresponding spatial operation is applied at each instant t to $E(t)$ and the spatial granularity sG. For example, the operation **stIntersect**(`stHighNoise`, `sAreas`) returns the spatio-temporal granularity representing spatial granularities whose granules are only those parts of CPS areas where high noise pollution has been surveyed.

The second extension allows one to apply operations to two spatio-temporal granularities. It is $stG' = st{<}Oper{>}(stG, stH, Par)$ such that at each instant t, stG' returns the granularity resulting from $<Oper>(stG.E(t), stH.E(t), Par)$. In other words, the operation is applied at each instant t to the spatial granularities associated to t in stG and stH. For example, the operation

stSelectIntersect(`stQuarters`, `stHighNoise`)

returns the spatio-temporal granularity obtained by selecting, at each time point, quarters in Verona that intersect areas where high noise pollution has been measured.

5.3 Raster-based spatio-temporal granularities

In this section we extend the notion of spatial granularity for raster maps to the spatio-temporal case. To do that, we follow the same approach used for vector-based spatio-temporal granularities (see Section 5.2). This approach has been used

also by Frank [91] for representing time series of raster maps. However, they do not deal with granularities but only with maps.

5.3.1 Spatio-temporal granularities

We denote with Σ_L the set of all raster spatial granularities (see Section 4.3) over the label set L. Hence, a raster-based evolution over L, that represents the evolution over time of a raster-based spatial granularity, is defined as a total function $\epsilon : \mathbb{Z} \to \Sigma_L$ that associates to each time point the spatial granularity that is valid on it. In raster granularities we consider that in some instants the corresponding raster maps is unknown or undefined. Thus, in order to ensure that an evolution is a total mapping, we impose that, whenever no spatial granularity is valid at one time point, the evolution associates $\sigma_\perp$ to this time instant. Where, we denote with $\sigma_\perp$ the spatial granularity that associates to each point in $\mathbb{Z}^2$ the value $\perp$. The spatial granularity valid at time t is $\epsilon(t)$. The spatial granule with label l at time t is denoted with $\gamma_{\epsilon(t)}(l)$.

We define a raster-based spatio-temporal granularity τ over a label set L as a pair $\langle tG, \epsilon \rangle$ composed by a temporal granularity tG [23] and a raster-based evolution ϵ over L. The temporal granularity tG associates to each index i of an index set I (e.g., $\mathbb{Z}$) the set of all time points belonging to the time granule identified by i. In this way, it aggregates time points and the spatial granularities valid on them. The spatio-temporal granule $\tau(i, l)$ represents the evolution during time granule i of the spatial granule l. We call $\mathcal{T}_L$ the set of all raster spatio-temporal granularities over the label set L.

Fig. 5.4 depicts the spatio-temporal granularity $\langle \texttt{tYears}, \texttt{LandUsage} \rangle$. To each time point t_i (aggregated with respect to the temporal granularity `tYears`), the spatial granularity valid on it is associated. In Fig. 5.5 the spatio-temporal granule representing residential cells during the time granule 2010 is depicted.

5.3.2 Relationships

We now define relationships between raster spatio-temporal granularities extending the spatial ones given in Section 4.3. For this purpose we use the same approach we followed in previous section about relationships between spatio-temporal granularities based on the vector data model. Thus, the basic ideas for defining relationships between two spatio-temporal granularities is to compare, instant by instant, the spatial granularities they store. However, we can constrain instants to be compared. In spatio-temporal granularities, time is arranged in two levels, granule and time point level. Thus, we introduce a temporal quantification operator at both these levels. Possible temporal quantification operators are *All* ($\forall$) and *Exists* ($\exists$). These quantifiers, prefixed to the spatial relation R, specify when R must be verified in order that the spatio-temporal relation holds. The temporal quantifier at the granule level allows us to require that the relationship must hold between spatial granularities associated to all or at least one time granule. On the other hand, the temporal quantifier at the time point level allows us to require that the relationship must hold between spatial granularities associated to all or at least one time point in considered time granules. In this way, for example, given two

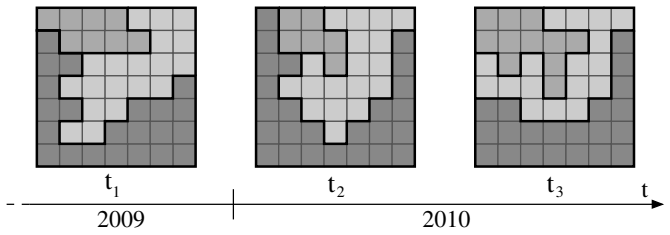


Fig. 5.4: An example of raster-based spatio-temporal granularity

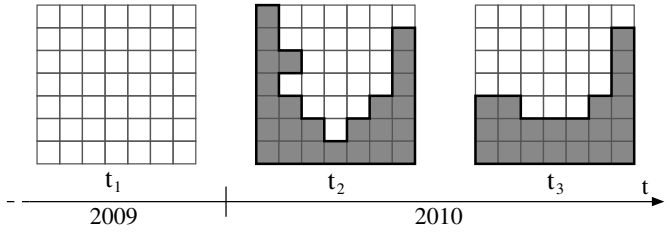


Fig. 5.5: The $\tau(2010, \mathsf{Residential})$ spatio-temporal granule

spatio-temporal granularities $\tau_1 = \langle tG_1, \epsilon_1 \rangle$ and $\tau_2 = \langle tG_2, \epsilon_2 \rangle$ the spatio-temporal relation $\forall\exists R(\tau_1, \tau_2)$ corresponds to impose that

$$\forall i \in I.\exists t \in tG_1(i).R(\epsilon_1(t), \epsilon_2(t)),$$

i.e., in each time granule of tG_1 there exists an instant on which R holds between the spatial granularities associated to that instant in τ_1 and τ_2. Similar definitions can be given for the other three possible combinations of the two time quantification operators (i.e., $\forall\forall$, $\exists\exists$, $\exists\forall$) whose meaning is the same described in the previous section. For example, the spatio-temporal relationship $\forall\forall$**FinerThan**(`stCultures`, `stSoilType`) checks whether, at every time point of each time granule, the cultures types are finer than the soil types.

5.3.3 Operations

Operations over spatial granularities can be similarly extended in order to achieve a definition over spatio-temporal granularities. The main idea is to redefine them in order to accept spatio-temporal parameters and to return a spatio-temporal granularity. To do that, we apply the same approach we used for vector-based spatio-temporal operation in previous section. We remember that, in this approach, each spatio-temporal operation (whose name is equal to the correspondent spatial one) simply applies, instant by instant, the corresponding spatial operation to all spatial granularities associated to the spatio-temporal granularity that has been provided as parameter. This extension is reflected by signatures of operations. Unary (**Grouping**, **Subset**, and **Relabel**) and binary (all the other ones) operations over spatial granularities have the following signatures (where Par generically represents the type of non-spatial parameters requested by unary operations):

$Op_1 : \Sigma_L \times Par \rightarrow \Sigma_{L_1}$ and
$Op_2 : \Sigma_{L_1} \times \Sigma_{L_2} \rightarrow \Sigma_L$,

respectively, and they can be extended as follow:

$Op_1 : \mathcal{T}_L \times Par \to \mathcal{T}_{L_1}$,
$Op_2 : \mathcal{T}_{L_1} \times \mathcal{T}_{L_2} \to \mathcal{T}_L$,
$Op_2 : \mathcal{T}_{L_1} \times \Sigma_{L_2} \to \mathcal{T}_L$, and
$Op_2 : \Sigma_{L_1} \times \mathcal{T}_{L_2} \to \mathcal{T}_L$

in order to accept spatio-temporal parameters. Binary operations can accept two spatio-temporal parameters or one spatio-temporal granularity plus a spatial one.

Formally, the evolution resulting from a unary spatio-temporal operation Op_1, applied to $\langle tG, \epsilon \rangle$, is ϵ' such that $\forall t \in \mathbb{Z}.\epsilon'(t) = Op_1(\epsilon(t), Par)$, i.e., Op_1 is applied to each spatial granularity evolving in ϵ. While evolutions resulting from binary operations $\tau' = Op_2(\tau_1, \tau_2)$, $\tau'' = Op_2(\tau_1, \sigma)$, and $\tau''' = Op_2(\sigma, \tau_2)$ are ϵ', ϵ'', and ϵ''', respectively, such that, for each time point $t \in \mathbb{Z}$:

$\epsilon'(t) = Op_2(\epsilon_1(t), \epsilon_2(t))$,
$\epsilon''(t) = Op_2(\epsilon_1(t), \sigma)$, and
$\epsilon'''(t) = Op_2(\sigma, \epsilon_2(t))$.

Spatio-temporal operations allow one to calculate, for example,

$$\texttt{stResidential} = \textbf{Subset}(\texttt{stLandUsage}, \{\mathsf{Residential}\})$$

representing the spatio-temporal granularity in which, at each time point t, only the granule labeled with Residential is selected. On the other hand, the binary operation

$$\textbf{SelectInterset}(\texttt{stPollution}, \texttt{stResidential})$$

creates the granularity representing at each time point the pollution granules intersecting residential areas. While with **SelectIntersect**(`MineralRes`, `stUnused`) we calculate the granularity representing at each time the mineral resources (that do not change over time) that intersect unused areas (that evolve over time).

Finally, it is useful to define the new **Remodulate** operation that, given a spatio-temporal granularity $\tau = \langle tG, \epsilon \rangle$ and a temporal granularity tH, creates a new spatio-temporal granularity replacing with tH the temporal granularity in τ. Formally:

$$\textbf{Remodulate}(\tau, tH) \triangleq \langle tH, \tau.\epsilon \rangle$$

For example, from `stLandUsage` based on the temporal granularity `tMonths`, we can obtain the land usage based on years with **Remodulate**(`stLandUsage`, `Years`).

6

An inference system for relationships between granularities

In previous chapters, we said that spatio-temporal reasoning on data qualified with granularities may depend on the relationships between the involved granularities. Different relationships may require different approaches and algorithms for the comparison and transformation of the qualified data. Thus, the evaluation of the relationships between the granularities is an important task. Thus, in this chapter we discuss the algorithms for the evaluation of the relationships we proposed in the frameworks. In particular our goal is to study their computational complexity in time. We show that they may be computational heavy, especially due to the large number of granules and granularities to be checked. In order to avoid the execution of these heavy algorithms for determining the relationships that hold between given temporal, spatial, and spatio-temporal granularities, we introduce an inference system for relationships between granularities. Indeed, in several cases, we can know whether a relationship holds or not by inferring that from other relationships that we already know to hold or not. Thus, given a starting set of relationships between granularities that are supposed to hold, the inference system is able to infer all the other relationships that definitely hold between the involved granularities, i.e., the relationships that hold in any possible model satisfying the relationships in the starting set. The inference system comprises a set of inference rules. We define the semantics of the inference system and we prove its soundness and completeness with respect to this semantics. We show also an example where inference rules have been applied for inferring relationships that otherwise are not clear to hold and should need to be evaluated through algorithms.

The chapter is organized as follow. In next section we will present main proposals in literature about inference of spatio-temporal relationships. In Section 6.2 we discuss the computational complexity of the algorithms for the evaluation of relationships between granularities. In Section 6.3 we present the basic ideas we used to design the inference system, while in Section 6.4 we introduce the semantics of the relationships, i.e., the properties of relationships that should be preserved also by inference rules. In Section 6.5 we present the inference rules in the inference system. Since the ideas we used for studying the inference rules for the three kinds of granularities (i.e., temporal, spatial, and spatio-temporal granularities) are the same, we will discuss here only the inference system for spatial granularities. In Section 6.6 we prove that the proposed inference system is sound and complete

with respect the semantics we introduced. In Section 6.7 we show a little example demonstrating how the inference system is able, applying iteratively the rules, to infer, starting from a starting set of relationships, relationships that otherwise would not be clear to hold. Finally, in Section 6.8 we briefly discuss the temporal and spatio-temporal inference rules by presenting features distinguishing them from the spatial one.

6.1 Related work

The well-known framework for temporal relationships is the Allen's interval algebra [6]. Allen proposes a notion of temporal interval and defines a set of relationships between intervals (e.g., before, during). Relationships are represented by using a constraints network. The network is built and expanded applying a transitivity table. Given a relationship between two intervals A and B and a relationship between B and C, the transitivity table calculates the relationships definitely true between A and C. The Allen's approach and notions have been further developed and extended in several direction, e.g., fuzzy temporal intervals [15] and temporal semi-intervals [92].

Similar frameworks have been proposed also for spatial regions, especially based on Region Connection Calculus (RCC) [171]. Randell et al. present an algebra allowing one to express topological relationships (e.g., contains, overlap, disjoint) between spatial regions. Moreover, similarly to the Allen's proposal, they define a transitivity table in order to infer composition of two RCC relationships. The RCC framework has been further extended to consider regions with a single hole in [208].

In [190], Sistla and Yu present a similar framework focused on pictorial databases. They propose nine relationships (e.g., left_of, above, inside) and study a deductive system able to infer new relationships from a given set. In particular they studied transitivity, symmetry, and other properties of relationships.

Wiebrock et al. [217] propose a model to represent spatial relationships based on transformation matrices. Then, they define an inference system based on matrices manipulation.

In [30] Bittner proposed the notion of stratified spatio-temporal map spaces, a spatio-temporal extension of stratified map spaces [197]. Based on this notion, Bittner introduces a set of spatio-temporal relationships (e.g., different-time-same place, same-time-same-place, different-time-different-place) between two moving spatial regions. Finally, the author proposes a composition operator for these relationships, also considering different levels of map space (i.e., different temporal and spatial granularities).

6.2 Algorithms for relations and operations

In order to provide the pseudo-code of algorithms, we use an object-based approach to procedure calls. Hence, the statement a`.foo(`b`)` means that the function `foo` is called over the object a with parameter b.

Algorithm 1 $GroupsInto(G, H)$

```
Require: G and H spatial granularities
1: for all w ∈ H.V do
2:   RS = Search_G(w)
3:   u = emptyGeometry()
4:   for all g ← extract(RS) do
5:     if G(g).within(H(w)) then
6:       u = u.union(G(g))
7:     else
8:       return  false
9:   if not(u.equals(H(w))) then
10:     return  false
11: return  true
```

In Algorithm 1 the pseudo-code implementing the weak relation $GroupsInto$ is reported. This procedure requires two spatial granularities G and H, and returns `true` if and only if G groups weakly into H. It consists of a loop (lines 1–10) that checks whether each granule of H is equal to the union of a set of granules of G.

For each granule w belonging to H, the procedure retrieves the granules of G not disjoint to w by using the function $Search_G$. This function can be implemented by using R-trees [110] in order to reduce its computational complexity.

Granules of G intersecting w are collected in RS. We can distinguish two kinds of granules inside it (lines 5–8): the covered ones and the others. If g is contained in (i.e., within) w (line 8), we join it to a cumulative temporary geometry u (line 6). On the other hand, if a granule $g \in RS$ is not within w (i.e., g intersects w but it is not contained in w) then we know without other information that w cannot be obtained by the union of granules of G and, hence, G does not group into H. As a matter of fact, if we take g, we obtain a geometry larger than w, while if we do not take it we obtain a geometry smaller of w. It is thus impossible to obtain exactly w and the algorithm returns `false`.

Once we have processed all granules g intersecting w, we check (line 9) whether their union u is equal to w or not. In the first case we continue the algorithm returning to line 2 and processing next granule of H. In the second case we return `false` because we found a granule (i.e., w) that is not the union of a set of granules belonging to G and hence G does not group into H.

Finally, when we have processed all granules of H and we know that every one is the union of a set of granules of G, we return `true` (line 11).

We note that geometrical functions (e.g., `overlaps`, `within`, and `equals`) and R-trees are common functions and structures implemented by several spatially extended database systems (e.g., PostGIS/PostgreSQL [168,172], Oracle [159]).

Studying the computational complexity of this algorithm, we assume that spatial functions (e.g., `overlaps`, `within`, and `equals`) are implemented in time $O(1)$. The same assumption can made for the function `extract`. Hence, in the rest of this section we express computational complexity with respect to basic and geometrical operations. Moreover, given two granularities G and H, we call n and m the cardinalities of $|G.V|$ and $|H.V|$, respectively.

Algorithm 2 $GroupsInto_R(G, H)$

```
Require: G and H spatial granularities, R spatial relation
1: if not(GroupsInto(G, H)) then
2:   return false
3: L1 = G.D_A^{-1}(R)
4: L2 = H.D_A^{-1}(R)
5: for all (v1,v2)_{L1} do
6:   w1 = v1.belongsTo(H)
7:   w2 = v2.belongsTo(H)
8:   if not(w1.equals(w2)) ∧ ∄(w1, w2)_{L2} then
9:     return false
10: return true
```

The loop in lines 1–10 is executed exactly m times, one for each granule of H. Each repetition requires $O(n)$ operations. As a matter of fact, every operation inside it can be calculated in $O(1)$ but $\texttt{Search}_G$ (line 2) that, in the worst-case, requires $O(n)$ operations. Moreover, the `for` loop in lines 4–8 is repeated $O(n)$ times in the worst-case and each repetition requires $O(1)$ operations. Thus, the whole algorithm requires $m \cdot O(n)$ operations. Hence, its computational complexity in time is $O(mn)$.

We note that this is the worst-case performance. When G does not group into H, probably the algorithm stops earlier. Moreover, the algorithm can be optimized by implementing efficiently $\texttt{Search}_G$. Each time a granule of G is retrieved, it can be removed from next search since it is useless to compare it with two different granules of H. Indeed, if a granule $g \in G.V$ intersects $w \in H.V$, then we may either use it to obtain w or we can remove it because it cannot be within another granule $w_1 \in H$ (otherwise w intersects w_1 and this is not possible cause the definition of granularity). Implementing this optimization, we obtain that each granules of G is processed just one time during the whole algorithm. Hence the weak relation $GroupsInto$ can be tested using $O(m + n)$ operations.

Algorithms implementing relations $FinerThan(G, H)$, $Subgranularity(G, H)$, $Partition(G, H)$, and $Partition_t(G, H)$ have the same computational complexity. On the other hand, relations $CoveredBy$, $Disjoint$, and $Overlaps$ can be calculated in the worst-case in $O(n+m)$ time, given that we have to calculate the image of the granularities.

Algorithm 2 reports the strong version of $GroupsInto$. It requires two spatial granularities G and H and a spatial relation between granules R. It returns `true` if and only if G groups strongly into H with respect to R.

First of all, the algorithm checks (line 1) whether G groups weakly into H or not. In the second case the procedure returns `false` (line 2). The rest of the algorithm checks whether edges in G representing R are preserved also in H or not.

The procedure retrieves labels L_1 and L_2 corresponding to the relation R in G and H, respectively (lines 3–4). The `for` loop (lines 5–9) is repeated one time for each edge, $(v_1, v_2)_{L_1}$, between two granules v_1 and v_2 of G labeled with L_1. During each repetition the algorithm retrieves (lines 6–7), by using the function

`belongsTo`, granules w_1 and w_2 of H containing v_1 and v_2, respectively. We note that these two granules necessarily exist since G groups weakly into H. In lines 8–9 the algorithm checks whether exists or not an edge in H between w_1 and w_2 representing R (i.e., labeled with L_2). In the second case it returns `false` without other tests. Otherwise the procedure continues going back to line 5 and processing the next edge in G. We note that the definition of $GroupsInto_R$ requires that w_1 and w_2 must be different. This test is reported in the `if` statement at line 8. Finally, when the algorithm has processed every edge $(a, b)_{L_1}$ in G finding that each one is preserved in H, it returns `true` (line 10).

Let us analyze the computational complexity of the algorithm. The `if` statement in lines 1–2 consists mainly of a procedure call checking whether G groups weakly into H or not. As we have seen in Algorithm 1, this task requires in the worst-case $O(m+n)$ operations. Lines 3 and 4 require $O(|G.\Sigma_A|)$ and $O(|H.\Sigma_A|)$ operations, respectively. The `for` loop is repeated $O(n^2)$ times, one for each edge in G. The function `belongsTo`, and hence lines 6 and 7, can be computed using $O(m)$ operations. While the `if` statement at line 8 consists mainly in searching an edge in H. Hence, it requires $O(m^2)$ operations. Then, the whole `for` loop requires $O(n^2) \cdot (2 \cdot O(m) + O(m^2))$, i.e., $O(n^2 m^2)$, operations.

Thus, we may observe that $GroupsInto_R(G, H)$ can be calculated in the worst-case by using $O(|G.\Sigma_A| + |H.\Sigma_A| + n^2 m^2)$ operations.

Algorithms implementing strong relations $FinerThan_R(G, H)$ and $Subgranularity_R(G, H)$ have the same computational complexity.

6.3 The overall approach

As we said in the previous section, the algorithms for relationships between granularities can be computational heavy. In spite of their computational complexity is not very high, we have to consider that they can be applied to many granularities and to granularities with many granules, hence the algorithms can require an important amount of time. For this reason, it may be useful to introduce an inference system for relationships. As we mentioned before, the inference system avoid in some cases the execution of the algorithms for evaluating relationships.

Given a set $\mathcal{R}$ of relationships between spatial granularities over a set $\mathcal{G}$ of granularities, the inference system automatically infers all other relationships definitely valid over $\mathcal{G}$. In other words, it propagates the given constraints.

Note that the inference system does not know the actual definition of the granularities over which it will operate (i.e., their graphs, granules, and granules extent), it knows only some relationships between them. In other words, the system works on an abstraction of the real granularities contained in the database. This abstraction does not consider the low level representation of the granularities, i.e., granules, extents, and graphs, but only the properties, the relationships of the granularities. Thus, the system is not able to compute a truth value for each possible relationship between two given granularities, but only for someones. In some cases, nothing can be decided about some relationships.

We studied several kinds of rules for answering to the following questions.

- Are relationships reflexive?

- Knowing that $R(G, H)$ is valid, what other relationships must be valid between G and H?
- Knowing that $R(G, H)$ is valid, what other relationships must be valid between H and G?
- What other relationships can be inferred between G and H knowing that R_1 and R_2 are both valid between them?
- Is it possible to concatenate the relationships? In other words, knowing that $R_1(G_1, G_2)$ and $R_2(G_2, G_3)$ are valid, what can we infer about G_1 and G_3? Note that, assuming R_1 and R_2 are the same relationship, we study also the transitivity of the relationships.

Rules in proposed inference system allow only premises composed either by one relationship or by a conjunction of relationships. While the conclusion of rules must be just one relationship. Disjunctions, in premises and conclusions, are not permitted.

The proposed inference system does not infer only relationships that definitely hold on the considered set of granularities, it returns also the set of relationships that surely do not hold (assuming that considered granularities are not equivalent). We denote that a relationship $R(G, H)$ does not hold with $\neg R(G, H)$.

Besides relationships presented in Chapter 4, we consider also the equivalence relationship. We say that two granularities are equivalent, $G \approx H$, if both contain exactly the same granules with the same extents, without regard to their labels, i.e. for each granule of G, there exists a granule in H with the same spatial extent and vice versa. Equivalence is important because if two granularities are equivalent all relationships, but $Disjoint$ and $Overlap$, hold between them. In other words, the equivalence relationship implies any other relationships, but $Disjoint$ and $Overlap$. Moreover, the equivalence relationship is monotonic, i.e., if a relationship R holds between two granularities G and H and the granularity I is equivalent to G (resp., to H) then the relationship R holds also between I and H (resp., between G and I).

Thus, for example, if $GroupsInto(G, H)$ is in the given relationships set, the system infers that also $CoveredBy(H, G)$ must be valid. Moreover, if also $\neg G \approx H$ is in the set, the system infers that $SubGranularity(G, H)$ cannot be true, i.e., $\neg SubGranularity(G, H)$.

The inference system is based on the application of the set of rules on a starting set of relationships. This set can be obtained also by analyzing how the granularities we are interested in have been created. Knowing which operation has been used to define a granularity (see Chapter 4), we can infer some relationships between it and the operand granularities. However, these rules are not really part of the inference system because they do not infer relationships from other relationships. These rules can be considered as the first and starting point for the application of the inference system. For example, if $G' = SelectInside(G_1, G_2)$ we can deduce that G_1 groups into G' and that G' is finer than both G_1 and G_2.

Inference rules from operations used to define spatial granularities are depicted in Tables B.6 and B.7.

6.4 Semantics of relationships between granularities

As we said in the previous section, the inference system is theoretically based on an abstraction of our frameworks for granularities. In this model only relationships are important, while it does not regard how granularities are defined. In this section we present this abstract model and its semantics. Moreover we show that this abstraction in consistent with our theoretical framework.

Definition 6.1. *A spatial granularity model is a pair $(\mathcal{W}, \mathcal{R})$, where $\mathcal{W}$ is a non-empty set of worlds and $\mathcal{R}$ is a set of binary relationships over $\mathcal{W}$.*

Let $\mathcal{M}$ be a model and λ an interpretation on it that maps each granularity label to a world in $\mathcal{W}$. Validity of relationships between spatial granularities is represented by the smallest relationship $\models^{\mathcal{M},\lambda}$ (if any) satisfying:

- the following constraints (where Γ represents a set of relationships) that represent the general system behavior:
  - $\models^{\mathcal{M},\lambda} R(G_1, G_2)$ iff $R(\lambda(G_1), \lambda(G_2)) \in \mathcal{R}$
  - $\models^{\mathcal{M},\lambda} \neg R(G_1, G_2)$ iff $R(\lambda(G_1), \lambda(G_2)) \notin \mathcal{R}$
  - $\models^{\mathcal{M},\lambda} R_1(G_1, G_2) \wedge \Gamma$ iff $\models^{\mathcal{M},\lambda} R(G_1, G_2)$ and $\models^{\mathcal{M},\lambda} \Gamma$
  - $\Gamma \models^{\mathcal{M},\lambda} R(G_1, G_2)$ iff $\models^{\mathcal{M},\lambda} \Gamma$ implies $\models^{\mathcal{M},\lambda} R(G_1, G_2)$
- the constraints in Figures A.2 and A.3 that represent the behavior with respect to relationships between granularities, which semantics is the one informally presented in Chapter 4.

We remember that the notation $\bigwedge_{x \in S} .P(x)$ means that, for each value taken by x in the set S, the predicate $P(x)$ holds.

For example, the (SED) constraint imposes that if in the model $D(G_1, G_2)$ is valid, then in the model also $\neg R(G_1, G_2)$ must be valid for each spatial relationship R. On the other hand, (EGI) says that when $GI(G_1, G_2)$ is valid in the model then also $CB(G_2, G_1)$ is valid.

Now, we show that this model is sound, that is, that each assertion on it is true also in the theoretical model for spatial granularities. We remark that this is necessary in order to show that the abstraction over which the inference system is based (i.e., the model) respects the proposed framework.

Proposition 6.2. *The proposed model, and its semantics, for the inference system over spatial granularities is sound.*

Proof. We have to show that each property in Fig.s A.2 and A.3 is valid also in the theoretical framework for spatial granularities. We present just some cases while the other ones are analogous.

(trans) We need to prove that *GroupsInto*, *FinerThan*, *SubGranularity*, *Partition*, and *CoveredBy* are transitive also in our framework. Let us consider *GroupsInto* relationship and let G_1, G_2, and G_3 such that G_1 groups into G_2 and G_2 groups into G_3. Thus, by the definition of *GroupsInto*, each granule of G_2 is equal to the union of a set of granules of G_1, and each granule of G_3 is equal to the union of a set of granules of G_2. Since, each granule of G_3 is the union of granules in G_2 and each of these is in turn equal to the union of some granules

of G_1, then we can conclude that each granule of G_3 is equal to the union of some granules in G_1. Thus, G_1 groups into G_3.
The treatment of $FinerThan$, $SubGranularity$, $Partition$, and $CoveredBy$ is very similar.

(refl) We need to show that $GroupsInto$, $FinerThan$, $SubGranularity$, $Partition$, and $CoveredBy$ are reflexive. Let us consider $GroupsInto$ relationship and let G a spatial granularity. Obviously, each granule of G is equal to union of the singleton made up of just itself, thus G groups into itself, i.e., $GroupsInto(G, G)$.
The treatment of $FinerThan$, $SubGranularity$, $Partition$, and $CoveredBy$ is very similar.

(antirefl) We need to show that $Disjoint$, $Overlap$ are antireflexive. Let us consider $Disjoint$ and let G a spatial granularity. Obviously the image of G cannot be disjoint by itself, thus G cannot be disjoint by itself, i.e., $\neg Disjoint(G, G)$.
The treatment of $Overlap$ is very similar.

(antisymm) We need to show that $GroupsInto$, $FinerThan$, $SubGranularity$, and $Partition$ are antisymmetric. Let us consider $GroupsInto$ relationship and let G_1 and G_2 two spatial granularities such that G_1 groups into G_2 and G_2 groups into G_1. Each granule g_1 of G_1 is equal to the union of a set S_2 of granules of G_2, and, in turn, each granule g_2 in this set is equal to the union of a set S_1 of granules in G_1. But, S_2 and S_1 must be the singletons containing just g_2 and g_1 respectively, otherwise g_1 would be equal to the union of other granules in G_1, that it is not possible since granules in a granularity cannot intersect each other. Thus, each granule g_1 in G_1 must be equal to a granule g_2 in G_2. Repeating the same argument starting from granules in G_2, we obtain that G_1 and G_2 must contain the same granules, i.e., they are equivalent.
The treatment of $FinerThan$, $SubGranularity$, and $Partition$ is very similar.

6.5 The inference system

Tables from B.1 to B.7 present, in tabular form, the rules of the inference system for spatial granularities. These rules answer to questions reported in Section 6.3. We remind that all rules concluding the non validity of a relationship have also the premise that considered granularities are not equivalent, otherwise all relationships hold obviously between them.

In these tables, the ✓symbol means that the corresponding relationship can be inferred, the × symbol means that the relationship does not hold, and, finally, the - symbol means that nothing can be decided about that relationship.

In particular, Table B.1 tell us which other relationships between G and H can be inferred from a relationship $R(G, H)$. Rules in Table B.2 infer relationships between H and G starting from $R(G, H)$. Table B.3 contains rules inferring from a pair of relationships $R_1(G, H)$ and $R_2(G, H)$. Finally, tables B.4 and B.5 represent the composition rules for relationships between spatial granularities.

The antisymmetry of relationships is stated, as a special case, by Table B.2, while transitivity is a special case of the relationship concatenation studied in tables B.4 and B.5.

Besides rules in these tables the inference system includes also rules stating the reflexivity

$$\bigwedge_{R\in\{GI,FT,SG,P,CB\}} . \vdash R(G,G)$$

and antireflexivity

$$\bigwedge_{R\in\{D,O\}} . \nvdash R(G,G)$$

Moreover, the system needs also some rules for describing the behavior of equivalence (where the meaning of R is not specified it may be any relationship we introduced):

- $\vdash G \approx G$
- $G \approx H \vdash H \approx G$
- $G_1 \approx G_2 \wedge G_2 \approx G_3 \vdash G_1 \approx G_3$
- $\bigwedge_{R\in\{GI,FT,SG,P,CB\}} . G \approx H \vdash R(G,H)$
- $R(G_1,G_2) \wedge G_1 \approx G_3 \vdash R(G_3,G_2)$
- $R(G_1,G_2) \wedge G_2 \approx G_3 \vdash R(G_1,G_3)$

The first three rules state reflexivity, symmetry, and transitivity of equivalence, respectively. The fourth rule states that equivalence implies any other relationship, while last two rules state the monotonicity of equivalence with respect to all other relationships.

When the set of relationships the system is considering contains both a relationship and its negation, the set is said to be inconsistent and the system infers the special symbol $\perp$, i.e., $R(G,H) \wedge \neg R(G,H) \vdash \perp$.

Finally, we introduce the *RAA* rule representing the *reductio ad absurdum*.

$$\dfrac{\begin{array}{c}[\neg R(G,H)] \\ \vdots \\ \perp\end{array}}{R(G,H)} \quad \text{(RAA)}$$

When a derivation that includes $\neg R(G,H)$ reaches a contradiction, the system can infer that $R(G,H)$ must hold.

This rule is suitable for our inference system because all relationships are Boolean and each relationship has for sure a truth value, i.e. either $R(G,H)$ or $\neg R(G,H)$ must be true. This rule is based on the notion of discharged assumption that is standard in Natural Deduction [16] proof systems. The relationship $\neg R(G,H)$ is discharged during the rule application.

Note that by using (RAA) we can obtain contrapositives of the other rules. For example, from a rule of the form $R(G_1,H_1) \vdash R(G_2,H_2)$ we can obtain the equivalent rule $\neg R(G_2,H_2) \vdash \neg R(G_1,H_1)$ with the following proof:

$$\dfrac{\dfrac{\dfrac{[\neg\neg R(G_1,H_1)]^2 \quad [\neg R(G_1,H_1)]^1}{\bot}}{R(G_1,H_1)}\,{}^1}{\dfrac{R(G_2,H_2) \qquad\qquad \neg R(G_2,H_2)}{\bot}}$$
$$\neg R(G_1,H_1)\,{}^2$$

Contraposition allows us, for example, to obtain from rules in the inference system that if G is not covered by H, then G is not finer than H.

With very similar proofs, from rules like $R_1(G_1,H_1)\wedge R_2(G_2,H_2)\vdash R_3(G_3,H_3)$ we can obtain equivalent rules $R_1(G_1,H_1)\wedge\neg R_3(G_3,H_3)\vdash\neg R_2(G_2,H_2)$ and $R_2(G_2,H_2)\wedge\neg R_3(G_3,H_3)\vdash\neg R_1(G_1,H_1)$. In this cases contraposition allows us to obtain for example from the antisymmetry rule $\neg G\approx H\wedge GI(G,H)\vdash\neg GI(H,G)$ the rule $GI(G,H)\wedge GI(H,G)\vdash G\approx H$, i.e., if a granularity G groups into H and vice versa then G and H are equivalent.

Finally, also the contrapositive of (RAA) can be obtained in the same way.

6.6 Soundness and Completeness of Inference System

In this section we study soundness and completeness of proposed inference system for spatial granularities. The soundness theorem states that whenever a relationship is inferred by the system, then it holds in every model that satisfies the given starting set of relationships from which the considered relationship has been inferred.

Theorem 6.3 (Soundness). *The proposed inference system for spatial granularities is sound, i.e.* $\mathcal{R}\vdash R(G,H)$ *implies* $\mathcal{R}\models^{\mathcal{M},\lambda}R(G,H)$ *for every model* $\mathcal{M}$ *and every interpretation* λ*.*

Proof. The proof is by induction on the structure of the derivation of $R(G,H)$. The base case is when $R(G,H)\in\mathcal{R}$ and is trivial. Due to the similarity of inference rules and semantics, the proof that all rules are sound is trivial. The only interesting case is the application of RAA.

Let $\mathcal{R}_1$ be $\mathcal{R}\cup\{\neg R(G,H)\}$. By the induction hypothesis, $\mathcal{R}_1\models^{\mathcal{M},\lambda}\bot$ for every model $\mathcal{M}$ and every interpretation λ. Now, consider an arbitrary model $\mathcal{M}$ and an arbitrary interpretation λ, we assume $\models^{\mathcal{M},\lambda}\mathcal{R}$ and prove $\models^{\mathcal{M},\lambda}R(G,H)$. Since $\not\models^{\mathcal{M},\lambda}\bot$, by the induction hypothesis we obtain $\not\models^{\mathcal{M},\lambda}\mathcal{R}_1$, that, given the assumption $\models^{\mathcal{M},\lambda}\mathcal{R}$ leads to $\not\models^{\mathcal{M},\lambda}\neg R(G,H)$. Thus, since a model associates a truth value to each relationship, $\models^{\mathcal{M},\lambda}R(G,H)$. □

Now, we prove the completeness of the proposed inference system for spatial granularities. The system is complete if it is able to infer all the relationships that definitely hold starting from a given set of relationships. The completeness will be proved by proving its contrapositive, i.e., if the system is not able to infer a relationship then it means that there exists a model (the counterpart of an example) where that relationship does not hold, i.e., that the relationship does not hold definitely. Models are defined on maximally consistent sets, thus we prove

that the given starting set of relationships can be extended by using the proposed inference system in such way that the resulting set of relationships is maximally consistent (i.e., no other relationships can be added to the set without obtaining an inconsistent set). We now introduce the notions of consistency and maximally consistency.

Definition 6.4 (Consistency). *A set $\mathcal{R}$ of relationships between granularities is said to be* consistent *if $\mathcal{R} \nvdash \bot$. It is said* inconsistent *otherwise.*

Proposition 6.5. *Let $\mathcal{R}$ be a consistent set of relationships. For each relationship $R(G_1, G_2)$, either $\mathcal{R} \cup \{R(G_1, G_2)\}$ or $\mathcal{R} \cup \{\neg R(G_1, G_2)\}$ is consistent.*

Proof. Let us suppose that $\mathcal{R} \cup \{R(G_1, G_2)\}$ and $\mathcal{R} \cup \{\neg R(G_1, G_2)\}$ are both inconsistent. Thus, $\mathcal{R} \cup \{R(G_1, G_2)\} \vdash \bot$ and $\mathcal{R} \cup \{\neg R(G_1, G_2)\} \vdash \bot$. By using RAA, we obtain that $\mathcal{R} \vdash \neg R(G_1, G_2)$ and $\mathcal{R} \vdash R(G_1, G_2)$ then $\mathcal{R}$ is inconsistent (contradiction). □

Definition 6.6 (Maximal consistency). *A set $\mathcal{R}$ of relationships between granularities is maximally consistent with respect to a set of granularities $\mathcal{G}$ iff the following two conditions hold: (1) $\mathcal{R}$ is consistent and (2) for each relationship $R(G_1, G_2)$ with $G_1, G_2 \in \mathcal{G}$, either $R(G_1, G_2) \in \mathcal{R}$ or $\neg R(G_1, G_2) \in \mathcal{R}$.*

Let $\mathcal{R}$ be a maximally consistent set of relationships between granularities. With $G^{\mathcal{R}}$ we denote the set of labels, representing granularities, occurring in $\mathcal{R}$.

We now prove that each given consistent starting set of relationships can be extended to a maximally consistent set of relationships by using the proposed inference system.

Lemma 6.7. *Each set $\mathcal{R}$ of relationships between granularities can be extended to $\mathcal{R}^*$, a maximally consistent set with respect to $G^{\mathcal{R}}$.*

Proof. Let $r_1, r_2, \ldots$ be an enumeration of all possible relationships, and their negation, over $G^{\mathcal{R}}$. We iteratively build a sequence of consistent sets of relationships by defining $\mathcal{R}_0 = \mathcal{R}$ and

$$\mathcal{R}_{i+1} = \begin{cases} \mathcal{R}_i & \text{if } \mathcal{R}_i \cup \{r_{i+1}\} \text{ is inconsistent} \\ \mathcal{R}_i \cup \{r_{i+1}\} & \text{if } \mathcal{R}_i \cup \{r_{i+1}\} \text{ is consistent} \end{cases}$$

We define $\mathcal{R}^* = \bigcup_{i \geq 0} \mathcal{R}_i$. Now we prove that $\mathcal{R}^*$ is maximally consistent.

1. First we prove consistency. Suppose that $\mathcal{R}^*$ is inconsistent, thus there exists i such that R_{i-1} is consistent while R_i is inconsistent. Of course, it is not possible that R_i is inconsistent since it has been built from R_{i-1} adding r_i only if it remain consistent. Thus, $\mathcal{R}^*$ is consistent.
2. We prove now the maximality. Suppose $\mathcal{R}^*$ is not maximal, thus there exists $R(G_1, G_2)$ such that $R(G_1, G_2) \notin \mathcal{R}^*$ and $\neg R(G_1, G_2) \notin \mathcal{R}^*$. There exist i and j such that $R(G_1, G_2) = r_i$ and $\neg R(G_1, G_2) = r_j$. Let us suppose that $i < j$ (the other case is symmetric). Since $R(G_1, G_2) \notin \mathcal{R}^*$ we know that $\mathcal{R}_{i-1} \cup \{r_i\}$ is inconsistent, thus also $\mathcal{R}_{j-1} \cup \{r_i\}$ it is (since $i < j$, $\mathcal{R}_{j-1}$ includes at least $\mathcal{R}_{i-1}$). By the Proposition 6.5 we can conclude that $\mathcal{R}_{j-1} \cup \{r_j\}$ must be consistent and thus $r_j = \neg R(G_1, G_2) \in \mathcal{R}^*$ (contradiction). □

We used our inference system for obtaining a maximally consistent set of relationships. We now define some related notions and we prove some properties of maximally consistent set of relationships. In particular, we define what a model for a set of relationships is. We remember that a model can be considered as an example of granularities satisfying all relationships on which the model has been defined.

Definition 6.8. *Let $\mathcal{R}$ be a maximally consistent set of relationships between granularities. We define the binary relationship $\equiv^{\mathcal{R}}$ over $G^{\mathcal{R}}$ such that for each $G_1, G_2 \in G^{\mathcal{R}}$, $G_1 \equiv^{\mathcal{R}} G_2$ iff $G_1 \approx G_2 \in \mathcal{R}$.*

Proposition 6.9. *Given $\mathcal{R}$ a maximally consistent set of relationships between granularities, $\equiv^{\mathcal{R}}$ is an equivalence relation.*

Proof. It is trivial by the rules stating reflexivity, symmetry, transitivity, and monotonicity of the equivalence relationship. □

In the following we will use the notation $[G]^{\mathcal{R}}$ to indicate the equivalence class containing the label G, i.e., $[G]^{\mathcal{R}} = \{H \mid G \equiv^{\mathcal{R}} H\}$.

Definition 6.10. *Let $\mathcal{R}$ be a maximally consistent set of relationships between granularities. The* canonical model *$\mathcal{M} = (\mathcal{W}, \Sigma)$ is such that $\mathcal{W} = \{[G]^{\mathcal{R}} \mid G \in G^{\mathcal{R}}\}$ and $\Sigma = \{R(G_1, G_2) \in \mathcal{R}\}$. We define the* canonical interpretation *$\lambda : G^{\mathcal{R}} \to \mathcal{W}$ such that $\lambda(G) = [G]^{\mathcal{R}}$ for each $G \in G^{\mathcal{R}}$.*

We now prove that the model of a set of relationships satisfies all properties of relationships we defined in our framework (e.g., reflexivity of $GroupsInto$, $FinerThan$, $SubGranularity$, $Partition$, and $CoveredBy$).

Proposition 6.11. *Given $\mathcal{R}$ a maximally consistent set of relationships, its canonical model $\mathcal{M}$ is a Kripke model for our inference system.*

Proof. We have to prove that the model respects properties stated in the semantics of the relationships between spatial granularities. We will show only some cases, the other ones are similar.

(Refl) Suppose there exists a world $\mathcal{W}_1 \in \mathcal{W}$ such that $R(\mathcal{W}_1, \mathcal{W}_1) \notin \Sigma$ (with $R \in \{GI, FT, SG, P, CB\}$). Thus, there exists a label $G_1 \in G^{\mathcal{R}}$ such that $\lambda(G_1) = \mathcal{W}_1$ and $R(G_1, G_1) \notin \mathcal{R}$. By the maximality of $\mathcal{R}$, we know that $\neg R(G_1, G_1) \in \mathcal{R}$. This leads to the inconsistency of $\mathcal{R}$, given that in $\mathcal{R}$ we can derive $R(G_1, G_1)$ and then $\bot$.

(Antirefl) Suppose there exists a world $\mathcal{W}_1 \in \mathcal{W}$ such that $D(\mathcal{W}_1, \mathcal{W}_1) \in \Sigma$ or $O(\mathcal{W}_1, \mathcal{W}_1) \in \Sigma$. Thus, there exists a label $G_1 \in G^{\mathcal{R}}$ such that $\lambda(G_1) = \mathcal{W}_1$ and $D(G_1, G_1) \in \mathcal{R}$ or $O(G_1, G_1) \in \mathcal{R}$. This leads to the inconsistency of $\mathcal{R}$, given that in $\mathcal{R}$ we can derive $\neg D(G_1, G_1)$ and $\neg O(G_1, G_1)$. □

Now we prove two lemmas linking the inference of a relationship to its validity in a model.

Lemma 6.12. *Let $\mathcal{R}$ be a maximally consistent set of relationships between granularities, $\mathcal{R} \vdash R(G_1, G_2)$ iff $R(G_1, G_2) \in \mathcal{R}$.*

Proof. ($\Leftarrow$) If $R(G_1, G_2) \in \mathcal{R}$ then trivially $\mathcal{R} \vdash R(G_1, G_2)$.
($\Rightarrow$) Suppose $R(G_1, G_2) \notin \mathcal{R}$, thus, by the maximality of $\mathcal{R}$, $\neg R(G_1, G_2) \in \mathcal{R}$ and $\mathcal{R} \vdash \neg R(G_1, G_2)$. This leads to the inconsistency of $\mathcal{R}$ since, by hypothesis, $\mathcal{R} \vdash R(G_1, G_2)$. Contradiction. □

Lemma 6.13. *Let $\mathcal{R}$ be a maximally consistent set of relationships between granularities, $\mathcal{M}$ its canonical model, and λ the canonical interpretation. Then, $R(G_1, G_2) \in \mathcal{R}$ iff $\mathcal{R} \vDash^{\mathcal{M},\lambda} R(G_1, G_2)$.*
Proof. ($\Rightarrow$) if $R(G_1, G_2) \in \mathcal{R}$ and $\vDash^{\mathcal{M},\lambda} \mathcal{R}$ then trivially $\vDash^{\mathcal{M},\lambda} R(G_1, G_2)$.
($\Leftarrow$) By hypothesis $\mathcal{R} \vDash^{\mathcal{M},\lambda} R(G_1, G_2)$. Suppose $R(G_1, G_2) \notin \mathcal{R}$. By the maximality of $\mathcal{R}$, $\neg R(G_1, G_2) \in \mathcal{R}$, hence $\mathcal{R} \vDash^{\mathcal{M},\lambda} \neg R(G_1, G_2)$, and then $\mathcal{R} \vDash^{\mathcal{M},\lambda} \bot$ (contradiction). □

Finally, we prove the completeness of the inference system by contraposition. We prove that if the system does not infer a relationship it is because it does not hold definitely, i.e., there exists a model where this relationship is not valid.

Theorem 6.14 (Completeness). *The inference system for relationships between spatial granularities is complete, i.e., if $\mathcal{R} \nvdash R(G_1, G_2)$ then there exist a model $\mathcal{M}$ and an interpretation λ such that $\mathcal{R} \nvDash^{\mathcal{M},\lambda} R(G_1, G_2)$.*
Proof. If $\mathcal{R} \nvdash R(G_1, G_2)$ then $\mathcal{R} \cup \{\neg R(G_1, G_2)\}$ is consistent, otherwise $\mathcal{R} \cup \{\neg R(G_1, G_2)\} \vdash \bot$ and hence $\mathcal{R} \vdash R(G_1, G_2)$. $\mathcal{R} \cup \{\neg R(G_1, G_2)\}$ can be extended to a maximally consistent set $\mathcal{R}^*$. Let $\mathcal{M}$ be its canonical model and λ its canonical interpretation. $\mathcal{R}^* \vDash^{\mathcal{M},\lambda} \neg R(G_1, G_2)$ thus $\mathcal{R}^* \nvDash^{\mathcal{M},\lambda} R(G_1, G_2)$. Thus we can conclude that $\mathcal{R} \nvDash^{\mathcal{M},\lambda} R(G_1, G_2)$. □

6.7 Example

To better understand the proposed system, let us consider an example about spatial granularities. Let A, B, and C three spatial granularities. About them we know only that A is a subgranularity of B and groups into C, and that B is finer than C. By just using these three information the inference system deduces, by applying the concatenation rule to $SubGranularity(A, B)$ and $FinerThan(B, C)$ that also A is finer than C. Thus, since A groups into and is finer than C, A partitions C. Moreover, by applying rules reported in Tables B.1 and B.2 the inference system deduces also that the granularities are all covered by each other, i.e., they have the same image. Finally, applying again rules in Table B.2 to relationships $SubGranularity(A, B)$ and $CoveredBy(B, A)$, the inference system discovers that A and B are equivalent, i.e., they have the same granules eventually with different labels. Thus, of course, also B partitions C.

6.8 Rules for temporal and spatio-temporal relationships

In the previous section we presented the inference system for spatial granularities and we proved its soundness and completeness. Notions and proofs about the inference system for temporal granularities are very similar to the spatial ones and can be obtained from them just knowing that in the temporal case:

- *CoveredBy*, *Disjoint*, and *Overlap* relationships are not considered;
- the *GroupsPeriodicallyInto* relationship is added, but it is just a special case of the *GroupsInto*;
- the equivalence relationship is replaced by $ShiftEquivalent$.

Table B.8 describes which relationships between G and H can be inferred from a relationship $R(G, H)$. Rules in Table B.9 infer relationships between H and G starting from $R(G, H)$. Table B.10 contains rules inferring from a pair of relationships $R_1(G, H)$ and $R_2(G, H)$. Finally, Table B.11 represents the composition rules for relationships between temporal granularities. Finally, tables B.12 and B.13 infer relationships knowing which operations have been used to define temporal granularities.

On the other hand, the spatio-temporal case is slightly different thus we briefly detail this case.

A spatio-temporal granularity represents the evolution over time points (aggregated by a temporal granularity) of spatial granularities. Thus, the inference system for spatio-temporal granularities has to manage both temporal and spatial granularities and adds some new ad-hoc rules over spatio-temporal granularities. For the same reason, a model for spatio-temporal granularities contains spatial, temporal, and spatio-temporal granularities; thus, its semantics extends the semantics of spatial and temporal granularities models and add the new constraints depicted in figures A.4 and A.5.

We remind that a spatio-temporal relationship is made up of four parts:

- a temporal quantifier Q composed by a quantification on time granules and one on time points;
- the spatial relationship R that is temporally quantified by Q;
- two spatio-temporal granularities compared by the relationship.

Rules we studied answer to the same questions used for spatial and temporal granularities (see Section 6.3). As for the other inference systems, also in this case each rule has a premise that can be a conjunction of spatial, temporal, and spatio-temporal relationships, and one conclusion that is a spatio-temporal relationship.

These rules specify which quantifier can be inferred in the conclusion relationship while the spatial relationship is obtained applying the inference system for the spatial granularities.

Note that, premises of spatio-temporal rules include also the temporal relationship between the temporal granularities of the involved spatio-temporal granularities, because the quantifier in the conclusion relationship depends also by it.

Tables B.14 and B.15 contain rules inferring relationships between stG and stH starting from a relationship $QR(stG, stH)$ and the relationships between the temporal granularities involved in stG and stH. Rules in Table B.16 infer relationships between stH and stG starting from a relationship $QR(stG, stH)$. Tables from B.18 to B.26 represent concatenation table for spatio-temporal relationships.

For example, from Table B.16 we can conclude that from $\forall\exists R(stG, stH)$ we may infer $\exists\exists R'(stH, stG)$ if the temporal granularity of stG is a subgranularity of the one of stH, where R' is such that in the spatial inference system $R(G, H) \vdash R'(H, G)$.

As in spatial and temporal inference systems, the spatio-temporal system needs some auxiliary rules. Table B.27 describes when it is possible to infer $\perp$, i.e., a contradiction. Temporal quantification introduces uncertainty (e.g., in $\exists\exists GI(G,H)$ we do not know the exact time point in which $GroupsInto$ relationship hold) and this uncertainty influences inference rules. For example, from $\exists\forall R(G,H)$ and $\exists\forall\neg R(G,H)$ we cannot infer $\perp$ since the two considered temporal granules may be different. But we can do that from $\forall\forall R(G,H)$ and $\exists\exists\neg R(G,H)$. Table B.27 shows the quantifiers combinations that allow the system to infer a contradiction.

Considering the RAA rule for spatio-temporal granularities, its trivial formulation would be

$$\frac{\begin{array}{c} QR(stG,stH) \\ \vdots \\ \perp \end{array}}{\neg QR(stG,stH)}$$

However, since our framework for spatio-temporal granularities does not allow to put a negation before the quantifiers, but only to quantify a negation of a relationship, we always rewrite $\neg Q$ by using the usual translations for which $\forall \equiv \neg\exists\neg$ and $\exists \equiv \neg\forall\neg$. Table B.28 summarizes the RAA rule for spatio-temporal granularities.

Since spatio-temporal relationships are a temporally quantified version of spatial relationships, we remind that in any case the inference system from any relationship $QR(stG,stH)$ can infer $QR'(stG,stH)$ where R' is such that $R(G,H) \vdash R'(G,H)$ in the spatial inference system.

We say that two spatio-temporal granularities are equivalent, $stG \approx stH$ iff they are based on the same temporal granularity and $\forall\forall \approx (stG,stH))$. Note that equivalence of spatial granularities can be treated like all other spatial relationships, thus reflexivity, symmetry, and transitivity in the spatio-temporal case follow from spatial and spatio-temporal rules. Thus, we have just to specify the monotonicity rules:

- $QR(stG_1,stG_2) \wedge stG_1 \approx stG_3 \vdash QR(stG_3,stG_2)$
- $QR(stG_1,stG_2) \wedge stG_2 \approx stG_3 \vdash QR(stG_1,stG_3)$

Considering, changes and considerations described in this section, soundness and completeness proofs for spatio-temporal inference system is similar to those for spatial and temporal inference systems.

7

Using granularities for querying a spatio-temporal psychiatric database

After the two previous chapters, where we defined the theoretical notions of spatial and spatio-temporal granularities and their frameworks, in this chapter we continue toward the application of granularities for the management of spatio-temporal data in a clinical database. Since, in general, our goal is to use granularities for qualifying, enriching, and retrieving spatio-temporal data in relational DBMSs, we need in first place to represent granularities in relational DBMSs. For this reason, we design a database for temporal, spatial, and spatio-temporal granularities. This database contains all the information about granularities that are also contained in their theoretical representation: the definitions of the granularities, including their granules and, eventually, the operations used to create them, the relationships between the granularities, and, considering spatio-temporal granularities, the history of spatial granularities and their granules. This database will be integrated with a spatio-temporal clinical database and will be used to provide spatio-temporal clinical data with a qualification based on granularities.

We show how this database for granularities can be integrated with a spatio-temporal database for qualifying and retrieve data based on granularities. For this purpose, we design the PCR database for psychiatric data we introduced in Chapter 3 and we integrate it with the database for granularities, showing how psychiatric data may be enriched and qualified by using granularities. The designed PCR database store all the information about psychiatric patients that had contacts with the Verona Community-based Psychiatric Service (CPS). These data include personal data, that may evolve in time and which history has to be kept, and residence and domicile, that correspond to spatial locations that may be used to geographically analyze patients' data. Moreover, PCR contains all contacts occurred in last 30 years between patients and the psychiatric service. PCR includes both clinical and statistical information about contacts, that are also temporally and spatially qualified. As we mentioned in Chapter 3 the PCR database is currently used by the personnel of the Verona CPS for clinical, administrative, and research purposes. Thus, the analysis of its data also exploiting spatio-temporal information is an actual need for the personnel of the Verona CPS.

We use this psychiatric database enriched with granularities for introducing a spatio-temporal query language dealing with data qualified with granularities. The proposed query language is based on SQL [122] and T4SQL [57] and allows one to

perform spatio-temporal selection, join, and grouping, specifying how the system has to manage temporal, spatial, and spatio-temporal dimensions. For this purpose the query language allows the user to specify a semantics for each temporal, spatial, and spatio-temporal dimension involved in a query. Moreover, new constructs have been proposed in order to extend usual query language features with new capabilities based on granularities. The query language is presented through some example queries on the PCR database we designed. We see how data about patient and their contacts can be retrieved with respect to spatio-temporal constraints based on granularities. Several queries are proposed both for illustrating the query language capabilities and for showing how clinical spatio-temporal data qualified with granularities can be exploited.

This chapter is organized as follow. In the next section we present main related work about the conceptual modeling of spatio-temporal database with regard to granularities. In Section 7.2 we design our database for granularities. We also show some example queries for retrieving information about granularities from the proposed database, and we explain how granularities can be used for qualifying data in an existing spatio-temporal database. In Section 7.4 we present our spatio-temporal query language dealing with data qualified with granularities. The query language is presented through some example queries on the PCR database we introduced in Chapter 3.

7.1 Related work

In this section we discuss the main related work about the conceptual modeling of spatio-temporal databases with regard to granularities.

In Sections 4.1 and 5.1 we discussed main proposals in literature about the definition of spatial and spatio-temporal granularities. Among all the discussed proposals, only the proposals by Camossi et al. [34,35,37] deal with the implementation of granularities in real DBMSs by extending the ODMG model [38, 155]. As a matter of facts, their approach allows one to define their definition of granularities in ODBMSs that implement the ODMG model. Conversely, all the other discussed proposals deal only with formal and theoretical definitions of spatio-temporal granularities.

On the other hand, some authors proposed conceptual models for modeling spatio-temporal databases [5, 100, 129, 130, 162, 205, 213]. Tryfona and Jensen [205] propose to extend usual conceptual models (e.g., the Entity-Relationship model) with new constructs for representing temporal, spatial, and spatio-temporal information. New constructs allow one to represent entities, attributes, and relationships enriched with temporal, spatial, and spatio-temporal meanings. The type of these new constructs are declared by an annotation in the symbol of the constructs themselves. Considering entities, authors introduce new notations for associating a temporal and/or a spatial qualification to entities. Temporal entities can be associated with temporal dimensions, including "existence time", "valid time", "transaction time", and "bitemporal time". On the other hand, spatial entities represent objects with an associated spatial position. This position can be a point, a line, a region, or a generic geometry. Thus, note that there is not a parallelism between

temporal entities and spatial entities. The first ones use temporal dimensions (i.e., a temporal information associated to a data with different semantics) while the second ones use a spatial information without a specified semantics eventually with different data types. Spatio-temporal entities are those entities that have both a temporal dimension and a spatial position.

Similarly, annotation can be added to attributes for associating them to a temporal dimension and/or a spatial location. Furthermore, new constructs have been added for representing temporal, spatial, and spatio-temporal relationships. Relationships show again the different approach to temporal and spatial information. Temporal relationships are relationships with an associated temporal dimension and allow one to capture the changes in the relationships with respect to that dimension. Thus, temporal relationships can relate also non-temporal entities. On the other hand, spatial relationships represent associations among the geometries of the involved entities. Thus, only spatial entities can be related by a spatial relationship.

Finally, Tryfona and Jensen show how new proposed ER constructs can be represented in the classic ER model.

A similar approach based on annotations has been used in [53, 100] for defining TimeER, a temporal extension of the Extended ER (EER) model described by Elmasri and Navathe [76]. In TimeER classical constructs from ER and EER models are maintained with their usual meaning, while new notations are added for modeling time-varying information. In particular, entities, attributes, and relations in the ER schema can be annotated with a label representing the temporal dimension associated with the considered database objects. TimeER allows one to model four types of temporal dimensions, namely, valid time, transaction time, lifespan (i.e., the time over which a database object is defined), and user-defined time (i.e., an uninterpreted attribute domain of date and time with no special query language support). Database objects may be annotated with labels indicating the temporal dimensions that will be supported by the system on these objects, e.g., `VT` indicates valid time support, `LS` indicates lifespan support, and `TT` indicates transaction time support.

An annotated model has been proposed also by Khatri et al. [129, 130]. They extend the Unifying Semantic Model (USM) [170] using an annotation-based approach. This solution, called ST-USM, does not need to define new datatypes and structures. USM is an extended version of the Entity-Relationship model. They model interested reality using USM, then they add annotations to entities in the schema. Spatio-temporal annotations are composed by a temporal annotation and a spatial one separated by a double forward slash (//). Temporal annotations allow one to specify valid and/or transaction time, each one with an associated (possibly different) granularity. On the other hand, spatial annotations specify spatial geometries and spatial granularities expressing them. Thus, with respect to proposals we have described before, Khatri et al. introduce granularities in conceptual models. For this purpose, they use the definition of temporal granularity proposed by Bettini et al. [23,27], as we do here in this dissertation. For spatial granularities, authors extend the Worboys approach [226, 227] and in their proposal a spatial granularity may represent any partition of a spatial domain.

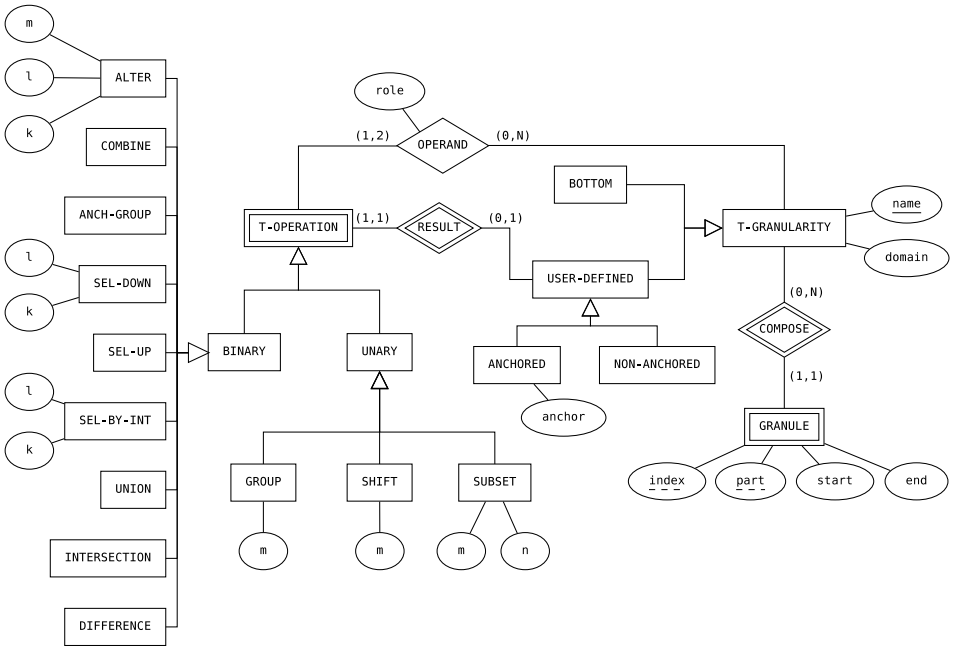


Fig. 7.1: Conceptual schema for temporal granularities

7.2 Representing granularities in relational DBMSs: STGran

In order to qualify information recorded in relational databases, we need to represent in RDBMSs (Relational DataBase Management Systems) also granularities. In this section, we conceptually design a database able to represent temporal, spatial, and spatio-temporal granularities. The design is reported by using the Enhanced Entity-Relationship model [76].

7.2.1 Conceptual modeling

The database can be divided in three intersecting parts. The three parts model temporal granularities, spatial granularities, and spatio-temporal granularities, respectively. These parts are integrated through entities shared by them.

Considering only temporal and spatial granularities, our conceptual schema can represent them independently. They are not enforced to participate in a spatio-temporal granularity. Hence, our proposal is useful also in order to represent individual temporal and spatial granularities.

Fig. 7.1 depicts the ER diagram representing temporal granularities. Temporal granularities are mainly represented by using the two entities `T-Granularity` and `Granule`. The former contains general information about granularities (e.g., the name). The later is a weak entity representing granules composing a granularity. The definition of temporal granularities [27] allows granules to have holes, i.e., to be divided in disconnected parts that, however, have to be considered all together. In order to represent time granules with holes, the `Granule` entity actually contains

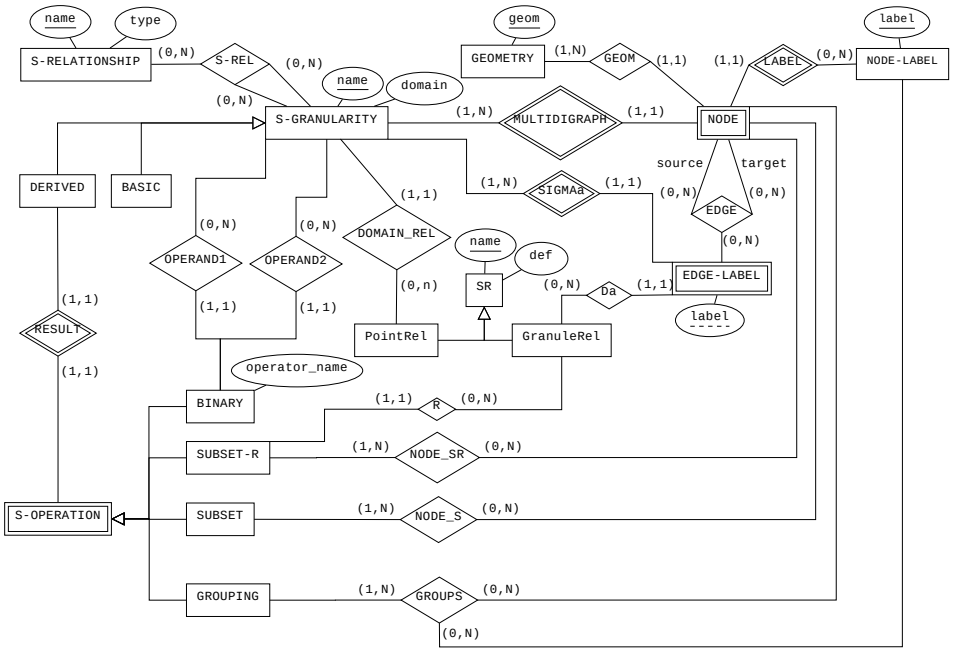


Fig. 7.2: Conceptual schema for spatial granularities

information about parts of granules instead of granules. Each part of a temporal granule has a start point (included in the granule) and an end point (not included in the granule). Parts are identified by the `index` of the granule they belong to and by the `part` index that numbers the part inside the granule. Thus, all entity instances with the same `index` value and belonging to the same granularity represent all parts of the `index`-th granule. In order to keep the database in a consistent state, some constraints must be imposed to ensure that the order induced by granule attributes `(index,part)` must be the same of the time points belonging to them (i.e., the first property of a temporal granularity [27]), and that parts of granules are disjoint.

Our database schema allows us to represent both "bottom" granularities (i.e., system defined granularities using which user can define new granularities, e.g., `Seconds`) and "user-defined" granularities which can be obtained by applying operations to other granularities (e.g., `Days` $= Group_{86400}$(`Seconds`) [154]). Moreover, user-defined granularities can be divided in anchored ones and non-anchored ones [54].

The anchor is the greatest lower bound of the set of time domain elements corresponding to the origin of the granularity [23]. In other words, the anchor represents the time point from which we start to count instants in the granularity. Representing non-anchored granularities [54] we permit to represent sets of granularities. If an user-defined granularity (e.g., `Days`) is not anchored, the system knows only the "shape" of its granules but they cannot be calculated by the system. For example, when we define `Days` as the result of a grouping operation

based on `Seconds`, the system knows how many seconds compose a day but does not know exactly what seconds because it does not know what is the anchor and then the starting point of the granularity. Subsequently, a non-anchored granularity can be anchored specifying the anchor (e.g., the first second of 2009) obtaining a particular granularity (e.g., `Days2009`) belonging to the family whose granules start from the specified anchor. To each different anchor corresponds a different granularity in the set.

Granularities may be either result or operand of operations. Operations represented are those defined in the calendar algebra [154]. Each operation stored in `T-Operation` has exactly one resulting granularity that identifies the operation itself and one or two operand granularities (it depends whether the operation is unary or binary). Moreover, depending by the applied operator, operations may have also numerical parameters (from zero to three parameters). Operations allow the user to define a granularity using already defined ones. As for granules, since a granularity is defined by the operation defining it rather than its granules, the user can delay the calculation of granules to the moment they will be really needed in order to evaluate a query. In other words, the system allows to store both the definition of a temporal granularity specifying the operation to apply to calculate the granularity itself and (some of) the granules in the granularity. For example, a user may define `Days2009` as the result of a grouping operation applied to `Hours` and only subsequently he may calculate some its granules, e.g., granules corresponding to days in May.

Fig. 7.2 depicts the conceptual schema for a database representing spatial granularity. The main entity `S-Granularity` contains main information about spatial granularities. Similarly to the temporal ones, spatial granularities can be divided into basic ones (directly introduced in the system, using geometrical data) and derived ones (resulting from operations applied to already defined granularities). Each granularity is represented by its multidigraph. Thus, a granularity is made up of some nodes (represented by the weak entity `Node`) identified by the name of the granularity and the node label (represented in the entity `Node-Label`). Moreover, each node has a geometry representing its corresponding spatial extent. Each edge of the multidigraph is represented by a ternary relationship defined by the source and the target nodes and the edge label. Each edge label corresponds, through the relation `Da` [19], to a spatial relation between granules contained in the `GranuleRel` entity. Since this correspondence may be different in different granularities, edge labels are identified by their name and by the granularity they belong to. Spatial relationships between points are represented through the `PointRel` entity. These relationships can be associated to spatial granularities in order to define a default relationship for comparing and ordering points in spatial granularities. Entities `GranuleRel` and `PointRel` totally partition in a Is-A hierarchy the `SR` entity.

In the schema we also explicitly represent relationships existing between spatial granularities (e.g., *FinerThan*, *Subgranularity*) by the `Spatial-Rel` entity and the `S-Rel` relation. These relationships are not determined when a new granularity is defined but the first time the system or the user need to know them to evaluate a query. In this way the system calculates them only if and when they are needed, and only once.

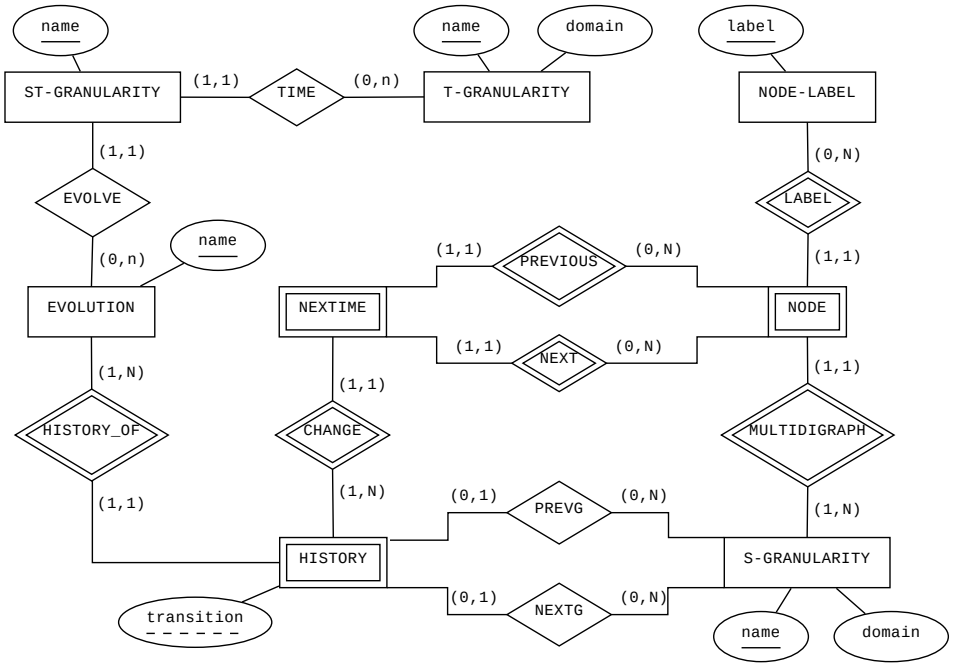


Fig. 7.3: Conceptual schema for spatio-temporal granularities

Each derived granularity is the result of an operation (e.g., *Provinces* may be the result of a grouping operation applied to *Municipalities*). The `Operation` entity is represented by using an Is-A hierarchy which subclasses represent the types of possible operations and are linked with entities representing the parameters of the operations (e.g., `Node` and `Node-Label`). All binary operations require only granularities as parameters and are represented in the `Binary` entity with an additional attribute representing the operator name (e.g., "Combining" or "SelectInside").

Fig. 7.3 shows the conceptual schema of the database representing spatio-temporal granularities. We note that in this schema entities `T-Granularity` and `S-Granularity` are shared with schemata in Fig. 7.1 and 7.2. These two entities are the links between the schemata. A spatio-temporal granularity is related to exactly one temporal granularity and one evolution. The theoretical definition of spatial evolutions [19] requires that a spatial granularity is associated to each time point. This point-based representation is here replaced by an interval-based representation. Thus, spatial granularities are associated to time intervals during which they are valid. This change is based on the observation that, in common applications, spatial granularities do not change every instant, and thus the interval-based representation is more efficient than the point-based one. It represents only relevant information, i.e., the granularities actually recorded without useless repetitions. For representing these valid time intervals we do not represent their start and end times, but the time points in which transitions between two spatial granularities happen. Entity `History` represents these transitions. Moreover, in order to allow one to represent lacks in information about valid time of granularities (i.e., granu-

larity A is valid until t_1, granularity B is valid since t_2, with $t_2 > t_1$, while between t_1 and t_2 there are no information about which granularity is valid) or holes inside valid time of a spatial granularity (i.e., a granularity A is valid from t_1 to t_2 but not between t_3 and t_4 with $t_1 < t_3 < t_4 < t_2$), we represent both the end (`prev_t`) of a valid time interval and the start (`next_t`) of the following valid time interval. When these two time points collide, it means that there is no holes between valid time intervals. Relations `PrevG` and `NextG` represent the spatial granularity valid until `prev_t` and the granularity valid since `next_t`, respectively.

Representing the transition time rather than the valid time we obtain also a better representation of $nextTime$. The mapping $nextTime$ represents the history of single granules inside an evolution, i.e., transitions from a granule to its successors. These transition instants between granules are the same of those between granularities stored in `History`. The weak entity `Nextime` describes for each transition point the previously valid granule and the next valid one. For each transition, we may have a different `Nextime` instance for each granule.

7.2.2 Logical design

The second phase of the database design process is the logical design and it requires to arrange data into a logical structure [76]. This phase starts from the conceptual model and translates it to the logical one. Before the translation, we need to restructure the first model in order to simplify the translation and optimize the schema. The main objects affected by this restructuring phase are primary keys and Is-A hierarchies. Considering primary keys, we note that in order to optimize the database performance it is better to reduce the number of attributes participating to them. Hence, we substitute composed and non-integer keys with ad hoc ones (i.e., `id`) which have a smaller internal representation. Following similar considerations, we remove weak entities (their keys are inherently composed by at least two attributes). We translate them in strong entities substituting their keys (the weak one and the external one) with ad-hoc `id`s.

Obviously, these changes have not to affect the information requirements (e.g., functional dependencies) gathered and represented in the conceptual model. Hence, we have to define constraints in order to ensure that these requirements remain valid also after these changes and the mapping to the logical model. Considering the introduction of new `id` primary keys instead of non-numerical and weak keys, we impose constraints imposing that attributes participating to keys in the conceptual model have to be unique and not-null in the logical one (e.g., the pair `(label, sgranularity)` in table `Node`).

Considering Is-A hierarchies, we use several different (in some cases, non-standard) approaches for restructuring them. We transform Is-A hierarchies about `T-Granularity`, `S-Granularity`, and `SR` collapsing subclasses into superclasses. We introduce in `T-Granularity` a nullable attribute `anchor` to represent the anchor of anchored temporal granularity. In `SR` we introduce a new attribute `type` representing whether an instance represents a relation between points or granules. Moreover, we define a constraint about `SR` imposing that a spatial granularity have to be related through `Domain_Rel` with an instance of `SR` with attribute `type` set to "`point`" while an edge label have to be related with a relation with `type` set to "`granule`".

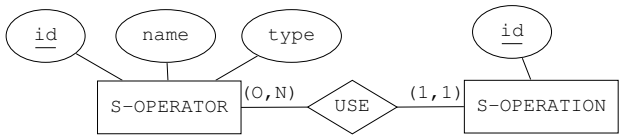


Fig. 7.4: Restructured conceptual schema for spatial operations

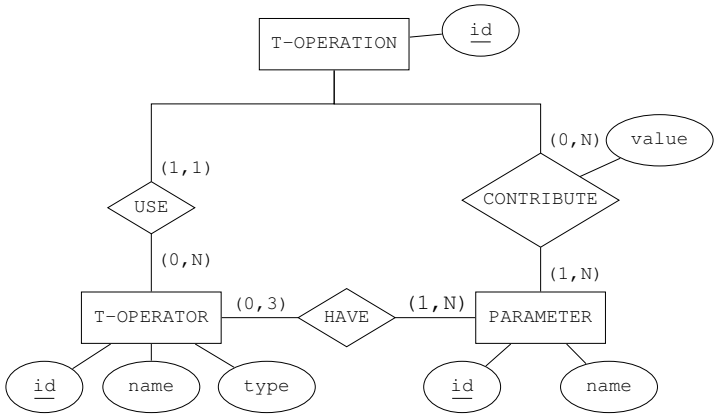


Fig. 7.5: Restructured conceptual schema for temporal operations

A similar approach has been used to transform the `S-Operation` hierarchy (see Fig. 7.4). We represent only the superclass and all relationships previously related to subclasses are now linked with the superclass. In superclass we would need a new attribute identifying the subclass to which one instance belongs and in particular the operator used by an operation. However, using a non-standard approach, we do not introduce this new attribute but a new entity representing spatial operators (`S-Operator`). Hence, in the new schema each spatial operation (i.e., each instance of `S-Operation`) is related to its operator through the relationship `Use`. Each operator is identified by an `id` and has a `name` and a `type` describing whether it is unary or binary. The same approach has been used restructuring `T-Operation` hierarchy. We keep information about operator and parameters used in each operation in two new entities `Operator` and `Parameter` (see Fig. 7.5).

We define some constraints to impose, both for temporal and spatial operations, that each operation instance uses exactly and only the parameters involved by its operator.

After these two kinds of changes have been performed, the translation from conceptual model to the logical one can be made through a well-known mapping algorithm [76].

7.2.3 Example query

In this section we show an example (reported by using SQL [76]) of query retrieving information about spatio-temporal granularities from the database we designed in the previous sections. For this purpose, Listing 7.1 shows a part of the

```
GRANULE(id, index, start, end, granularity)
T-GRANULARITY(id, name, domain, extentS, extentE, anchor)
ST-GRANULARITY(id, name, tgranularity, evolution)
EVOLUTION(id, name)
VALIDTIMEHISTORY(evolution, sgranularity, since, to)
S-GRANULARITY(id, name, domain, extent, domain_rel)
NODE(id, sgranularity, geom, label)
NODE-LABEL(id, label)
```

Listing 7.1: Logical design of a part of the schema

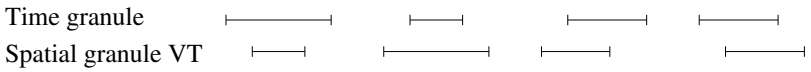


Fig. 7.6: The four possible intersections between valid times

logical design of the proposed database schema. In particular, only relations used in the proposed example are reported. As we mentioned before, the translation from conceptual model to the logical one has been obtained through the well-known mapping algorithm from the Entity-Relationship diagram to the relational schema [76].

Listing 7.2 reports the query needed to calculate, for a given spatio-temporal granularity $\$GivenSTG$, its spatio-temporal granules. We remember that a spatio-temporal granule is the set of the spatial granules, ordered with respect to time points, valid during a temporal granule. Hence, the query returns for each time granule completely included in a given interval $[\$GivenStart, \$GivenEnd]$: its index, the labels of spatial granules inside it and their valid time.

In order to calculate correct valid time of spatial granules, we have to intersect valid time of spatial granules with the span of time granules. Fig. 7.6 depicts the four different non-degenerate ways these two intervals can intersect each other. These four cases correspond to the following situations:

1. valid time of the time granule is included in the valid time of the spatial granule;
2. valid time of the spatial granule is included in the valid time of the time granule;
3. time granule starts before than the spatial granule and ends during the valid time of the spatial granule;
4. time granule stars during the valid time of the spatial granule and ends after than the spatial granule.

Thus, the query is constituted by four subquery merged by `UNION` operators. Each subquery calculates the intersection in one of the four previous cases. We completely report only the first subquery. In the other ones, since the `FROM` clause is equal to that of the first subquery, we omit it. The `FROM` clauses are constituted by a sequence of `JOIN` operators necessary to join together temporal granules and spatial granules.

```
SELECT Gr.INDEX AS tindex, Gr.start AS tstart, Gr.END AS tend,
       NL.label AS node
FROM ST-Granularity STG JOIN T-Granularity TG
     ON (STG.tgranularity = TG.id)
     JOIN Granule Gr ON (Gr.granularity = TG.id)
     JOIN Evolution E ON (STG.evolution = E.id)
     JOIN ValidTimeHistory VTH ON (VTH.evolution = E.id)
     JOIN S-Granularity SG ON (VTH.granularity = SG.id)
     JOIN Node N ON (N.sgranularity = SG.id)
     JOIN Node-Label NL ON (N.label = NL.id)
WHERE STG.name = $GivenSTG AND Gr.start > $GivenStart AND
      Gr.END < $GivenEnd AND Gr.start >= VTH.since AND
      Gr.END <= VTH.to
UNION
SELECT Gr.INDEX AS tindex, Gr.start AS tstart, VTH.to AS tend,
       N.label AS node
FROM TABLE_JOINS
WHERE STG.name = $GivenSTG AND Gr.start > $GivenStart AND
      Gr.END < $GivenEnd AND Gr.start >= VTH.since AND
      Gr.END > VTH.to AND Gr.start < VTH.to
UNION
SELECT Gr.INDEX AS tindex, VTH.since AS tstart, Gr.END AS tend,
       N.label AS node
FROM TABLE_JOINS
WHERE STG.name = $GivenSTG AND Gr.start > $GivenStart AND
      Gr.END < $GivenEnd AND Gr.start < VTH.since AND
      Gr.END <= VTH.to AND Gr.END > VTH.since
UNION
SELECT Gr.INDEX AS tindex, VTH.since AS tstart, VTH.to AS tend,
       N.label AS node
FROM TABLE_JOINS
WHERE STG.name = $GivenSTG AND Gr.start > $GivenStart AND
      Gr.END < $GivenEnd AND Gr.start < VTH.since AND
      Gr.END > VTH.to
```

Listing 7.2: Query to calculate spatio-temporal granules

The query uses, in the `FROM` clauses, a particular table not included in our schema proposal, `ValidTimeHistory`: as we detailed in Section 7.2.1, our database schema represents in entity `History` the transition between spatial granularities rather than their valid time. The query reported in Listing 7.3 shows how we may obtain `ValidTimeHistory` from table `History`.

Finally, the `WHERE` clauses restrict the selection to the given spatio-temporal granularity and time interval. Moreover, each clause imposes the conditions about span of temporal granules and valid time of spatial granularities representing one of the four intersection cases.

```
SELECT evolution, prevG AS granularity,
        NULL AS since, transition AS to
FROM history h1
WHERE transition = (SELECT MIN(h.transition)
                    FROM history h
                    WHERE h.prevG IS NOT NULL AND
                          h.evolution = h1.evolution)
      AND NOT EXISTS (SELECT h.transition
                  FROM history h
                  WHERE h.evolution = h1.evolution AND
                        h.transition < h1.transition)
UNION
SELECT h1.evolution AS evolution, h1.nextg AS granularity,
       h1.transition AS since, h2.transition AS to
FROM history h1 JOIN history h2
     ON (h1.evolution = h2.evolution AND h1.nextg = h2.prevg)
WHERE h2.transition = (SELECT MIN(h.transition)
                     FROM history h
                     WHERE h.transition > h1.transition AND
                           h.evolution = h1.evolution)
UNION
SELECT evolution, nextg AS granularity,
       transition AS since, NULL AS to
FROM history h1
WHERE transition = (SELECT MAX(h.transition)
                    FROM history h
                    WHERE h.nextG IS NOT NULL AND
                          h.evolution = h1.evolution) AND
      NOT EXISTS (SELECT h.transition
                  FROM history h
                  WHERE h.evolution = h1.evolution AND
                        h.transition > h1.transition)
```

Listing 7.3: Query to obtain valid time of spatial granularities in an evolution

7.2.4 Using STGran to qualify spatio-temporal data

As we mentioned in previous sections, granularities are useful in order to qualify and aggregate data, also during the querying phase. There are two ways to qualify and aggregate information by using granularities.

The first way needs to link directly information we are interested in to their corresponding granules in granularities (stored in the database described in Section 7.2). To do that we extend the existing database adding a reference (i.e., a foreign key) in each table containing data we want to qualify. This foreign key will refer to the granule (or node in the spatial case) in which data are located.

For example, if we consider a table containing data about malaria cases with the following schema:

```
MalariaCase(id, patient, physician)
```

we may add an attribute `municipality_granule` for spatially qualifying each case with the municipality where it has been surveyed.

Note that in this case, we may mean either that the qualified information is related to all the granule extent or that it is located in an unspecified point inside the granule.

We may add more references in order to qualify different information in the same table (e.g., in a table containing patients' personal data we can temporally qualify both birth and death dates) or to qualify the same information with several granularities.

The second way to aggregate data allows one to qualify data during the querying phase. In this case, we do not add a reference to granules with which we want to qualify information. We add a temporal and/or a spatial location to information in the database (e.g., the date or the geographical coordinates in which a malaria case has been surveyed). Then it is possible, in a query, to aggregate information by using granularities based on the temporal or spatial information we added. For example, supposing we added to each malaria case in the previous table the date in which it has been occurred:

```
MalariaCase(id, patient, physician, date)
```

we can query malaria cases aggregating dates with respect to a temporal granularity stored in the database (e.g., months).

We will see more examples of these two ways to qualify and aggregate data in Section 7.4.2.

7.3 ST4SQL: a spatio-temporal query language for dealing with granularities

In this section, we present our query language for dealing with spatio-temporal data and granularities. It is a spatio-temporal extension of T4SQL, the temporal query language proposed by Combi et al. [57] and discussed in Section 2.1.2. As T4SQL, ST4SQL is an SQL-based query language. It extends the temporal constructs in T4SQL to deal with temporal granularities and it adds new spatial and spatio-temporal constructs, similar to the temporal ones, for dealing also with spatial and spatio-temporal granularities. We now present the syntax and the semantics of ST4SQL.

7.3.1 The overall idea

As we already explained, one of the main contributions of T4SQL has been the introduction of four semantics for temporal queries. These semantics allows one to specify how the system has to manage temporal dimensions in order to evaluate queries. In particular, the proposed semantics for temporal querying are defined on time points. Thus, for example, the `SEQUENCED` semantics applied to valid time considers instant by instant the temporal domain and takes into account only those tuples whose valid time contains the considered time point. This approach allows one, for example, to query PCR (see Section 3.2) for retrieving diagnosis

of unemployed patients that are searching for a new job. Thus, the valid time of diagnoses must overlap the valid time of patients' employment status and this requirement is implicitly provided by the `SEQUENCED` semantics.

```
SEMANTICS SEQUENCED ON VALID
SELECT p.name, p.surname, d.description
FROM diagnosis AS d, patient_diagnosis AS pd, patient AS p,
     patient_employment AS pe, employment AS e
WHERE pd.diagnosis = d.id_code AND pd.patient = p.id_code AND
      pe.patient= p.id_code AND pe.employment = e.id_code AND
      e.description = 'unemployed searching for a new job'
```

However, we can consider semantics and similar queries also by using temporal granularities instead of time points. In this way, the system evaluates queries considering granules instead of time points and, thus, temporal dimensions associated to tuples are related to temporal granules. To do that, we can extend previous temporal semantics. In this case, each semantics requires the user to specify a time dimension (e.g., valid time) and a temporal granularity (e.g., months). Thus, semantics can be reformulated as follow:

- SEQUENCED(td,tG): the DBMS evaluates the submitted query only on those tuples of the relations referred in the `FROM` clause whose value for the given temporal dimension td intersects (i.e., overlaps, is contained, or contains) the same temporal granule of the temporal granularity tG.
- CURRENT(td,tG): specifying this semantics the DBMS evaluates the query only on those tuples whose value over td intersects the granule of tG containing the current date.
- NEXT(td,tG): this semantics considers only pairs of tuples belonging to the join of the tables specified in the `FROM` clause, that are related to the same entity and that are in two consecutive granules of tG, with respect to the temporal ordering.
- ATEMPORAL(td): it disables any support from the system, thus td is considered a classic attribute managed by the user and it does not require to specify a time granularity.

Semantics based on granularities allow one to "relax" T4SQL queries by relating tuples with respect to the granules of a temporal granularity instead than time points.

Similar definitions can be given also on spatial dimensions. Spatial semantics allow one to specify how the query engine must manage spatial dimensions associated to tuples. Similarly to the temporal ones, spatial semantics restrict the query evaluation only to those tuples whose value on a given spatial dimension meets some constraints. In particular, tuples are related to the granules of a given spatial granularity in order to select only some of them.

Semantics for spatial queries can be formulated as follow.

- SEQUENCED(sd,sG): the DBMS evaluates the submitted query selecting only those tuples of the relations referred in the `FROM` clause whose value for the given spatial dimension sd intersects the same spatial granule.
- CURRENT(sd,sG): specifying this semantics the DBMS evaluates the query only on those tuples whose value over sd intersects the granule of the spatial

granularity sG containing the current user's position (of course, it requires that the user's position is available or can be determined, for example by analyzing the user's IP address).

- NEXT(sd,sG,R): this semantics considers only those tuples related to the same entity and that are in two consecutive granules of sG with respect to the ordering (eventually non-total) defined by the spatial relationship R.
- SPACELESS(sd): it disables any support from the system, thus sd is considered a classic attribute managed by the user.

Note that for spatial semantics we consider only definitions based on granularities, thus we do not consider spatial semantics based on space points. This limitation is based on the consideration that joining only tuples spatially qualified exactly with the same point location has very little sense, especially considering that many errors and uncertainties can affect spatial surveys.

Considering spatio-temporal granularities, we note that they temporally qualify spatial granularities. Thus, they allow one to spatially aggregate data with respect to a spatial granularity whose definition changes over time. The value of tuples over a given temporal dimension is used to select the spatial granularity valid at the same time of considered tuples. Thus, a spatio-temporal semantics requires that both a temporal dimension and a spatial dimension are specified, we call this pair a spatio-temporal dimension.

In particular, a spatio-temporal granularity groups spatial granularities with respect to granules of a temporal granularity. Each tuple is spatially qualified by using spatial granularities valid during the time granules intersecting the value of the tuple over the given temporal dimension. Since this value may be an interval, and that (based on the definition of spatio-temporal granularity) during this interval the spatial granularity may evolve, several spatial granularities may be used to qualify each tuple. For this reason the user may specify an aggregate function to apply to spatial granularities in order to obtain just one spatial granularity among all granularities valid during the interval. Aggregate functions include, for example, `first` (that returns the spatial granularity valid at the first instant in the interval), `last` (that returns the spatial granularity valid at the last instant in the interval), and operations defined in Chapter 4 for calculating union, intersection, and difference of spatial granularities. If the aggregate function is not provided by the user, the system uses all spatial granularities valid in the interval, one by one, for querying the considered tuple, and all these combinations are used for obtaining the final resulting relation.

Since, spatio-temporal granularities enclose in one structure both a temporal and a spatial granularity, they can be used for temporal and spatial querying at the same time. For this purpose the user specifies a semantics on both the temporal and the spatial dimension making up the given spatio-temporal dimension. These temporal and spatial semantics are used as we have described before in this section, but in this case the temporal dimension is used also for selecting the spatial granularity valid at the same time of considered tuples.

7.3.2 The ST4SQL syntax

We remember that the main part of the syntax of T4SQL is the following [57]:

```
[SEMANTICS <sem> ON <dim> [TIMESLICE <ts_exp>]
        {, <sem> ON <dim> [TIMESLICE <ts_exp>]}
SELECT <sel_element_list>
       [, TGROUPING(<t_attr> [AS] <new_name>
                 {, <t_attr> [AS] <new_name>})]
FROM <tables>
WHERE <conditions>
WHEN <t_conditions>
GROUP BY <group_element_list>
HAVING <g_cond>
```

where:

- `<sem>` is the name of a semantics;
- `<dim>` is the name of a temporal dimension;
- `<ts_exp>` is a temporal expression resulting in a time point or an interval;
- `<t_attr>` is a temporal attribute;
- `<t_conditions>` is a set of selection conditions involving only temporal attributes.

The syntax of ST4SQL extends that of T4SQL (and that of SQL). Its (incomplete) BNF is as follows:

```
[SEMANTICS [<tsem> ON <tdim> [WITH TGRANULARITY <tG>]
           [THROUGH <attr>] [TIMESLICE <ts_exp>] {, ...},]
           [<ssem> ON <sdim> WITH SGRANULARITY <sG>
           [ORDERED BY <sr>] [THROUGH <attr>]
           [SPACESLICE <ss_exp>] {, ...},]
           [<tsem> ON <tdim> [TIMESLICE <ts_exp>] AND
           <ssem> ON <sdim> [SPACESLICE <ss_exp>]
           WITH STGRANULARITY <stG>
           [SIMPLIFY BY <s_aggr_funct>]]]
SELECT <sel_element_list> [WITH <exp> [AS] <dim> {, ...}]
       [, TGROUP(<t_attr>) [AS] <new_name> {, ...}]
       [, SGROUP(<s_attr>) [AS] <new_name> {, ...}]
[FROM <tables>]
[WHERE <conditions>]
[WHEN <t_conditions>]
[WHEREABOUTS <s_conditions>]
[GROUP BY <group_element_list>]
[HAVING <g_cond>]

<tsem> ::= ATEMPORAL | CURRENT | SEQUENCED | NEXT[(<duration>)]
<ssem> ::= SPACELESS | CURRENT | SEQUENCED | NEXT
<dim>  ::= <tdim> | <sdim>
<tdim> ::= VALID TIME | TRANSACTION | AVAILABILITY |
           INITIATING_ET | TERMINATING_ET
<sdim> ::= VALID SPACE
<group_element_list> ::= <group_element> {, <group_element>}
<group_element> ::= <attribute> | <temp_attr> ON <tG> |
                    <space_attr> ON <sG>
```

where:

- tG and sG represent a temporal and a spatial granularity, respectively;
- ts_exp and ss_exp represent expressions corresponding to a time interval and a spatial region, respectively;
- <sr> is the name of a spatial relationship between granules whose definition must be present in the SR relation of the proposed database for granularities.

The SEMANTICS clause allows one to specify the semantics to be applied for temporal, spatial, or spatio-temporal dimension. More semantics may be specified on different dimensions, while at most one semantics can be applied to a dimension, otherwise the query is considered to be not well-formed.

In particular, a semantics for a temporal dimension may be specified with the following syntax:

```
SEMANTICS <tsem> ON <tdim> [WITH TGRANULARITY <tG>]
          [THROUGH <attr>] [TIMESLICE <ts_exp>]
```

while, with a similar syntax, the user may specify a spatial semantics:

```
SEMANTICS <ssem> ON <sdim> WITH SGRANULARITY <sG>
           [ORDERED BY <sr>] [THROUGH <attr>]
           [SPACESLICE <ss_exp>]
```

and a spatio-temporal semantics:

```
SEMANTICS  <tsem> ON <tdim> [TIMESLICE <ts_exp>] AND
           <ssem> ON <sdim> [SPACESLICE <ss_exp>]
           WITH STGRANULARITY <stG>
           [SIMPLIFY BY <s_aggr_funct>]
```

The syntax of the SEMANTICS clause allows one to specify the parameters we introduced in the previous section. Thus, as we have already explained above, each semantics requires a dimension (<tdim> or <sdim>) on which it has to be applied. Moreover, optionally, a temporal semantics can be applied in relation to a temporal granularity <tG> that will be used to group and select the tuples on which the query has to be evaluated. Since spatial semantics are defined only on granularities, the WITH SGRANULARITY <sG> option is mandatory and <sG> is the name of the spatial granularity to be used.

The TIMESLICE and SPACESLICE options are optional and allow one to restrict the query evaluation only on those tuples whose value for the given temporal or spatial dimension intersects the value of <ts_exp> and <ss_exp> expressions, respectively. <ts_exp> must correspond to a time interval, when the WITH TGRANULARITY <tG> option is not specified, or to a list of labels of granules in the <tG> temporal granularity when the WITH TGRANULARITY <tG> option is present. Similarly, <ss_exp> is a list of labels of granules in the <sG> spatial granularity. Note that, the TIMESLICE and SPACESLICE options cannot be coupled with the CURRENT semantics.

The THROUGH <attr> option may be specified for both temporal and spatial NEXT semantics. The NEXT semantics requires that two instances of the FROM clause must be joined together in order to obtain pairs of tuples representing consecutive tuples corresponding to the same entity. Two tuples are considered to refer to

the same entity when they have same value for the key. The THROUGH <attr> option allows one to specify the attribute, instead of the key, the system has to use for identifying tuples referring to the same entity. Moreover, for the spatial NEXT semantics, the user has compulsorily to specify the spatial relationship to be used for ordering granules in <sG>. That can be done with the ORDERED BY <sr> option, where <sr> is the name of a spatial relationship between granules defined in the SR relation of the proposed database for granularities.

As we have already introduced before, the syntax for spatio-temporal semantics requires to specify a temporal semantics of a temporal dimension, a spatial semantics on a spatial dimension, and a spatio-temporal granularity <stG>. The SIMPLIFY BY <s_aggr_funct> option allows one to specify the aggregate function to apply to all spatial granularities valid during a temporal granule of the temporal granularity of <stG>.

Note that, when no semantics is specified by the user on a temporal or spatial dimension, the ATEMPORAL or SPACELESS semantics (depending by the dimension) is applied as default by the system on that dimension.

After the SEMANTICS clause, the query continues with the usual SELECT and FROM clauses. The SELECT clause specifies what attributes (or expression) have to be returned for each tuple in the resulting relation, while the FROM clause contains the list of relations (that, eventually, may be the result of a subquery) required for processing the query. The user can include the WITH option in the SELECT clause in order to force the inclusion of a dimension in the output relation. This option overwrites the dimension that would be included automatically by the system in the output relation. When the user does not specify how values on a dimension must be computed, the rule given by the semantics applied on that dimension is applied. In the SELECT and FROM clauses aliases for attributes and tables can be provided. However, they cannot be any of the language keywords. Besides the SQL keywords, in ST4SQL the following words are reserved: SEQUENCED, SEMANTICS, VALID, CURRENT, ATEMPORAL, NEXT, WITH, TGRANULARITY, SGRANULARITY, SPACELESS, STGRANULARITY, TIMESLICE, SPACESLICE, SIMPLIFY, AVAILABILITY, INITIATING_ET, TERMINATING_ET, TGROUP, SGROUP, THROUGH, WHEN, WHEREABOUTS.

The WHERE clause contains the selection and join conditions to apply in order to select the set of tuples on which the query must be evaluated. The WHEN and WHEREABOUTS clauses are similar to the WHERE one, but they are used to keep divided the conditions on classic attributes (in the WHERE clause) from the temporal conditions (in the WHEN clause) and the spatial conditions (in the WHEREABOUTS clause). Temporal and spatial conditions are conditions involving the value of the tuples on temporal and spatial dimensions, respectively.

We introduce also some spatio-temporal functions for accessing the value of tuples on temporal and spatial dimensions. These accessors can be used both for including the value of the tuples on a dimension in the output relation and for specifying any condition on the value of the tuples on a dimension. Given a tuple of a relation R, the value it takes over the temporal and the spatial dimensions can be recovered by using the following functions:

VALIDT(R) returns the valid time of the R tuple;

`TRANSACTIONT(R)` returns the transaction time of the `R` tuple;
`AVAILABLET(R)` returns the availability time of the `R` tuple;
`INITIATING_ET(R)` returns the initiating event time of the `R` tuple;
`TERMINATING_ET(R)` returns the terminating event time of the `R` tuple;
`VALIDS(R)` returns the valid space of the `R` tuple;

In Section 7.2.4 we said that tuples in a database can be qualified with spatio-temporal granularities in two alternative ways:

- indirect: tuples are associated to a temporal and/or spatial location that is qualified with granularities at query time;
- direct: tuples are associated directly to a granule in a temporal, spatial, or spatio-temporal granularity.

The previous accessors work in both cases. In the first case they return the location associated to the tuples, while in the second case they return the granule associated to the tuples.

As usual, the `GROUP BY` clause allows one to define groups of tuples on which aggregate functions may be applied. The `GROUP BY` clause contains the list of grouping attributes that are required to be equal in all tuples of a group. As T4SQL, also ST4SQL allows the user to group tuples with respect to the their value on a temporal or spatial dimension. To do that the user can use in the `GROUP BY` clause the spatio-temporal accessors we introduced above. For example, `GROUP BY VALIDT(t)` specifies that tuples have to be grouped with respect to the valid time of the `t` relation. Moreover, the accessors may be followed by the `ON <G>` option that specifies that the groups are not defined according to the value returned by the considered accessor but by the granules of the `<G>` granularity, whose type (temporal or spatial) must agree with the type of the accessor. Thus, for example, `GROUP BY VALIDT(t) ON <tG>` groups together tuples whose valid time intersect the same granule in `<tG>`. On the other hand, in order to include the value of groups on a dimension in the output relation, the user can use the `TGROUP` and `SGROUP` functions in the `SELECT` clause. These functions require as parameter the accessor used in the `GROUP BY` clause and return the value of the accessor if the `ON <G>` option has be not specified together the accessor in the `GROUP BY` or the index of the granule in `<G>` if the `ON <G>` option has be specified. Thus, for example:

```
SELECT TGROUP(VALIDT(t))
FROM t
GROUP BY VALIDT(t) ON <tG>
```

returns a relation where each tuple represents a group of tuples in `t`. Each group contains the tuple whose value over the valid time dimension intersects the same granule of `<tG>`. The identifier of the granule (specified by `TGROUP(VALIDT(t))`) used for grouping together the tuples is included the output relation.

Finally, the `HAVING` clause allows one to specify conditions on the groups. The `HAVING` clause provides a condition that has to applied to each group of tuples, only groups that meet such condition are retrieved in the output relation. As usual, the `HAVING` clause may include aggregate functions.

7.3.3 The ST4SQL semantics

All ST4SQL queries can be translated in equivalent SQL queries. Thus, it is possible to specify the semantics of ST4SQL with respect to the one of SQL. This can be done by showing how ST4SQL queries can be translated into SQL. Here we focus on main idea of this translation and ST4SQL semantics, while the complete semantics is reported in Appendix C.

The semantics of the accessors and other functions we introduced in the previous section about the syntax of ST4SQL has been already partially explained introducing them. More attention requires instead the `SEMANTICS` clause. As we have already explained, it allows one to specify how the temporal, spatial, and spatio-temporal dimensions have to be managed by the system for evaluating the query. In general, the clause is equivalent to some join and selection conditions that allow the system to select the right tuples on which the query have to be evaluated.

Let us briefly introduce the semantics of the `SEMANTICS` clause for some exemplifying cases. For example, let us consider the `ATEMPORAL` semantics. When it is specified on a temporal dimension (e.g., valid time), it disable any system support on the given dimension. Thus, the dimension associated to tuples is considered as usual atemporal attributes without any specific meaning. In this case the user has to manage these attributes when she needs. Of course, in this case the submitted query is translated into an equivalent SQL query without adding anything. Thus, given the following ST4SQL query:

```
SEMANTICS ATEMPORAL ON <tdim>
SELECT <sel_element_list>
FROM <tables>
WHERE <conditions>
```

its SQL equivalent query is:

```
SELECT <sel_element_list>
FROM <tables>
WHERE <conditions>
```

Together with the `ATEMPORAL` semantics the `TIMESLICE <ts_exp>` option can be specified. In this case `<ts_exp>` must represent a time interval (t_1, t_2). The meaning of the `TIMESLICE` option is to limit the query evaluation only to those tuples whose value for the given temporal dimension intersects the (t_1, t_2) interval. Since, no other relation between involved tables is required by the semantics, this constraint must be formulated separately for each table in the `FROM` clause. Thus, let

```
SEMANTICS ATEMPORAL ON <tdim> TIMESLICE <ts_exp>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

be the considered ST4SQL query. Then, it is equivalent to the following SQL query.

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions> AND intersect(VALIDT(T1),<ts_exp>) AND ...
      AND intersect(VALIDT(Tn),<ts_exp>)
```

where we supposed that `<tdim>` is `VALID TIME`. Otherwise, the correspondent accessor must be used in place of the `VALIDT` function. `intersect(i1,i2)` is a function we defined for the sake of simplicity. It checks whether the given time intervals intersects each other. It can be easily implemented using usual SQL comparison operators and pair accessors.

Finally, together with the `TIMESLICE` option, the `WITH TGRANULARITY <tG>` option can be specified in order to limit the query evaluation only to some granules of `<tG>`. In this case, `<ts_exp>` represents a pair `(gstart,gend)` representing a granule interval where `gstart` and `gend` are the indexes of the first and the last granule in `<tG>` in the interval, respectively. The two elements of the pair are accessed by using `<ts_exp>$start` and `<ts_exp>$end`, respectively. In order to apply the specified timeslice, the `FROM` clause has to be extended with a subquery for calculating the time interval corresponding to the granule interval. Moreover, the `WHERE` clause must include, for each table T_i that includes the temporal dimension `tdim` (e.g., valid time), the conditions to constrain the query evaluation only to the calculated interval. Thus, let

```
SEMANTICS ATEMPORAL ON <tdim> WITH TGRANULARITY <tG>
          TIMESLICE <ts_exp>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

be the considered ST4SQL query. It is equivalent to the following SQL query.

```
WITH granules AS (
     SELECT g.start, g.|END|
     FROM t-granularity AS tg, granule AS g
     WHERE g.|TGRANULARITY| = tg.id AND
            tg.name = <tG> AND
            g.|INDEX| BETWEEN <ts_exp>$start
                    AND <ts_exp>$|END| )
SELECT <sel_element_list>
FROM T1, T2, |\ldots|, Tn
WHERE <conditions> AND
      VALIDT(T1) && ANY granules
      AND |\ldots| AND
      VALIDT(Tn) && ANY granules
```

where `&&` is the operator representing the `intersect` function and `&& ANY` allows to check whether the first operand intersects any interval contained in the subquery provided as second operand.

To better explain the semantics let us consider an example. The following query retrieves the patients who in 2010 or 2011 had at least a contact lasted more than one hour. In this case, the valid time is represented by the date when contacts occurred.

```
SEMANTICS ATEMPORAL ON VALID TIME
  WITH TGRANULARITY Years TIMESLICE (2010,2011)
SELECT DISTINCT p.name, p.surname
FROM patient AS p JOIN contact AS c
     ON c.patient=p.id
WHERE c.duration > '01:00:00'
```

According to the semantics of the `ATEMPORAL` semantics (and the corresponding translation rules), the corresponding SQL query is the following.

```
WITH granules AS (
     SELECT g.start, g.|END|
     FROM t-granularity AS tg, granule AS g
     WHERE g.|TGRANULARITY| = tg.id AND
           tg.name = 'Years' AND
           g.|INDEX| BETWEEN 2010 AND 2011)
SELECT DISTINCT p.name, p.surname
FROM patient AS p JOIN contact AS c
     ON c.patient=p.id
WHERE c.duration > '01:00:00' AND
      VALIDT(c) && ANY granules
```

Note that the condition on `VALIDT(p)`, that should result applying the translation rules, has been omitted since the `patient` relation has no valid time.

Let us now consider a more complex semantics: the spatial `NEXT` semantics. Let

```
SEMANTICS NEXT ON <sdim> WITH SGRANULARITY <sG>
          ORDERED BY <sr> THROUGH <attr>
SELECT <attr_list>
FROM T1, ..., Tn
WHERE <jconds> AND <sconds>
```

be the considered ST4SQL query where `<attr_list>` is a list of names of attributes in the T_1, ..., T_n relations, while `<jconds>` and `<sconds>` are the sets of join and selection conditions. Note that, the distinction between join and selection conditions is not specified by the user, but can be easily computed by the system.

Note that when the `NEXT` semantics is specified, also the `ORDERED BY` and `THROUGH` options are mandatory.

As we explained in Section 7.4.2, the `NEXT` semantics allows one to evaluate a query on the pairs of tuples representing two consecutive states of an object. To do that, the first thing we have to do is to calculate the table containing the tuples representing the objects we are interested in. This table is the one resulting from the joining of all relations in the original `FROM` clause (i.e., `T1`, ..., `T2`) with respect to the join conditions `<jconds>`.

Thus, in the first place we calculate the `instance` relation as follow:

```
(SELECT
 FROM T1, ..., Tn
 WHERE <jconds>
) AS instance
```

where, as usual, when the user does not specify the join condition between all relations, the Cartesian product is applied.

We use this new relation in the SQL translation of the ST4SQL query provided by the user. In particular, in the translated `FROM` clause we have:

- to join two copies of the `instance` relation with respect to their values for the `<attr>` attribute, provided by the user as parameter of the `THROUGH` option;
- to add the relations containing the information about the `<sG>` spatial granularity.

Then, in the `WHERE` clause, we have to select only pairs whose two component tuples intersect two consecutive spatial granules in `<sG>` with respect to order defined by the `<sr>` spatial relation.

When the `NEXT` semantics is applied, the user can refer in the `SELECT` and `WHERE` clauses to the value of an attribute in the successor tuple by using the `NEXT(<attr>)` function. That is translated in the SQL query as `tnext.<attr>`, while all other attributes are referred as `t.<attr>`.

The `SELECT` clause of the resulting SQL query contains all the attributes named by the user in the original ST4SQL query, where, eventually, the `NEXT(<attr>)` function is translated as explained.

Concluding, the SQL query equivalent to the considered one is:

```
SELECT <attr_list>
FROM sgranularity AS sg
     INNER JOIN node AS n ON (n.granularity = sg.id)
     INNER JOIN node AS nnext ON (nnext.granularity = sg.id),
     instance AS t
     INNER JOIN instance AS tnext ON t.<attr> = tnext.<attr>
WHERE sg.name = <sG> AND n.geom.st_intersects(VALIDS(t))
      AND nnext.geom.st_intersects(VALIDS(tnext)) AND
      <sconds> AND NOT EXISTS (SELECT
            FROM node
            WHERE node.granularity = sg.id AND
                  evaluate(<sr>,n.geom,node.geom) AND
                  evaluate(<sr>,node.geom,nnext.geom) )
```

where the `evaluate(r,g1,g2)` function evaluates the spatial relationship `r` between geometries `g1` and `g2`.

The translation of the other new constructs and clauses is reported in Appendix C.

7.4 Querying PCR by using granularities

In this section we present the design of the Verona Psychiatric Case Register (PCR) database that we introduced in Section 3.2. In particular we show the conceptual schema of PCR and how it has been enriched with granularities for allowing spatio-temporal queries. After that, the designed database for PCR will be used in Section 7.4.2 for exemplifying several kinds of spatio-temporal queries

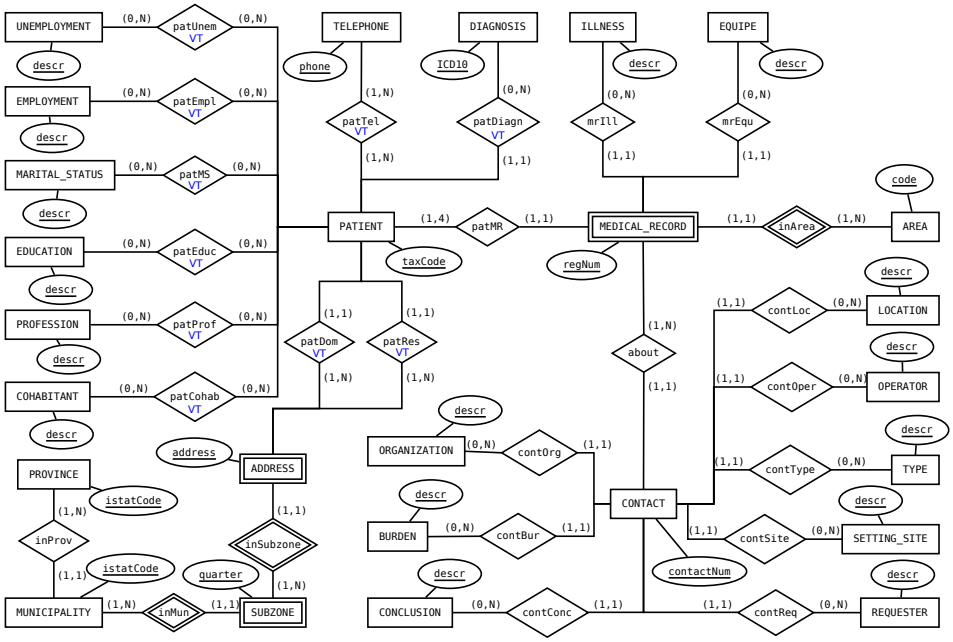


Fig. 7.7: TimeER schema of PCR database

based on granularities. These queries will be reported by using an extended version of T4SQL, a temporal query language proposed by Combi et al. [57] that we introduced in Section 7.1.

7.4.1 Integrating PCR and STGran

Fig. 7.7 depicts the conceptual schema of the database for the Verona PCR. The schema has been modeled by using the TimeER model [53]. In the schema only identification attributes are reported, while for a complete list of attributes we refer to the corresponding relational schema depicted in Fig. 7.8, that we will describe later in this section.

Personal information about patients is represented through in the `Patient` entity. As we have mentioned in the previous section, the Verona Health District is divided in four catchment areas. Health structures in each area may aid only patients living in that area (of course exceptions are made for emergency cases). In past years, when the information system was only paper-based, these area structures worked isolated from each other. Thus, each patient has a different medical record, with different registry numbers, in each area where he lived. For this reason, the relation between `Patient` and `Medical_Record` (that contains patients' medical records) is not a 1:1 relation. Moreover, medical records are identified by a registry number that is unique only inside a single catchment area (represented by the `Area` entity). Thus, `Medical_Record` is a weak entity identified by the `Area` entity.

Several historical information about a patient are stored in the database: for example, patients' employment, marital status, profession, education, cohabitant, and diagnosis. Each concept, to which corresponds an entity in the schema, has a valid time. We remember that in TimeER participation to a temporal relation may change over time and that the cardinality ratios associated to the relation apply at any time point. Thus, for example, at each time point a patient may have at most one associated diagnosis, while she may have more associated telephone numbers.

One domicile and one residence are associated to each patient. Both are represented as addresses located in a subzone in a municipality. Subzones represent quarters inside municipalities. Since in two different municipalities there could be two subzones with the same name, `Subzone` is a weak entity identified also by the `Municipality` entity. For the same reason also the `Address` entity is a weak entity identified by `Subzone`.

Each different patients' contact with CPS is represented through the `Contact` entity. Some auxiliary contact data are also stored, e.g., location, characteristic, type, and conclusion of the contact, operators that participated to the contact, and contact requester (including, potentially, the patient itself). The `Contact` entity is related to `Medical_Record` instead to `Patient` because contacts are entered by the CPS staff working in the area where patients live; thus, contacts contain also the area information (especially before the integration of the four catchment areas and their information systems). Note that `Contact` is a temporal entity since it has an associated valid lifespan `LS` [124]. This lifespan represents the time point when the contact occurred. Thus, in the relation schema for PCR it is treated as a valid time.

Fig. 7.8 depicts the relational schema of the database for the Verona PCR that will be used in Section 7.4.2 for exemplifying spatio-temporal queries on PCR. This relational schema can be obtained from the ER schema we presented before (see Fig. 7.7) through the well-known mapping algorithm [76] extended with mappings proposed by Gregersen et al. [101] for temporal objects introduced in TimeER.

Considering the standard non-temporal database objects, note that each table in the relational schema is identified by a new surrogate key, i.e., an integer database-generated identier without any inherent meaning. These key attributes have type `serial`, that, in PostgreSQL [168] (the DBMS we used for implementing our proposal), represents sequential automatically generated integer values. The only exceptions to this principle is the `Operator` table (identified by the operator's username) and tables representing municipalities, provinces, and regions (the Italian administrative subdivisions).

On the other hand, temporal entities and relationships we represented in the ER schema of PCRhave been translated by using the mapping proposed by Gregersen et al. [101]. In particular, temporal relations have been mapped to new tables in the relational schema in which, besides classical data and foreign keys, attributes for representing the annotated temporal dimension have been added. For example, the relation between `Patient` and `Profession` has been mapped to the `Patient_profession` table. This table contains

- two foreign keys `patient` and `profession` referring to `Patient` and `Profession`, respectively;

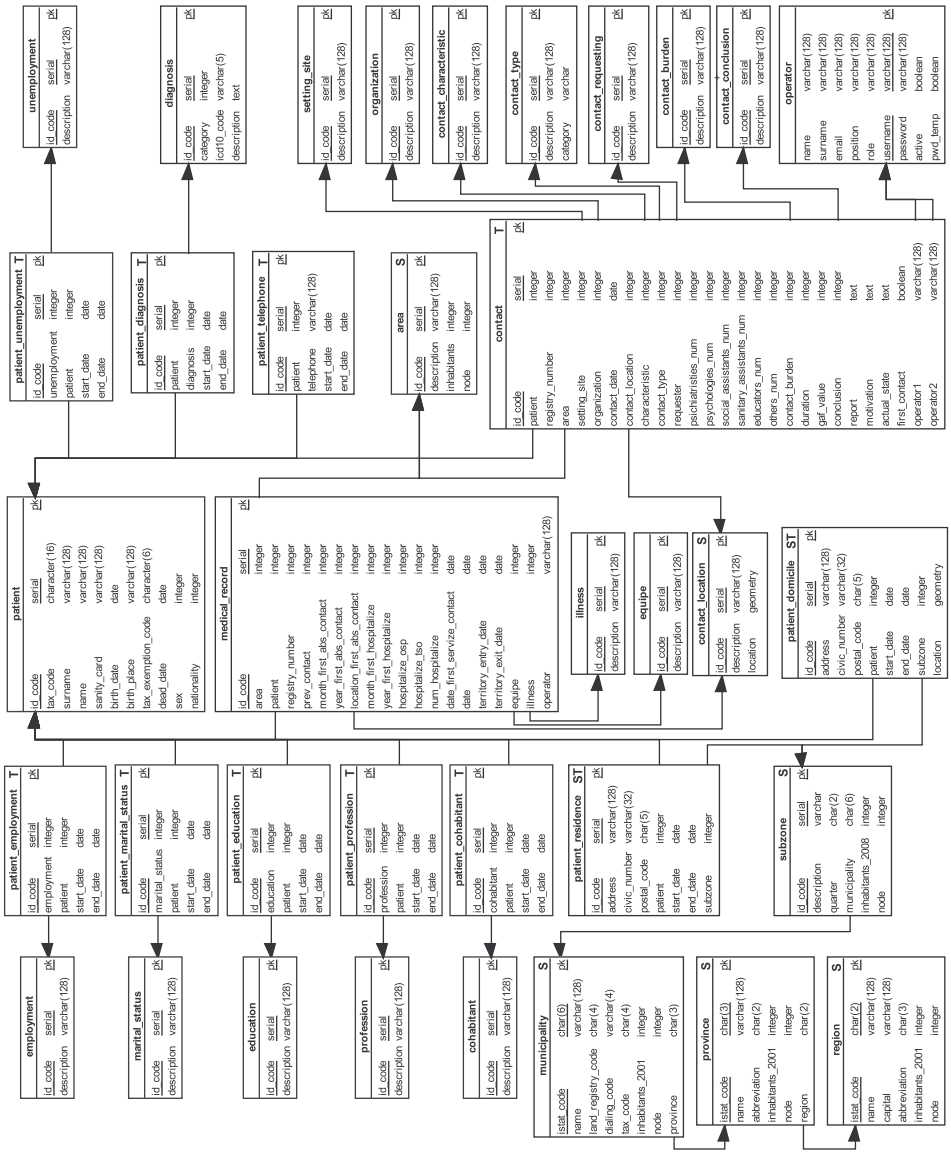


Fig. 7.8: Relational schema of PCR database

- two attributes `start_date` and `end_date`, with type `date`, for representing the valid time interval.

Note that, also relations with type 1:N, that usually are represented in the relational model by adding a foreign key into the participating table at the *N-side* of the relation, when are temporally extended are mapped to a new table, as usually happen for M:N relations. For example, the relation between `Patient` and `Diagnosis` are mapped to a new table reporting foreign keys referring to the two participating tables and two `date` attributes for representing start and end point of the valid time dimension.

In the ER schema in Fig. 7.7 we introduced only one temporal entity, i.e., the `CONTACT` entity and we annotated it with the lifespan temporal dimension. This lifespan is added in the corresponding `contact` relation in the relational schema as a `contact_date` attribute for representing the date when contacts occurred.

In order to perform spatial and spatio-temporal epidemiological analyses, besides temporal dimensions, the PCR database has been enriched also with geographic information. In Fig. 7.8 tables containing temporal, spatial, and spatio-temporal data have been annotated, respectively, with `T`, `S`, and `ST` beside their name. In particular, in the `patient_domicile` table the `location` attribute has been added. It is a point representing the position of the patient's domicile on the Earth surface. The same thing has been done in the `contact_location` table for describing the geographical position of hospitals, emergency rooms, and other CPS structures. Note that patients' domicile is a spatio-temporal information since it is also temporally qualified. Domiciles locations allow us to qualify and aggregate patients' domicile in both ways we introduced in Section 7.2.4. `Domicile` and `Residence` tables are also related to `Municipality` and thus to `Province` and `Region`, in turn. These three tables contain data related to the spatial granularities stored in the database for granularities we discussed in Section 7.2. Thus, domiciles and residences can be aggregated with respect to these granularities. To do that, these three tables have been enriched with the `node` foreign key referring to the `Node` table, i.e., each tuple in `Municipality`, `Province`, and `Region` has a reference to the spatial granule representing it. The same happen also to `Area` and `Subzone` tables. Thus, the database for granularities (described in Section 7.2) and PCR have been connected by the addition of these foreign keys. These references are depicted in Fig. 7.9. The ER schema depicted in the figure shows only entities of PCR and STGran that are in relation.

Note that spatial data added to PCR represent the spatial dimension of PCR entities. As it happens for the temporal data that represent valid time, spatial data represent valid space, i.e., the space location (point, line, or region) where an information is located in the space.

On the other hand, the domiciles locations allow us to aggregate, during the querying phase, patients' domiciles also with respect to other spatial and spatio-temporal granularities not stored in PCR. For example, in the next section we will show some queries on PCR for retrieving patients' data also with respect to pollution data.

In particular we gathered daily information about PM_{10}, ozone (O_3), nitrogen dioxide (NO_2), sulfur dioxide (SO_2), and carbon monoxide (CO) levels in Verona municipality from 2003 till now. These data have been surveyed by six stations

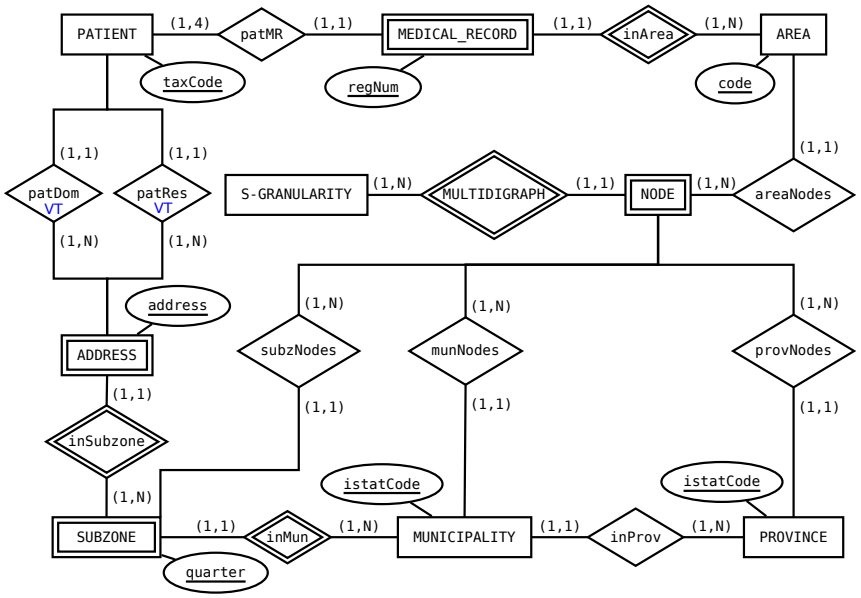


Fig. 7.9: TimeER schema showing connections between PCR and STGran

scattered in the Verona municipality and thus can be used to divide the municipality in six areas, for example by using the Thiessen polygons (also called Voronoi diagrams) [12]. Moreover we stored information about average day and night noise pollution, air and water pollution in municipalities in the Verona province in 2004. Despite these last data have been gathered in 2004, since they are average values, we can consider them to be valid also in other years and, thus, we use them for querying also data in other years.

Data about each different pollutant allowed us to define new spatial or spatio-temporal granularity by partitioning the Verona municipality and province with respect to the pollution level that has been surveyed. In these granularities, granules represent areas with the same pollution level. These new granularities have been used to analyze data about patients and patients' contacts.

7.4.2 Querying PCR using spatio-temporal granularities

In this section we show how it is possible to query a real spatio-temporal database (in our case the PCR database) by using our framework for granularities and what kind of new queries it allows us to specify. We show that through some examples. In these examples we query PCR in order also to find out data that can be used for detecting trends or patterns in patients' contacts with the CPS and correlations between patients, their contacts with the CPS, and pollution in the patients' domicile area. We implemented the databases described in previous sections by using PostgreSQL [168] and its spatial extension PostGIS [172]. Note that however PostgreSQL, as any other commercial DBMS, does not provide any support for the automatic management of temporal and spatial dimensions, except

temporal and spatial data types and related functions and operators that users can apply on them.

In order to provide a better and more complete explanation of possible queries based on granularities, we adopt the four semantics for temporal queries proposed in [57] (see Section 2.1.2). We use these semantics for classifying possible kinds of queries based on granularities. We informally extend these semantics to the spatial and spatio-temporal cases using granularities. We introduce semantics and this extended SQL language through examples, while its formal definition will be described in a proposal that is not published yet.

All the queries that will be presented in this non-formalized extension of T4SQL [57] can be translated in standard SQL [122] queries and evaluated in any DBMS supporting temporal and spatial datatypes. The translation, of course, produces a long and less readable query. For this reason here we present queries by using a T4SQL-based extended query language.

Extending the T4SQL syntax, semantics can be specified before a `SELECT` statement by following the BNF [115]:

```
SEMANTICS <sem> ON <td> [WITH TGRANULARITY <tG>]
          [TIMESLICE <ts_exp>] {, ...}
```

where the `TIMESLICE` token allows one to restrict the tuples to be considered to those whose value for the temporal dimension intersects the interval or date (when `WITH TGRANULARITY` is not used) or granule (when `WITH TGRANULARITY` is used) specified by the value of `<ts_exp>`. At most one semantics can be given to any temporal dimension. When no semantics is specified for a temporal dimension the system applies the `ATEMPORAL` semantics.

These new features may be applied to PCR. For example, the following query allows us to retrieve for each patient her contacts and the patient's diagnosis in the same month of the contacts. While the semantics originally proposed by Combi et al. allow one only to retrieve the diagnosis that is valid in the same time instant when the contact occurred, we can "relax" this query by using, for example, the `Months` granularity and retrieve diagnoses whose valid time intersects the same month in which contacts occurred.

```
SEMANTICS SEQUENCED ON VALID WITH TGRANULARITY Months
SELECT contact.patient, contact.report, diagnosis.icd10_code
FROM contact, patient_diagnosis, diagnosis
WHERE contact.patient = patient_diagnosis.patient AND
      patient_diagnosis.diagnosis = diagnosis.id_code
```

The `SEQUENCED` semantics provides that the considered granule is included in the output relation. Thus, the previous query results is a relation that, besides the patient's code, the contact report and the diagnosis ICD code, has also the month in which that data occurred. This query can be translated in the following standard SQL query:

```
SELECT g.index, c.patient, c.report, d.icd10_code
FROM contact AS c, patient_diagnosis AS pd, diagnosis AS d,
     t-granularity AS tg, granule AS g
WHERE c.patient = pd.patient AND pd.diagnosis = d.id_code AND
      g.tgran = tg.id AND tg.name = 'Months' AND
      c.contact_date BETWEEN g.start AND g.end AND
      INTERSECT(pd.start_date, pd.end_date, g.start, g.end)
```

Where `intersect` is a procedure for checking whether the interval defined by the first two parameters intersects the interval defined by the last two parameters.

By using the `CURRENT` semantics we can retrieve the patients' professions in the current month:

```
SEMANTICS CURRENT ON VALID TIME WITH TGRANULARITY Months
SELECT name, surname, profession.description
FROM patient, patient_profession, profession
WHERE patient_profession.patient = patient.id_code AND
      patient_profession.profession = profession.id_code
```

Another example of query based on temporal granularities is that for the calculation of the average number of contacts per patient with respect to the mental disease diagnosed in the same month when contacts occurred:

```
SEMANTICS SEQUENCED ON VALID TIME WITH TGRANULARITY Months
SELECT d.category, COUNT(c.id_code)/COUNT(DISTINCT c.patient)
FROM contact AS c, patient_diagnosis AS pd, diagnosis AS d
WHERE c.patient = pd.patient AND pd.diagnosis = d.id_code
GROUP BY d.category
```

This query can be modified for obtaining in each month the average number of contacts per patient with respect to diagnosed mental disease. To do that, we extend the temporal `GROUP BY` clause introduced in [57] for adding the possibility to group together tuples according to the temporal granule intersecting the value of the tuples over the specified temporal dimension.

```
SELECT TGROUP(VALIDT(c)), d.category,
       COUNT(c.id_code)/COUNT(DISTINCT c.patient)
FROM contact AS c, patient_diagnosis AS pd, diagnosis AS d
WHERE c.patient = pd.patient AND pd.diagnosis = d.id_code
GROUP BY VALIDT(c) ON Months, d.category
```

Where we remember that `VALIDT(R)` returns the valid time of tuples in the `R` relation, and `TGROUP` is used for including in the `SELECT` clause the index of the time granule to which the other resulting data are associated (i.e., in this case it includes in the result relation the index of the month).

The `SEQUENCED` spatial semantics allows us to retrieve, for example, the contacts of patients living (at the time instant when contacts occurred) in the same subzone (i.e., quarter) where contacts took place.

```
SEMANTICS SEQUENCED ON VALID TIME,
          SEQUENCED ON VALID SPACE
          WITH SGRANULARITY Subzones
SELECT c.patient, c.report
FROM patient_domicile AS pdom, contact AS c,
     contact_location AS cl
WHERE c.patient = pdom.patient AND
      c.contact_location = cl.id_code
```

Similarly to what happens in the temporal case, the `SEQUENCED` semantics requires that the spatial granule intersecting the specified spatial dimensions is included in the output relation. Thus, in the previous query both the time instant and the spatial granule are returned besides other required data.

We said in Section 7.2.4 that classical data can be spatially qualified both adding a point location to the data and adding a direct reference to the spatial granule where data are located. The latter is the approach used for spatially qualifying patients' residences. Thus, patients' residence can be used, similarly to what done with domiciles, for retrieving data. For example, the following query returns the patients whose residence is in a municipality that intersects an area with a high level of acoustic noise.

```
SEMANTICS ATEMPORAL ON VALID TIME, SEQUENCED ON VALID SPACE
          WITH SGRANULARITY DayAcousticNoise
          SPACESLICE 'High'
SELECT p.tax_code, p.name, p.surname
FROM patient AS p, patient_residence AS pr, subzone AS s,
     municipality AS m
WHERE pr.patient = p.id_code AND pr.subzone = s.id_code AND
      s.municipality = m.istat_code
```

Where the `SPACESLICE` token allows one to restrict the query evaluation only to the specified spatial granule.

As we explained before, the `CURRENT` semantics can be used to restrict the query evaluation only on those tuples whose value over the specified spatial dimension intersects the spatial granule containing the current geographical position of the query submitter. This information can be locally saved, for example, in a system configuration file, or can be inferred (eventually from the DBMS server machine) at evaluation time from the IP address that has submitted the query. In the PCR case, the `CURRENT` semantics can be used, for example, by a physician for retrieving diagnoses only of patients currently living in the catchment area where she works.

```
SEMANTICS CURRENT ON VALID TIME, CURRENT ON VALID SPACE
          WITH SGRANULARITY CatchmentAreas
SELECT DISTINCT p.tax_code, p.name, p.surname, d.icd10_code
FROM patient_domicile AS pdom, patient AS p,
     patient_diagnosis AS pdia, diagnosis AS d
WHERE pdom.patient = p.id_code AND p.id_code = pdia.patient
      AND pdia.diagnosis = d.id_code
```

The NEXT semantics allows one to relate tuples whose spatial dimension intersects a granule A with tuples related to the same objects and whose value for the spatial dimension intersects granules next to A with respect to the order defined by a given spatial relationship. For example, the following query relates the number of contacts for each patient in a quarter and in the quarters that are South of it (this may be useful, for example, for studying the correlation between the wind direction and the disease rate).

```
SEMANTICS NEXT ON VALID SPACE WITH SGRANULARITY Quarters
          ORDERED BY South THROUGH c.patient
SELECT c.patient, COUNT(c.id_code), COUNT(NEXT(c.id_code))
FROM contact AS c JOIN contact_location AS cl
         ON c.contact_location = cl.id_code
GROUP BY c.patient
```

The first thing done in order to evaluate a query with the NEXT semantics is the join between two instances of the relation in the FROM clause or (when the FROM clause contains more relations) of the relation resulting from the evaluation of the join conditions (given by the JOIN operator or in the WHERE clause) on the relations in the FROM clause. These two instances are then joined on the basis of the attributes specified after the THROUGH token. Among the obtained pairs of tuples the system selects only those whose values over the given spatial dimension intersect two consecutive granules with respect to the order defined by the relationship specified after the ORDERED BY token. The function NEXT used in the SELECT clause returns the value on the successor tuple of the attribute given as parameter.

By using the SPACELESS semantics the user disables any support from the system and she has, eventually, to manage explicitly the spatial data and granularities. In this case no spatial granularity has to be specified. For example, the following query retrieves the patients living in a highly polluted area and contacting CPS structures located in little polluted areas.

```
SEMANTICS SEQUENCED ON VALID TIME WITH TGRANULARITY Weeks,
          SPACELESS ON VALID SPACE
SELECT DISTINCT p.tax_code, p.name, p.surname
FROM patient AS p JOIN contact AS c ON p.id_code=c.patient
     JOIN patient_domicile AS pd ON pd.patient = p.id_code
     JOIN contact_location AS cl
          ON c.contact_location = cl.id_code,
     s-granularity AS sg JOIN node AS n1 ON n1.sgran = sg.id
     JOIN nodelabel AS nl1 ON n1.label = nl1.id
     JOIN node AS n2 ON n2.sgran = sg.id
     JOIN nodelabel AS nl2 ON n2.label = nl2.id
WHERE sg.name = 'PM10' AND nl1.label = 'High' AND
      nl2.label = 'Low' AND n1.geom.st_contains(pd.location) AND
      n2.geom.st_contains(cl.location)
```

Where we remember that st_contains is a procedure that checks whether the spatial object on which it is called contains the spatial object provided as parameter.

Spatial granularities can be used also for grouping tuples. The following query exemplify this usage. It retrieves the average number of contacts per patient grouped with respect to the diagnosis category and the day acoustic noise level.

```
SELECT SGROUP(VALIDS(cl)), d.category,
       COUNT(c.id_code)/COUNT(DISTINCT c.patient)
FROM contact AS c, patient_diagnosis AS pd, diagnosis AS d,
     contact_location AS cl
WHERE c.patient = pd.patient AND pd.diagnosis = d.id_code AND
      c.contact_location = cl.id_code
GROUP BY VALIDS(cl) ON DayAcousticNoise, d.category
```

Where we remember that `VALIDS(R)` returns the valid space of tuples in the `R` relation, while `SGROUP` is used for including in the `SELECT` clause the index of the space granules used in the `GROUP BY` clause. Note that the geometry of granules in the result can be obtained subsequently by joining the `Node` table through the granule index.

In previous examples about the usage of spatial granularities, we always aggregated patients' domiciles using a unique `Municipality` granularity, thus supposing that this granularity was always valid in time, i.e., it never changes over time. However, patients' domiciles are spatio-temporal data, thus they may be aggregated by using the spatial granularity that actually was valid at the time when domiciles were valid. Spatio-temporal granularities can be used in queries for this purpose. As a matter of fact, spatio-temporal granularities temporally qualify spatial granularities. Thus, they allows one to spatially aggregate data with respect to a spatial granularity whose definition changes over time. The value of tuples over a given temporal dimension is used to select the spatial granularity valid at the same time of considered tuples. Thus, spatio-temporal semantics require that both a temporal dimension and a spatial dimension are specified, this pair is a spatio-temporal dimension.

A spatio-temporal granularity groups spatial granularities with respect to granules of a temporal granularity. Each tuple is spatially qualified by using spatial granularities valid during the time granules intersecting the value of the tuple over the given temporal dimension. Since this value may be an interval, and that (based on the definition of spatio-temporal granularity) during this interval the spatial granularity may evolve, several spatial granularities may be used to qualify each tuple. For this reason the user may specify an aggregate function to apply to spatial granularities in order to obtain just one spatial granularity among all granularities valid during the interval. Aggregate functions include, for example, `first` (that returns the spatial granularity valid at the first instant in the interval), `last` (that returns the spatial granularity valid at the last instant in the interval), and operations defined in Chapter 4 for calculating union, intersection, and difference of spatial granularities. If the aggregate function is not provided by the user, the system uses all spatial granularities valid in the interval, one by one, for querying the considered tuple, and all these combinations are used for obtaining the final resulting relation.

Since, spatio-temporal granularities enclose in one structure both a temporal and a spatial granularity, they can be used for temporal and spatial querying at

the same time. For this purpose the user specifies a semantics on both the temporal and the spatial dimension making up the given spatio-temporal dimension. These temporal and spatial semantics are used as we have described before in this section, but in this case the temporal dimension is used also for selecting the spatial granularity valid at the same time of considered tuples.

We note that specifying the `SEQUENCED` semantics on the spatial dimension and the `CURRENT` semantics (or a timesliced `SEQUENCED` semantics) on the temporal one, a user can query data with respect to a spatially qualified timeslice. On the other hand, specifying the `SEQUENCED` semantics on the temporal dimension and the `CURRENT` semantics (or a spacesliced `SEQUENCED` semantics) on the spatial one, a user can query data with respect to the evolution over time of spatial granules. Of course, the `ATEMPORAL` and `SPACELESS` semantics cannot be used in spatio-temporal semantics.

In summary, a spatio-temporal semantics may be specified by using the following BNF syntax:

```
SEMANTICS <sem> ON <td> [TIMESLICE <ts_exp>] AND
          <sem> ON <sd> [SPACESLICE <ss_exp>]
          WITH STGRANULARITY <stG>
          [SIMPLIFY BY <s_aggr_funct>]
```

Where `td` and `sd` are a temporal and a spatial dimension, respectively, and `s_aggr_funct` is a spatial aggregate function.

For example, the following query retrieves information about patients' domicile valid in 2010 and located, during this interval, in an area with high or medium PM_{10} pollution level.

```
SEMANTICS SEQUENCED ON VALID TIME
          TIMESLICE period '[2010-01-01,2010-12-31]' AND
          SEQUENCED ON VALID SPACE SPACESLICE 'High','Medium'
          WITH STGRANULARITY PM10_Weeks
SELECT DISTINCT p.tax_code, p.name, p.surname
FROM patient_domicile AS pd, patient AS p
WHERE pd.patient = p.id_code
```

Note that `PM10_Weeks` is a spatio-temporal granularity representing daily PM_{10} surveys associated to the `Weeks` temporal granularity. Thus, the query is evaluated on those tuples whose value over the valid space intersects a PM_{10} granule valid in a week intersecting also the tuples value over the valid time. Moreover, as happen for temporal and spatial semantics, the resulting relation includes the temporal and spatial granules qualifying tuples.

8

Conclusions

This thesis focused on the notions of spatial and spatio-temporal granularities and their usage for the management of spatio-temporal data in a clinical psychiatric database. Conversely to temporal granularities about which a solid, well-known, and accepted literature exists since the end of the nineties, researchers do not agree on a common definition and formalization of spatial and spatio-temporal granularities. Moreover, these terms have been used with different meanings. Some proposals about spatial and spatio-temporal granularities consider them as synonyms of "multi-resolution". On the other hand, main proposals about temporal granularities and some proposals about spatial and spatio-temporal granularities are based on the idea that granularities can be considered as meta-data through which spatio-temporal data can be managed and queried. This means that granularities can be used for qualifying and enriching other spatio-temporal data in order to exploit their spatio-temporal information. However, at the best of our knowledge, all proposals about spatial and spatio-temporal granularities following this same approach do not define a complete framework for spatial and spatio-temporal granularities. In particular, proposals in literature do not define a set of relationships and operations over spatial and spatio-temporal granularities. This leads to the impossibility to use them in real databases for managing spatio-temporal data. Indeed, operations allow one to create new granularities from already defined ones, while relationships allow one to know how data associated with different granularities are related and can be compared.

Starting from these considerations, in this dissertation we defined two complete theoretical frameworks for dealing with spatial and spatio-temporal granularities, with the idea that they represent meta-data useful for managing other spatio-temporal data. Both frameworks have been defined for both the vector model and the raster model for the representation of spatial data.

We also presented an inference system for relationships between granularities. Relationships allow one to know how granularities and their granules are related. This allows one also to know how data qualified with those granularities are related and, thus, how they can be compared also when they are associated with different granularities. On the other hand, the evaluation of relationships between granularities can be computational heavy. For this reason, we studied an inference system for avoiding this evaluation. Indeed, in many cases relationships can

be inferred from other relationships that we already know to hold. The proposed inference system comprises a set of rules that, when some relationships hold, infer the other relationships that definitely hold (i.e., relationships that hold in any possible model satisfying the relationships in the premises). Studied rules include rules for the concatenation of relationships and for the inverse of relationships.

After this first foundational and theoretical part of the dissertation, we applied our notions to the management of spatio-temporal data in a clinical psychiatric database. For this reason, we introduced a motivating scenario showing off the importance and the possible usages of spatio-temporal granularities in clinical epidemiological studies. The scenario includes a spatio-temporal clinical database for the management of psychiatric patients, their personal and clinical information, and their contacts with the psychiatric services in the province of Verona, in Italy. Despite we applied our approach to spatio-temporal granularities to a clinical database, we note that notions we proposed in this dissertation can be applied to spatio-temporal databases in any application or research field.

In order to allow one to use granularities for the management of spatio-temporal data stored in relational DBMSs, we designed a database for storing temporal, spatial, and spatio-temporal granularities. The database stores information about granularities (e.g., the name and the shape of their granules), the operations used for creating them, and relationships between them. This database can be integrated with existing spatio-temporal databases in order to qualify data in the original database with granularities. We showed how this can be done through some examples on the introduced motivating scenario. Thus, we proposed a query language to retrieve data from a spatio-temporal database enriched with granularities. In particular, the proposed query language allows one to specify for each temporal, spatial, and spatio-temporal dimension a semantics defining how the system has to manage the given dimension during the query evaluation. The four proposed semantics are based on granularities and allow one, together with the other proposed constructs, to retrieve qualified data with respect to spatio-temporal constraints (spatio-temporal selection) and to join data with respect to the granules they belong to (spatio-temporal join). Moreover, the proposed query language allows one to group data with respect to the granules they belong to in order to apply on them aggregate functions. This permits one to aggregate data in granularities and to obtain data representing properties valid in the whole granules. All proposed queries based on granularities have been exemplified on the motivating scenario.

We applied the proposed spatio-temporal query language to the clinical database we designed for managing data collected by the Verona Community-based Psychiatric Service. Spatio-temporal data about psychiatric patients and their contacts with psychiatric services have been retrieved by using spatio-temporal queries based on granularities. The proposed approach and query language allowed to define in an easier way otherwise complex queries. Moreover, the proposed approach can be used by epidemiologists for better exploiting spatio-temporal data in the clinical psychiatric database.

Concluding, all notions, concepts, and approaches proposed in this thesis allow one to use granularities for enriching spatio-temporal databases and for retrieving data with respect to spatio-temporal constraints. For this purpose, we extended

a query language in order to allow the two basic features of any query language, selection and join, to be based on granularities.

For future work we plan to extend the designed database for granularities in order to store also granularities based on raster maps. Moreover, we consider to implement the inference system (along the procedures for evaluating relationships), extending our proposal to consider a database system able to manage granularities with inference capabilities. Finally, we are interested in to introduce granularities and related ideas in data warehouses analyses and report design, allowing spatio-temporal data analyses based on granularities to be performed also by less technically knowledgeable personnel.

References

1. J. J. Abellan, S. Richardson, and N. Best. Use of space-time models to investigate the stability of patterns of disease. *Environmental Health Perspectives*, 116(8):1111–1119, August 2008.
2. T. Abraham and J. F. Roddick. Survey of spatio-temporal databases. *GeoInformatica*, 3(1):61–99, 1999.
3. R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. S. P. Chen, editors, *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press.
4. W. Ahrens and I. Pigeot, editors. *Handbook of epidemiology*. Springer, 1994.
5. A. Ait-Braham, B. Theodoulidis, and G. Karvelis. Conceptual modelling and manipulation of temporal databases. In P. Loucopoulos, editor, *Entity-Relationship Approach - ER'94, Business Modelling and Re-Engineering, 13th International Conference on the Entity-Relationship Approach, Manchester, U.K., December 13-16, 1994, Proceedings*, volume 881 of *Lecture Notes in Computer Science*, pages 296–313. Springer, 1994.
6. J. F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983.
7. F. Amaddeo, J. Beecham, P. Bonizzato, A. Fenyo, M. Knapp, and M. Tansella. The use of a case register to evaluate the costs of psychiatric care. *Acta Psychiatrica Scandinavica*, 95(3):189–198, 1997.
8. F. Amaddeo, G. Bisoffi, R. Micciolo, M. Piccinelli, and M. Tansella. Frequency of contact with community-based psychiatric services and the lunar cycle: a 10-year case-register study. *Social Psychiatry and Psychiatric Epidemiology*, 32:323–326, 1997. 10.1007/BF00805436.
9. F. Amaddeo and M. Tansella. Information systems for mental health. *Epidemiologia e psichiatria sociale*, 18:1–4, 2009.
10. J. M. Angulo and M. D. Ruiz-Medina. Spatio-temporal modeling of environmental and health processes. *Stochastic Environmental Research and Risk Assessment*, 22(1):1–2, Mar. 2008.
11. L. Anselin, I. Syabri, and Y. Kho. GeoDa: an introduction to spatial data analysis. *Geographical Analysis*, 38:5–22, 2006.
12. F. Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, 1991.
13. P. Aylin, R. Maheswaran, J. Wakefield, S. Cockings, L. Jarup, R. Arnold, G. Wheeler, and P. Elliott. A national facility for small area disease mapping and rapid initial assessment of apparent disease clusters around a point source: the

uk small area health statistics unit. *Journal of Public Health Medicine*, 21:289–298, 1999.
14. C. W. Bachman. The programmer as navigator. *Commun. ACM*, 16:653–658, November 1973.
15. S. Badaloni and M. Giacomin. A fuzzy extension of Allen's interval algebra. In *AI*IA 99:Advances in Artificial Intelligence,*, volume 1792 of *LNCS*, pages 155–165. Springer, 1999.
16. J. Barwise. *Language, Proof and Logic*. Center for the Study of Language and Inf, City, 2002.
17. L. Beale, J. J. Abellan, S. Hodgson, and L. Jarup. Methodologic issues and approaches to spatial epidemiology. *Environmental Health Perspectives*, 116(8):1105–1110, August 2008.
18. A. Belussi, C. Combi, and G. Pozzani. Towards a formal framework for spatio-temporal granularities. In *Proceedings of the 15th International Symposium on Temporal Representation and Reasoning, TIME 2008*, pages 49–53, Montréal, Canada, June 2008. IEEE Computer Society.
19. A. Belussi, C. Combi, and G. Pozzani. Formal and conceptual modeling of spatio-temporal granularities. In B. C. Desai, D. Saccà, and S. Greco, editors, *Proceedings of the International Database Engineering and Applications Symposium, IDEAS 2009*, pages 275–283, Cetraro, Calabria, Italy, Sept. 2009. ACM.
20. A. Belussi, C. Combi, G. Pozzani, and F. Amaddeo. Dealing with multigranular spatio-temporal databases to manage psychiatric epidemiology data. *Journal of Biomedical Informatics*, 2011. Submitted to.
21. B. Bennett, D. R. Magee, A. G. Cohn, and D. C. Hogg. Enhanced tracking and recognition of moving objects by reasoning about spatio-temporal continuity. *Image and Vision Computing*, 26(1):67–81, Jan. 2008.
22. M. Berndtsson and J. Mellin. Active database, active database (management) system. In L. Liu and M. T. Özsu, editors, *Encyclopedia of Database Systems*, pages 27–28. Springer US, 2009.
23. C. Bettini, C. E. Dyreson, W. S. Evans, R. T. Snodgrass, and X. S. Wang. A glossary of time granularity concepts. *LNCS*, 1399:406–413, 1998.
24. C. Bettini, S. G. Jajodia, and S. X. Wang. *Time Granularities in Databases, Data Mining and Temporal Reasoning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000.
25. C. Bettini, X. Sean, and W. S. Jajodia. A general framework and reasoning models for time granularity. In *Proceedings of the Third International Workshop on Temporal Representation and Reasoning (TIME-96*, pages 104–111. IEEE Computer Society Press, 1996.
26. C. Bettini, X. S. Wang, and S. Jajodia. Testing complex temporal relationships involving multiple granularities and its application to data mining. In *15th ACM SIGACT-SIGMOD-SIGART Symposium on the Principles of Database Systems*, pages 68–78, Montreal, Canada, 1996. ACM Press.
27. C. Bettini, X. S. Wang, and S. Jajodia. A general framework for time granularity and its application to temporal reasoning. *Annals of Mathematics and Artificial Intelligence*, 22(1-2):29–58, 1998.
28. R. Billen and E. Clementini. A model for ternary projective relations between regions. In E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, and E. Ferrari, editors, *Advances in Database Technology - EDBT 2004, 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004, Proceedings*, volume 2992 of *Lecture Notes in Computer Science*, pages 310–328. Springer, 2004.

29. T. Bittner. Granularity in reference to spatio-temporal location and relations. In S. M. Haller and G. Simmons, editors, *FLAIRS Conference*, pages 466–470. AAAI Press, 2002.
30. T. Bittner. Reasoning about qualitative spatio-temporal relations at multiple levels of granularity. In F. van Harmelen, editor, *Proceedings of the 15th European Conference on Artificial Intelligence – ECAI 2002*, Lyon, France, July 2002. IOS Press.
31. M. H. Böhlen, C. S. Jensen, and B. Skjellaug. Spatio-temporal database support for legacy applications. In *SAC*, pages 226–234, 1998.
32. M. H. Böhlen, C. S. Jensen, and R. T. Snodgrass. Temporal statement modifiers. *ACM Trans. Database Syst.*, 25(4):407–456, Dec. 2000.
33. M. Breunig, C. Türker, M. Böhlen, S. Dieker, R. Güting, C. Jensen, L. Relly, P. Rigaux, H.-J. Schek, and M. Scholl. Chapter 7: Architectures and implementations of spatio-temporal database management systems. In *Spatio-Temporal Databases*, volume 2520 of *Lecture Notes in Computer Science*, pages 263–318. Springer Berlin / Heidelberg, 2003.
34. E. Camossi, M. Bertolotto, and E. Bertino. A multigranular object-oriented framework supporting spatio-temporal granularity conversions. *Int. J. Geogr. Inf. Sci.*, 20(5):511–534, 2006.
35. E. Camossi, M. Bertolotto, and E. Bertino. Querying multigranular spatio-temporal objects. In S. S. Bhowmick, J. Küng, and R. Wagner, editors, *Proceedings of DEXA 2008*, volume 5181 of *Lecture Notes in Computer Science*, pages 390–403. Springer, 2008.
36. E. Camossi, M. Bertolotto, E. Bertino, and G. Guerrini. Issues on modeling spatial granularities. In *COSIT 03 Workshop: Fundamental Issues in Spatial and Geographic Ontology*, Ittingen, Switzerland, September 2003.
37. E. Camossi, M. Bertolotto, E. Bertino, and G. Guerrini. A multigranular spatiotemporal data model. In E. Hoel and P. Rigaux, editors, *GIS-03*, pages 94–101. ACM Press, Nov. 7–8 2003.
38. R. G. G. Cattell, D. K. B. M. Berler, J. Eastman, D. Jordan, C. Russell, O. Schadow, T. Stanienda, and F. Velez, editors. *The Object Data Standard: ODMG 3.0*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000.
39. Centers for Disease Control and Prevention. Environmental Public Health Tracking. URL: http://www.cdc.gov/nceh/tracking/.
40. Centers for Disease Control and Prevention. Guidelines for investigating clusters of health events. In *MMWR Recommendations and Reports*, volume 39 of *Morbidity and Mortality Weekly Report (MMWR)*, pages 1–16. July 1990.
41. Centers for Disease Control and Prevention. Avian influenza. http://www.cdc.gov/flu/avian/.
42. Centers for Disease Control and Prevention. HIV/AIDS. http://www.cdc.gov/hiv/.
43. Centers for Disease Control and Prevention. Severe Acute Respiratory Syndrome (SARS). http://www.cdc.gov/ncidod/sars/.
44. Centers for Disease Control and Prevention. Passive smoking: Beliefs, attitudes, and exposures, 1986. http://www.cdc.gov/mmwr/preview/mmwrhtml/00000014.htm.
45. S. Chakravarthy and S.-K. Kim. Resolution of time concepts in temporal databases. *Inf. Sci.*, 80(1-2):91–125, 1994.
46. C. X. Chen. Spatio-temporal query languages. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 1125–1128. Springer, 2008.
47. C. X. Chen and C. Zaniolo. SQLST: A spatio-temporal data model and query language. In *ER*, pages 96–111, 2000.
48. E. Clementini, P. D. Felice, and D. Hernández. Qualitative representation of positional information. *Artif. Intell.*, 95(2):317–356, 1997.

49. E. Clementini, P. D. Felice, and P. van Oosterom. A small set of formal topological relationships suitable for end-user interaction. In D. J. Abel and B. C. Ooi, editors, *Advances in Spatial Databases, Third International Symposium, SSD'93, Singapore, June 23-25, 1993, Proceedings*, volume 692 of *Lecture Notes in Computer Science*, pages 277–295. Springer, 1993.
50. E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.
51. A. G. Cohn, B. Bennett, J. Gooday, and N. M. Gotts. Qualitative spatial representation and reasoning with the region connection calculus. *GeoInformatica*, 1(3):275–316, 1997.
52. C. Combi, G. Cucchi, and F. Pinciroli. Applying object-oriented technologies in modeling and querying temporally oriented clinical databases dealing with temporal granularity and indeterminacy. *Information Technology in Biomedicine, IEEE Transactions on*, 1(2):100–127, 2002.
53. C. Combi, S. Degani, and C. S. Jensen. Capturing temporal constraints in temporal er models. In Q. Li, S. Spaccapietra, E. S. K. Yu, and A. Olivé, editors, *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings*, volume 5231 of *Lecture Notes in Computer Science*, pages 397–411. Springer, 2008.
54. C. Combi, M. Franceschet, and A. Peron. Representing and reasoning about temporal granularities. *J. Log. Comput*, 14(1):51–77, 2004.
55. C. Combi, M. Gozzi, J. M. Juárez, B. Oliboni, and G. Pozzi. Conceptual modeling of temporal clinical workflows. In *14th International Symposium on Temporal Representation and Reasoning (TIME 2007), 28-30 June 2007, Alicante, Spain*, pages 70–81. IEEE Computer Society, 2007.
56. C. Combi and A. Montanari. Data models with multiple temporal dimensions: Completing the picture. In K. R. Dittrich, A. Geppert, and M. C. Norrie, editors, *Advanced Information Systems Engineering, 13th International Conference, CAiSE 2001, Interlaken, Switzerland, June 4-8, 2001, Proceedings*, volume 2068 of *Lecture Notes in Computer Science*, pages 187–202. Springer, 2001.
57. C. Combi, A. Montanari, and G. Pozzi. The T4SQL temporal query language. In M. J. Silva, A. H. F. Laender, R. A. Baeza-Yates, D. L. McGuinness, B. Olstad, Ø. H. Olsen, and A. O. Falcão, editors, *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal, November 6-10, 2007*, pages 193–202. ACM.
58. C. Combi, F. Pinciroli, M. Cavallaro, and G. Cucchi. Querying temporal clinical databases with different time granularities: the GCH-OSQL language. In *Proceedings of the Annual Symposium on Computer Application in Medical Care*, page 326. American Medical Informatics Association, 1995.
59. C. Combi, F. Pinciroli, G. Musazzi, and C. Ponti. Managing and displaying different time granularities of clinical information. In *Proceedings of the Annual Symposium on Computer Application in Medical Care*, page 954. American Medical Informatics Association, 1994.
60. C. Combi, F. Pinciroli, and G. Pozzi. Managing different time granularities of clinical information by an interval-based temporal data model. *Methods of information in medicine*, 34(5):458, 1995.
61. C. Combi, F. Pinciroli, and G. Pozzi. Managing time granularity of narrative clinical information: The temporal data model time-nesis. In *International Symposium on Temporal Representation and Reasoning (TIME)*, 1996.
62. C. Combi and G. Pozzani. An inference system for relationships between spatio-temporal granularities. Technical Report 80/2010, Department of Computer Science, University of Verona, Italy, Sept. 2010. http://www.di.univr.it/report.

63. C. Combi and G. Pozzani. An inference system for relationships between spatio-temporal granularities. In *12th International Symposium on Spatial and Temporal Databases, SSTD 2011*, Minneapolis, MN, USA, 2011. Submitted to.
64. C. Combi and R. Rossato. Representing trends and trend dependencies with multiple granularities. In *TIME '06: Proceedings of the Thirteenth International Symposium on Temporal Representation and Reasoning*, pages 3–10, Washington, DC, USA, 2006. IEEE Computer Society.
65. U. Dal Lago and A. Montanari. Calendars, time granularities, and automata. In C. S. Jensen, M. Schneider, B. Seeger, and V. J. Tsotras, editors, *Advances in Spatial and Temporal Databases, 7th International Symposium, SSTD 2001, Redondo Beach, CA, USA, July 12-15, 2001, Proceedings*, volume 2121 of *Lecture Notes in Computer Science*, pages 279–298. Springer, 2001.
66. U. Dal Lago, A. Montanari, and G. Puppis. Compact and tractable automaton-based representations of time granularities. *Theor. Comput. Sci*, 373(1-2):115–141, 2007.
67. M. DeMers. *GIS modeling in raster*. Wiley, 2002.
68. L. Deng, Z. Liang, and Y. Zhang. A fuzzy temporal model and query language for fter databases. In J.-S. Pan, A. Abraham, and C.-C. Chang, editors, *Eighth International Conference on Intelligent Systems Design and Applications, ISDA 2008, 26-28 November 2008, Kaohsiung, Taiwan, 3 Volumes*, pages 77–82. IEEE Computer Society, 2008.
69. M. Duckham, K. Mason, J. Stell, and M. Worboys. A formal approach to imperfection in geographic information. *Computers, Environment and Urban Systems*, 2001:89–103, July 03 2001.
70. F. Ederer, M. H. Myers, and N. Mantel. A statistical problem in space and time: do leukemia cases come in clusters? *Biometrics*, 20(3):626–638, September 1964.
71. M. Egenhofer and J. Herring. Categorizing binary topological relationships between regions, lines, and points in geographic databases. Technical report, University of Maine, Orono, Maine, Dept. of Surveying Engineering, 1992.
72. M. J. Egenhofer. A formal definition of binary topological relationships. In W. Litwin and H.-J. Schek, editors, *Foundations of Data Organization and Algorithms, 3rd International Conference, FODO 1989, Paris, France, June 21-23, 1989, Proceedings*, volume 367 of *Lecture Notes in Computer Science*, pages 457–472. Springer, 1989.
73. M. J. Egenhofer. Reasoning about binary topological relations. In O. Günther and H.-J. Schek, editors, *Advances in Spatial Databases, Second International Symposium, SSD'91, Zürich, Switzerland, August 28-30, 1991, Proceedings*, volume 525 of *Lecture Notes in Computer Science*, pages 143–160. Springer, 1991.
74. M. J. Egenhofer. Spatial sql: A query and presentation language. *IEEE Trans. Knowl. Data Eng.*, 6(1):86–95, 1994.
75. P. Elliott and D. Wartenberg. Spatial epidemiology: Current approaches and future challenges. *Environmental Health Perspectives*, 112(9):998–1006, June 2004.
76. R. Elmasri and S. Navathe. *Fundamentals of Database Systems*. Addison-Wesley Publishing Company, USA, 6th edition, 2010.
77. M. Erwig, R. H. Güting, M. Schneider, and M. Vazirgiannis. Spatio-temporal data types: An approach to modeling and querying moving objects in databases. *GeoInformatica*, 3(3):269–296, 1999.
78. M. Erwig and M. Schneider. Partition and conquer. In S. C. Hirtle and A. U. Frank, editors, *Proceedings of the International Conference on Spatial Information Theory, COSIT 1997*, volume 1329 of *LNCS*, pages 389–407, Laurel Highlands, Pennsylvania, USA, Oct. 1997. Springer.

79. M. Erwig and M. Schneider. Developments in spatio-temporal query languages. In *DEXA Workshop*, pages 441–449, 1999.
80. M. Erwig and M. Schneider. The honeycomb model of spatio-temporal partitions. In M. H. Böhlen, C. S. Jensen, and M. Scholl, editors, *Spatio-Temporal Database Management, International Workshop STDBM'99, Proceedings*, volume 1678 of *Lecture Notes in Computer Science*, pages 39–59, Edinburgh, Scotland, Sept. 10-11, 1999. Springer.
81. J. Euzenat. An algebraic approach to granularity in qualitative time and space representation. In *IJCAI (1)*, pages 894–900, 1995.
82. G. Faria, C. B. Medeiros, and M. A. Nascimento. An extensible framework for spatio-temporal database applications. In M. Rafanelli and M. Jarke, editors, *10th International Conference on Scientific and Statistical Database Management, Proceedings, Capri, Italy, July 1-3, 1998*, pages 202–205. IEEE Computer Society, 1998.
83. U. M. Fayyad, G. G. Grinstein, and A. Wierse. *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann, 2000.
84. I. D. Fent, D. Gubiani, and A. Montanari. Granular geograph: a multi-granular conceptual model for spatial data. In A. Calì, D. Calvanese, E. Franconi, M. Lenzerini, and L. Tanca, editors, *SEBD*, pages 368–379, 2005.
85. F. Fonseca, M. Egenhofer, C. Davis, and G. Câmara. Semantic granularity in ontology-driven geographic information systems. *Ann. Math. Artif. Intell.*, 36(1–2):121–151, 2002.
86. L. Forlizzi, R. H. Güting, E. Nardelli, and M. Schneider. A data model and data structures for moving objects databases. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*, pages 319–330. ACM, 2000.
87. M. Franceschet. *Dividing and Conquering the Layered Land*. PhD thesis, Dipartimento di Matematica e Informatica, Università di Udine, Italy, 2001.
88. M. Franceschet and A. Montanari. Temporalized logics and automata for time granularity. *TPLP*, 4(5-6):621–658, 2004.
89. J. R. Francica. Location-based services: Practices and products. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 623–627. Springer, 2008.
90. A. U. Frank. Qualitative spatial reasoning with cardinal directions. In H. Kaindl, editor, *Proc. 7th Austrian Conference on Artificial Intelligence, ÖGAI-91, Wien, 24.-27. September 1991*, volume 287 of *Informatik-Fachberichte*, pages 157–167. Springer, 1991.
91. A. U. Frank. Map algebra extended with functors for temporal data. In *ER (Workshops)*, volume 3770 of *Lecture Notes in Computer Science*, pages 194–207. Springer, 2005.
92. C. Freksa. Using orientation information for qualitative spatial reasoning. In A. U. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space, International Conference GIS - From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning, Pisa, Italy, September 21-23, 1992, Proceedings*, volume 639 of *Lecture Notes in Computer Science*, pages 162–178. Springer, 1992.
93. S. K. Gadia and V. Chopra. A relational model and sql-like query language for spatial databases. In N. R. Adam and B. K. Bhargava, editors, *Advanced Database Systems*, volume 759 of *Lecture Notes in Computer Science*. Springer, 1993.
94. V. Gandhi. Vector data. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 1217–1221. Springer, 2008.
95. F. Geerts, S. Haesevoets, and B. Kuijpers. First-order complete and computationally complete query languages for spatio-temporal databases. *ACM Transactions on Computational Logic*, 9(2):13:1–13:??, Mar. 2008.

96. S. W. Golomb. Run-length encodings. *IEEE Trans. Inform. Theory*, IT-12:399–401, 1966.
97. P. Goovaerts. Spatial uncertainty in medical geography: A geostatistical perspective. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 1106–1112. Springer, 2008.
98. R. K. Goyal. *Similarity assessment for cardinal directions between extended spatial objects*. PhD thesis, 2000. AAI9972143.
99. S. Greenland and J. Robins. Invited commentary: ecologic studies–biases, misconceptions, and counterexamples. *American Journal of Epidemiology*, 139(8):747, 1994.
100. H. Gregersen and C. S. Jensen. Conceptual modeling of time-varying information, 1998.
101. H. Gregersen, L. Mark, and C. S. Jensen. Mapping temporal ER diagrams to relational schemas. Technical Report TR-39, Aalborg Universitetsforlag, 1998.
102. L. Grigoletti, G. Perini, A. Rossi, A. Biggeri, C. Barbui, M. Tansella, and F. Amaddeo. Mortality and cause of death among psychiatric patients: a 20-year case-register study in an area with a community-based system of care. *Psychological medicine*, 39(11):1875–1884, 2009.
103. W. Gruber. *Modeling and Transformation of Workflows with Temporal Constraints*. Ios Pr Inc, City, 2004.
104. R. H. Güting. Geo-relational algebra: A model and query language for geometric database systems. In J. W. Schmidt, S. Ceri, and M. Missikoff, editors, *Advances in Database Technology - EDBT'88, Proceedings of the International Conference on Extending Database Technology, Venice, Italy, March 14-18, 1988*, volume 303 of *Lecture Notes in Computer Science*, pages 506–527. Springer, 1988.
105. R. H. Güting. An introduction to spatial database systems. *VLDB J.*, 3(4):357–399, 1994.
106. R. H. Güting. Moving object languages. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 732–740. Springer, 2008.
107. R. H. Güting. Moving objects databases and tracking. In L. Liu and M. T. Özsu, editors, *Encyclopedia of Database Systems*, pages 1770–1776. Springer US, 2009.
108. R. H. Güting, M. H. Böhlen, M. Erwig, C. S. Jensen, N. A. Lorentzos, M. Schneider, and M. Vazirgiannis. A foundation for representing and querying moving objects. *ACM T. Database Syst.*, 25(1):1–42, 2000.
109. R. H. Güting and M. Schneider. Realm-based spatial data types: The rose algebra. *VLDB J.*, 4(2):243–286, 1995.
110. A. Guttman. R-Trees: A dynamic index structure for spatial searching. In S. B. Navathe, editor, *Proceedings of the 10th ACM International Conference on Management of Data (SIGMOD)*, pages 47–57. ACM Press, 1985.
111. M. Halbey-Martin. GRASS. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 413–418. Springer, 2008.
112. S. Hay, C. Tucker, D. Rogers, and M. Packer. Remotely sensed surrogates of meteorological data for the study of the distribution and abundance of arthropod vectors of disease. *Annals of Tropical Medicine and Parasitology*, 90(1):1–19, 1996.
113. M. Hazewinkel. *Encyclopaedia of Mathematics*. Kluwer Academic Publishers, Boston, 2002.
114. P. Helland. Database management system. In L. Liu and M. T. Özsu, editors, *Encyclopedia of Database Systems*, pages 714–719. Springer US, 2009.
115. H. Henderson. *Encyclopedia of Computer Science and Technology*. Facts on File, New York, 2008.
116. I. Hertz-Picciotto. *Environmental Epidemiology*, chapter 30 of Modern Epidemiology. Lippincott Williams & Wilkins, 3rd edition, 2008.

117. K. Hornsby and M. Egenhofer. Identity-based change: a foundation for spatio-temporal knowledge representation. *International Journal of Geographical Information Science*, 14(3):207–224, 2000.
118. K. S. Hornsby and K. King. Modeling motion relations for moving objects on road networks. *GeoInformatica*, 12(4):477–495, 2008.
119. B. Huang and C. Claramunt. STOQL: An ODMG-based spatio-temporal object model and query language, 2007.
120. IBM. DB2. URL: http://www.ibm.com/software/data/db2/.
121. Imperial College London. EUROHEIS, 2003. URL: http://www.euroheis.org/archive/default.html.
122. International Organization for Standardization. ISO/IEC 9075-14:2008 Information technology - Database languages - SQL - Part 14: XML-Related Specifications (SQL/XML). URL: http://www.iso.org.
123. I. International Organization for Standardization. *ISO/IEC 13249-3:2006: Information technology — Database languages — SQL Multimedia and Application Packages — Part 3: Spatial*. 2006.
124. C. S. Jensen, C. E. Dyreson, M. H. Böhlen, J. Clifford, R. Elmasri, S. K. Gadia, F. Grandi, P. J. Hayes, S. Jajodia, W. Käfer, N. Kline, N. A. Lorentzos, Y. G. Mitsopoulos, A. Montanari, D. A. Nonen, E. Peressi, B. Pernici, J. F. Roddick, N. L. Sarda, M. R. Scalas, A. Segev, R. T. Snodgrass, M. D. Soo, A. U. Tansel, P. Tiberio, and G. Wiederhold. The consensus glossary of temporal database concepts - february 1998 version. In *Temporal Databases, Dagstuhl*, pages 367–405, 1998.
125. P. Jin and P. Sun. Ostm: A spatiotemporal extension to oracle. In J. Kim, D. Delen, J. Park, F. Ko, and Y. J. Na, editors, *NCM 2008, The Fourth International Conference on Networked Computing and Advanced Information Management, Gyeongju, Korea, September 2-4, 2008 - Volume 2*, pages 575–580. IEEE Computer Society, 2008.
126. I. Kamel. Indexing, Hilbert R-Tree, spatial indexing, multimedia indexing. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 507–512. Springer, 2008.
127. J. S. Kaufman. *Social Epidemiology*, chapter 26 of Modern Epidemiology. Lippincott Williams & Wilkins, 3rd edition, 2008.
128. E. T. Keravnou. A multidimensional and multigranular model of time for medical knowledge-based systems. *J. Intell. Inf. Syst.*, 13(1-2):73–120, 1999.
129. V. Khatri, S. Ram, and R. T. Snodgrass. Augmenting a conceptual model with geospatiotemporal annotations. *IEEE Trans. Knowl. Data Eng*, 16(11):1324–1338, 2004.
130. V. Khatri, S. Ram, R. T. Snodgrass, and G. M. O'Brien. Supporting user-defined granularities in a spatiotemporal conceptual model. *Ann. Math. Artif. Intell*, 36(1-2), 2002.
131. D. H. Kim, K. H. Ryu, and H. S. Kim. A spatiotemporal database model and query language. *Journal of Systems and Software*, 55(2):129–149, 2000.
132. M. Koubarakis, T. K. Sellis, A. U. Frank, S. Grumbach, R. H. Güting, C. S. Jensen, N. A. Lorentzos, Y. Manolopoulos, E. Nardelli, B. Pernici, H.-J. Schek, M. Scholl, B. Theodoulidis, and N. Tryfona, editors. *Spatio-Temporal Databases: The CHOROCHRONOS Approach*, volume 2520 of *Lecture Notes in Computer Science*. Springer, 2003.
133. G. E. Langran. *Time in geographic information systems*. PhD thesis, University of Washington, Seattle, WA, USA, 1989. UMI Order No: GAX90-00269.
134. R. Laurini, L. Paolino, M. Sebillo, G. Tortora, and G. Vitiello. A spatial sql extension for continuous field querying. In *28th International Computer Software and Applications Conference (COMPSAC 2004), Design and Assessment of Trustworthy*

Software-Based Systems, 27-30 September 2004, Hong Kong, China, Proceedings, pages 78–81. IEEE Computer Society, 2004.
135. J. Lee, G. Petersen, R. Stevens, and K. Vesanen. The influence of age, year of birth, and date on mortality from malignant melanoma in the populations of England and Wales, Canada, and the white population of the United States. *American Journal of Epidemiology*, 110(6):734, 1979.
136. J. A. C. Lema, L. Forlizzi, R. H. Güting, E. Nardelli, and M. Schneider. Algorithms for moving objects databases. *Comput. J.*, 46(6):680–712, 2003.
137. G. Ligozat. Reasoning about cardinal directions. *J. Vis. Lang. Comput.*, 9(1):23–44, 1998.
138. H. Lim. Raster data. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 949–955. Springer, 2008.
139. U. W. Lipeck and K. Neumann. Modelling and manipulating objects in geoscientific databases. In S. Spaccapietra, editor, *Entity-Relationship Approach: Ten Years of Experience in Information Modeling, Proceedings of the Fifth International Conference on Entity-Relationship Approach, Dijon, France, November 17-19, 1986*, pages 67–85. North-Holland, 1987.
140. E. Malinowski and E. Zimányi. *Advanced data warehouse design: From conventional to spatial and temporal applications*. Springer, 2008.
141. H. Mannila, H. Toivonen, and A. I. Verkamo. Discovering frequent episodes in sequences. In *KDD*, pages 210–215, 1995.
142. R. Martinez, M. Vidaurre, P. Najera, E. Loyola, C. Castillo-Salgado, and C. Eisner. SIGEpi: geographic information system in epidemiology and public health. *Epidemiol Bull*, 22(3):4–5, Sept 2001.
143. T. Mason, F. MacKay, and R. Hoover. *Atlas of cancer mortality for US counties, 1950-1969*. US Dept. of Health, Education, and Welfare, Public Health Service, National Institutes of Health, 1975.
144. W. C. McGee. The information management system IMS/VS. *IBM Systems Journal*, 16(2), 1977.
145. M. McKenney and M. Schneider. Spatial partition graphs: A graph theoretic model of maps. In D. Papadias, D. Zhang, and G. Kollios, editors, *Proceedings of the 10th International Symposium on Spatial and Temporal Databases, SSTD 2007*, volume 4605 of *Lecture Notes in Computer Science*, pages 167–184, Boston, Massachussets, USA, July 2007. Springer.
146. J. Mennis and C. D. Tomlin. Cubic map algebra functions for spatio-temporal analysis. *Cartogr. and Geogr. Inform.*, 32(1):17–32, 2005.
147. Microsoft Corporation. SQL server. URL: http://www.microsoft.com/sqlserver/2008/.
148. A. Montanari. *Metric and Layered Temporal Logic for Time Granularity*. PhD thesis, ILLC Dissertation Series 1996-02, University of Amsterdam, 1996.
149. A. Montanari, A. Peron, and A. Policriti. Theories of omega-layered metric temporal structures: Expressiveness and decidability. *Logic Journal of the IGPL*, 7(1):79–102, 1999.
150. A. Montanari and A. Policriti. Decidability results for metric and layered temporal logics. *Notre Dame Journal of Formal Logic*, 37(2):260–282, 1996.
151. H. Morgenstern. *Ecologic Studies*, chapter 25 of Modern Epidemiology. Lippincott Williams & Wilkins, 3rd edition, 2008.
152. M. M. Moro, N. Edelweiss, A. P. Zaupa, and C. S. dos Santos. Tvql - temporal versioned query language. In A. Hameurlain, R. Cicchetti, and R. Traunmüller, editors, *Database and Expert Systems Applications, 13th International Conference, DEXA 2002, Aix-en-Provence, France, September 2-6, 2002, Proceedings*, volume 2453 of *Lecture Notes in Computer Science*, pages 618–627. Springer, 2002.

153. E. Mota and D. Robertson. Representing interaction of agents at different time granularities. In *TIME '96: Proceedings of the 3rd Workshop on Temporal Representation and Reasoning (TIME'96)*, page 72, Washington, DC, USA, 1996. IEEE Computer Society.
154. P. Ning, X. S. Wang, and S. Jajodia. An algebraic representation of calendars. *Ann. Math. Artif. Intel.*, 36(1–2):5–38, 2002.
155. Object Data Management Group. ODMG Standard. URL: http://www.odbms.org/odmg.html.
156. Open Geospatial Consortium Inc. OpenGIS Specifications. URL: http://www.opengeospatial.org/.
157. S. Openshaw. Ecological fallacies and the analysis of areal census data. *Environment and Planning A*, 16(1):17–31, 1984.
158. Oracle. MySQL. URL: http://www.mysql.com/.
159. Oracle Corporation. Oracle database. URL: http://www.oracle.com.
160. Oracle Corporation. Oracle spatial. URL: http://www.oracle.com.
161. Oracle Corporation. Oracle database 11g workspace manager overview, September 2009.
162. C. Parent, S. Spaccapietra, and E. Zimányi. Spatio-temporal conceptual models: Data structures + space + time. In C. B. Medeiros, editor, *ACM-GIS*, pages 26–33. ACM, 1999.
163. N. Pearce and F. Merletti. Complexity, simplicity and epidemiology. *Int J Epidemiol*, 35(3):515–519, 2006.
164. R. Pertile, V. Donisi, L. Grigoletti, A. Angelozzi, G. Zamengo, G. Zulian, and F. Amaddeo. Drgs and other patient-, service-and area-level factors influencing length of stay in acute psychiatric wards: the veneto region experience. *Social Psychiatry and Psychiatric Epidemiology*, pages 1–10.
165. D. Pfeiffer, M. Stevenson, T. Robinson, K. Stevens, and D. Rogers. *Spatial analysis in epidemiology*. Oxford University Press, USA, 2008.
166. M. Poesio and R. J. Brachman. Metric constraints for maintaining appointments: Dates and repeated activities. In *AAAI*, pages 253–259, 1991.
167. M. Porta. *A Dictionary of Epidemiology*. Oxford University Press, Oxford Oxfordshire, 2008.
168. PostgreSQL Global Development Group. PostgreSQL. URL: http://www.postgresql.org.
169. G. Pozzani and E. Zimányi. Defining spatio-temporal granularities for raster data. In L. MacKinnon, editor, *Proceedings of the 27th International Information Systems Conference (BNCOD 2010)*, volume 6121 of *Lecture Notes in Computer Science*, Dundee, Scotland, June 2010. Springer.
170. S. Ram. Intelligent database design using the unifying semantic model. *Inf. Manage.*, 29(4):191–206, 1995.
171. D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *KR*, pages 165–176, 1992.
172. Refractions Research. PostGIS. URL: http://postgis.refractions.net.
173. L. Relly, H.-J. Schek, O. Henricsson, and S. Nebiker. Physical database design for raster images in concert. In M. Scholl and A. Voisard, editors, *Advances in Spatial Databases, 5th International Symposium, SSD'97, Berlin, Germany, July 15-18, 1997, Proceedings*, volume 1262 of *Lecture Notes in Computer Science*, pages 259–279. Springer, 1997.
174. E. Rose and A. Segev. Toosql - a temporal object-oriented query language. In R. Elmasri, V. Kouramajian, and B. Thalheim, editors, *Entity-Relationship Approach - ER'93, 12th International Conference on the Entity-Relationship Approach, Arlington, Texas, USA, December 15-17, 1993, Proceedings*, volume 823 of *Lecture Notes in Computer Science*, pages 122–136. Springer, 1993.

175. R. Rossato. *Temporal Functional Dependencies and Trends with multiple granularities*. PhD thesis, Dipartimento di Informatica, Università degli Studi di Verona, Italy, 2006.
176. A. Rossi, V. Morgan, F. Amaddeo, M. Sandri, M. Tansella, and A. Jablensky. Psychiatric out-patients seen once only in South Verona and Western Australia: a comparative case-register study. *Australian and New Zealand Journal of Psychiatry*, 39(5):414–422, 2005.
177. K. J. Rothman, S. Greenland, and T. L. Lash, editors. *Modern Epidemiology*. Lippincott Williams & Wilkins, 3rd edition, 2008.
178. K. J. Rothman, S. Greenland, and T. L. Lash. *Types of Epidemiologic Studies*, chapter 6 of Modern Epidemiology. Lippincott Williams & Wilkins, 3rd edition, 2008.
179. H. Sagan. *Space-Filling Curves*. Springer-Verlag, Berlin, 1994.
180. N. L. Sarda. Hsql: A historical query language. In *Temporal Databases*, pages 110–140. 1993.
181. H. R. Schmidtke and W. Woo. A size-based qualitative approach to the representation of spatial granularity. In M. M. Veloso, editor, *IJCAI 2007*, pages 563–568, 2007.
182. M. Schneider. *Spatial Data Types for Database Systems, Finite Resolution Geometry for Geographic Information Systems*, volume 1288 of *Lecture Notes in Computer Science*. Springer, 1997.
183. M. Schneider and T. Behr. Topological relationships between complex spatial objects. *ACM Trans. Database Syst*, 31(1):39–81, 2006.
184. A. A. Schuessler. Ecological inference. *Proc Natl Acad Sci USA*, 96(19):10578–10581, September 1999.
185. Y. Shahar. Dynamic temporal interpretation contexts for temporal abstraction. *Ann. Math. Artif. Intell.*, 22(1-2):159–192, 1998.
186. Y. Shahar and C. Combi. Timing is everything. Time-oriented clinical information systems. *Western Journal of Medicine*, 168(2):105, 1998.
187. S. Shekhar and H. Xiong, editors. *Encyclopedia of GIS*. Springer, 2008.
188. A. P. Sistla and O. Wolfson. Temporal triggers in active databases. *IEEE Trans. Knowl. Data Eng.*, 7(3):471–486, 1995.
189. A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Modeling and querying moving objects. In W. A. Gray and P.-Å. Larson, editors, *Proceedings of the Thirteenth International Conference on Data Engineering, April 7-11, 1997 Birmingham U.K*, pages 422–432. IEEE Computer Society, 1997.
190. A. P. Sistla and C. T. Yu. Reasoning about qualitative spatial relationships. *J. Autom. Reasoning*, 25(4):291–328, 2000.
191. S. Skiadopoulos and M. Koubarakis. Composing cardinal direction relations. *Artif. Intell.*, 152(2):143–171, 2004.
192. Small Area Health Statistics Unit (SAHSU) at Imperial College London. Rapid Inquiry Facility, 1999. URL: http://www.sahsu.org/sahsu_related_studies.htm#RIF.
193. B. Smith and B. Brogaard. Quantum mereotopology. *Ann. Math. Artif. Intell*, 36(1-2):153–175, 2002.
194. R. T. Snodgrass, editor. *The TSQL2 Temporal Query Language*. Kluwer, 1995.
195. J. Snow. *Snow on cholera*. Hafner, New York, 1965.
196. Spatial Analysis Lab at University of Illinois. GeoDa, 2003. URL: https://www.geoda.uiuc.edu/.
197. J. Stell and M. Worboys. Stratified map spaces: a formal basis for multi-resolution spatial databases. In T. Poiker and N. Chrisman, editors, *Proceedings of Eighth International Symposium on Spatial Data Handling*, pages 180–189, Vancouver, Canada, 1998.

198. J. Tello, M. Mazzi, M. Tansella, P. Bonizzato, J. Jones, and F. Amaddeo. Does socioeconomic status affect the use of community-based psychiatric services? A south Verona case register study. *Acta Psychiatrica Scandinavica*, 112(3):215–223, 2005.
199. J. E. Tello, J. Jones, P. Bonizzato, M. Mazzi, F. Amaddeo, and M. Tansella. A census-based socio-economic status (ses) index as a tool to examine the relationship between mental health services use and deprivation. *Social Science & Medicine*, 61(10):2096 – 2105, 2005.
200. F. TF, K. RF, V. RB, M. Boisen, H. FM, N. JA, and E. Tolchinsky. An evaluation of the 1954 poliomyelitis vaccine trials. *Am J Public Health*, 45:1–63, 1955.
201. B. Theodoulidis, A. Ait-Braham, and G. Karvelis. The ores temporal dbms and the ert-sql query language. In D. Karagiannis, editor, *Database and Expert Systems Applications, 5th International Conference, DEXA '94, Athens, Greece, September 7 - 9, 1994, Proceedings*, volume 856 of *Lecture Notes in Computer Science*, pages 270–279. Springer, 1994.
202. W. Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 133–192. MIT Press, Cambridge, MA, USA, 1990.
203. I. Timko, M. H. Böhlen, and J. Gamper. Sequenced spatio-temporal aggregation in road networks. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '09, pages 48–59, New York, NY, USA, 2009. ACM.
204. C. D. Tomlin and J. Berry. A mathematical structure for cartographic modeling in environmental analysis. In *Proceedings of the American Congress on Surveying and Mapping*, pages 269–283, 1979.
205. N. Tryfona and C. S. Jensen. Conceptual data modeling for spatiotemporal applications. *GeoInformatica*, 3(3), 1999.
206. D. C. Tsichritzis and F. H. Lochovsky. Hierarchical data-base management: A survey. *ACM Comput. Surv.*, 8:105–123, March 1976.
207. United States Department of Health, Education and Welfare. Smoking and health: report of the Advisory Committee to the Surgeon General of the Public Health Service, 1964. PHS Publ No 1103.
208. M. Vasardani and M. J. Egenhofer. Single-holed regions: Their relations and inferences. In T. J. Cova, H. J. Miller, K. Beard, A. U. Frank, and M. F. Goodchild, editors, *Geographic Information Science, 5th International Conference, GIScience 2008.*, volume 5266 of *LNCS*, pages 337–353, Park City, UT, USA, 23-26 2008. Springer.
209. J. R. R. Viqueira and N. A. Lorentzos. SQL extension for spatio-temporal data. *VLDB J.*, 16(2):179–200, 2007.
210. S. Šaltenis. Indexing the positions of continuously moving objects. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 538–543. Springer, 2008.
211. A. Wakefield, S. Murch, A. Anthony, J. Linnell, D. Casson, M. Malik, M. Berelowitz, A. Dhillon, M. Thomson, P. Harvey, et al. RETRACTED: Ileal-lymphoid-nodular hyperplasia, non-specific colitis, and pervasive developmental disorder in children. *The Lancet*, 351(9103):637–641, 1998.
212. S. D. Walter. *Spatial Epidemiology: Methods and Applications*, chapter Disease mapping: a historical perspective, pages 223–252. Oxford University Press, USA, 2001.
213. K. Wang, C. Fierbinteanu, and M. Maekawa. A conceptual framework for spatiotemporal data modeling. In V. Marík, W. Retschitzegger, and O. Stepánková, editors, *Database and Expert Systems Applications, 14th International Conference, DEXA 2003, Prague, Czech Republic, September 1-5, 2003, Proceedings*, volume 2736 of *Lecture Notes in Computer Science*, pages 57–66. Springer, 2003.

214. S. Wang and D. Liu. Spatio-temporal database with multi-granularities. In Q. Li, G. Wang, and L. Feng, editors, *Proceedings of the 5th International Conference on Web-Age Information Management, WAIM 2004*, volume 3129 of *LNCS*, pages 137–146, Dalian, China, July 2004. Springer.
215. X. S. Wang, C. Bettini, A. Brodsky, and S. Jajodia. Logical design for temporal databases with multiple granularities. *ACM Transactions on Database Systems*, 22(2):115–170, June 1997.
216. G. Weiderhold, S. Jajodia, and W. Litwin. *Integrating Temporal Data in a Heterogeneous Environment*, chapter 22, pages 563–579. Benjamin/Cummings, 1993.
217. S. Wiebrock, L. Wittenburg, U. Schmid, and F. Wysotzki. Inference and visualization of spatial relations. In C. Freksa, W. Brauer, C. Habel, and K. F. Wender, editors, *Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, volume 1849 of *Lecture Notes in Computer Science*, pages 212–224. Springer, 2000.
218. J. Wijsen. Reasoning about qualitative trends in databases. *Inf. Syst*, 23(7):463–487, 1998.
219. J. Wijsen. Temporal FDs on complex objects. *ACM Transactions on Database Systems*, 24(1):127–176, Mar. 1999.
220. J. Wijsen. A string-based model for infinite granularities. In *Proceedings of the AAAI Workshop on Spatial and Temporal Granularities*, pages 9–16. AAAI Press, 2000.
221. J. Wijsen. Trends in databases: Reasoning and mining. *IEEE Trans. Knowl. Data Eng*, 13(3):426–438, 2001.
222. O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez. Cost and imprecision in modeling the position of moving objects. In *Proceedings of the Fourteenth International Conference on Data Engineering, February 23-27, 1998, Orlando, Florida, USA*, pages 588–596. IEEE Computer Society, 1998.
223. O. Wolfson, A. P. Sistla, S. Chamberlain, and Y. Yesha. Updating and querying databases that track mobile units. *Distributed and Parallel Databases*, 7(3):257–387, 1999.
224. S. K. J. Wood. Plane sweep algorithm. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 876–878. Springer, 2008.
225. M. F. Worboys. A unified model for spatial and temporal information. *Comput. J.*, 37(1):36–34, 1994.
226. M. F. Worboys. Computation with imprecise geospatial data. *Computers, Environment and Urban Systems*, 22(2):85 – 106, 1998.
227. M. F. Worboys. Imprecision in finite resolution spatial data. *GeoInformatica*, 2(3):257–279, 1998.
228. M. F. Worboys and P. Bofakos. A canonical model for a class of areal spatial objects. In D. J. Abel and B. C. Ooi, editors, *Advances in Spatial Databases, Third International Symposium, SSD'93, Singapore, June 23-25, 1993, Proceedings*, volume 692 of *Lecture Notes in Computer Science*, pages 36–52. Springer, 1993.
229. World Health Organization. HealthMapper, 1999. URL: http://www.who.int/health_mapping/tools/healthmapper/en/.
230. World Health Organization (WHO). International classification of diseases (ICD). URL: http://www.who.int/classifications/icd/en/.
231. M. Yuan. Temporal GIS and applications. In S. Shekhar and H. Xiong, editors, *Encyclopedia of GIS*, pages 1147–1150. Springer, 2008.
232. X. Zhang, W. Liu, S. Li, and M. Ying. Reasoning about cardinal directions between extended objects. *CoRR*, abs/0909.0138, 2009.
233. K. Zimmermann. Enhancing qualitative spatial reasoning - combining orientation and distance. In *COSIT*, pages 69–76, 1993.

234. G. Zulian. An atlas of Verona psychiatric services. Section of Psychiatry and Clinical Psychology, Department of Medicine and Public Health, University of Verona, Italy. URL: http://www.psychiatry.univr.it/page_geo.
235. G. Zulian, V. Donisi, G. Secco, R. Pertile, M. Tansella, and F. Amaddeo. How are caseload and service utilisation of psychiatric services influenced by distance? a geographical approach to the study of community-based mental health services. *Social Psychiatry and Psychiatric Epidemiology*, pages 1–11, 2010.

Appendices

A

Semantics of relationships between granularities

$$\bigwedge_{R\in\{GI,FT,SG,P,GPI,SE\}} .R(G_1,G_2)\wedge R(G_2,G_3)\vDash^{\mathcal{M},\lambda} R(G_1,G_3) \quad \text{(Ttrans)}$$

$$\bigwedge_{R\in\{GI,FT,SG,P,GPI,SE\}} .\vDash^{\mathcal{M},\lambda} R(G,G) \quad \text{(Trefl)}$$

$$SE(G_1,G_2)\vDash^{\mathcal{M},\lambda} SE(G_2,G_1) \quad \text{(TsymmSE)}$$

$$\bigwedge_{R\in\{GI,FT,SG,P,GPI\}} .R(G_1,G_2)\wedge R(G_2,G_1)\vDash^{\mathcal{M},\lambda} G_1\approx G_2 \quad \text{(Tantisymm)}$$

$$P(G_1,G_2)\vDash^{\mathcal{M},\lambda} GI(G_1,G_2) \quad \text{(TEP1)}$$

$$P(G_1,G_2)\vDash^{\mathcal{M},\lambda} FT(G_1,G_2) \quad \text{(TEP2)}$$

$$GI(G_1,G_2)\wedge FT(G_1,G_2)\vDash^{\mathcal{M},\lambda} P(G_1,G_2) \quad \text{(TIP)}$$

$$GPI(G_1,G_2)\vDash^{\mathcal{M},\lambda} GP(G_1,G_2) \quad \text{(TIGP)}$$

$$SG(G_1,G_2)\vDash^{\mathcal{M},\lambda} FT(G_1,G_2) \quad \text{(TESG1)}$$

$$SG(G_1,G_2)\vDash^{\mathcal{M},\lambda} GI(G_2,G_1) \quad \text{(TESG2)}$$

$$GI(G_1,G_2)\wedge SG(G_1,G_2)\vDash^{\mathcal{M},\lambda} SE(G_1,G_2) \quad \text{(TISE)}$$

$$\bigwedge_{R\in\{GI,FT,SG,P,CB\}} .SE(G_1,G_2)\vDash^{\mathcal{M},\lambda} R(G_1,G_2) \quad \text{(TESE)}$$

$$R(G_1,G_2)\wedge SE(G_1,G_3)\vDash^{\mathcal{M},\lambda} R(G_3,G_2) \quad \text{(Tmon1)}$$

$$R(G_1,G_2)\wedge SE(G_2,G_3)\vDash^{\mathcal{M},\lambda} R(G_1,G_3) \quad \text{(Tmon2)}$$

Fig. A.1: Semantics of temporal relationships

$$\bigwedge_{R\in\{GI,FT,SG,P,CB\}} .R(G_1,G_2)\wedge R(G_2,G_3)\vDash^{\mathcal{M},\lambda} R(G_1,G_3) \quad \text{(Strans)}$$

$$\bigwedge_{R\in\{GI,FT,SG,P,CB\}} .\vDash^{\mathcal{M},\lambda} R(G,G) \quad \text{(Srefl)}$$

$$\bigwedge_{R\in\{D,O\}} .\vDash^{\mathcal{M},\lambda} \neg R(G,G) \quad \text{(Santirefl)}$$

$$\bigwedge_{R\in\{D,O\}} .R(G_1,G_2)\vDash^{\mathcal{M},\lambda} R(G_2,G_1) \quad \text{(Ssymm O/D)}$$

$$\bigwedge_{R\in\{GI,FT,SG,P\}} .R(G_1,G_2)\wedge R(G_2,G_1)\vDash^{\mathcal{M},\lambda} G_1\approx G_2 \quad \text{(Santisymm)}$$

$$P(G_1,G_2)\vDash^{\mathcal{M},\lambda} GI(G_1,G_2) \quad \text{(SEP1)}$$

$$P(G_1,G_2)\vDash^{\mathcal{M},\lambda} FT(G_1,G_2) \quad \text{(SEP2)}$$

$$GI(G_1,G_2)\wedge FT(G_1,G_2)\vDash^{\mathcal{M},\lambda} P(G_1,G_2) \quad \text{(SIP)}$$

$$\bigwedge_{R\in\{GI,FT,SG,P,CB,O\}} .D(G_1,G_2)\vDash^{\mathcal{M},\lambda} \neg R(G_1,G_2) \quad \text{(SED)}$$

$$\bigwedge_{R\in\{GI,FT,SG,P,CB,D\}} .O(G_1,G_2)\vDash^{\mathcal{M},\lambda} \neg R(G_1,G_2) \quad \text{(SEO)}$$

$$FT(G_1,G_2)\vDash^{\mathcal{M},\lambda} CB(G_1,G_2) \quad \text{(SEFT)}$$

$$SG(G_1,G_2)\vDash^{\mathcal{M},\lambda} FT(G_1,G_2) \quad \text{(SESG1)}$$

$$GI(G_1,G_2)\vDash^{\mathcal{M},\lambda} CB(G_2,G_1) \quad \text{(SEGI)}$$

$$SG(G_1,G_2)\vDash^{\mathcal{M},\lambda} GI(G_2,G_1) \quad \text{(SESG2)}$$

$$GI(G_1,G_2)\wedge CB(G_1,G_2)\vDash^{\mathcal{M},\lambda} FT(G_1,G_2) \quad \text{(SIFT)}$$

$$CB(G_1,G_2)\wedge CB(G_2,G_3)\vDash^{\mathcal{M},\lambda} CB(G_1,G_3) \quad \text{(Sconc1)}$$

$$CB(G_1,G_2)\wedge D(G_2,G_3)\vDash^{\mathcal{M},\lambda} D(G_1,G_3) \quad \text{(Sconc2)}$$

$$GI(G_1,G_2)\wedge D(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg CB(G_1,G_3) \quad \text{(Sconc3)}$$

$$D(G_1,G_2)\wedge CB(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg GI(G_1,G_3) \quad \text{(Sconc4)}$$

$$GI(G_1,G_2)\wedge O(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg FT(G_1,G_3) \quad \text{(Sconc5)}$$

$$GI(G_1,G_2)\wedge O(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg D(G_1,G_3) \quad \text{(Sconc6)}$$

$$P(G_1,G_2)\wedge O(G_2,G_3)\vDash^{\mathcal{M},\lambda} O(G_1,G_3) \quad \text{(Sconc7)}$$

$$D(G_1,G_2)\wedge O(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg GI(G_1,G_3) \quad \text{(Sconc8)}$$

$$O(G_1,G_2)\wedge GI(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg FT(G_1,G_3) \quad \text{(Sconc9)}$$

$$O(G_1,G_2)\wedge P(G_2,G_3)\vDash^{\mathcal{M},\lambda} O(G_1,G_3) \quad \text{(Sconc10)}$$

$$O(G_1,G_2)\wedge D(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg FT(G_1,G_3) \quad \text{(Sconc11)}$$

$$O(G_1,G_2)\wedge CB(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg GI(G_1,G_3) \quad \text{(Sconc12)}$$

$$O(G_1,G_2)\wedge CB(G_2,G_3)\vDash^{\mathcal{M},\lambda} \neg D(G_1,G_3) \quad \text{(Sconc13)}$$

Fig. A.2: Semantics of spatial relationships. First part.

$$GI(G_1, G_2) \wedge SG(G_1, G_2) \models^{\mathcal{M},\lambda} G_1 \approx G_2 \quad \text{(SeqI1)}$$

$$SG(G_1, G_2) \wedge CB(G_2, G_1) \models^{\mathcal{M},\lambda} G_1 \approx G_2 \quad \text{(SeqI2)}$$

$$G_1 \approx G_2 \wedge G_2 \approx G_3 \models^{\mathcal{M},\lambda} G_1 \approx G_3 \quad \text{(SeqTrans)}$$

$$\models^{\mathcal{M},\lambda} G \approx G \quad \text{(SeqRefl)}$$

$$G_1 \approx G_2 \models^{\mathcal{M},\lambda} G_2 \approx G_1 \quad \text{(SeqSymm)}$$

$$\bigwedge_{R \in \{GI,FT,SG,P,CB\}} .G_1 \approx G_2 \models^{\mathcal{M},\lambda} R(G_1, G_2) \quad \text{(SeqE)}$$

$$R(G_1, G_2) \wedge G_1 \approx G_3 \models^{\mathcal{M},\lambda} R(G_3, G_2) \quad \text{(Smon1)}$$

$$R(G_1, G_2) \wedge G_2 \approx G_3 \models^{\mathcal{M},\lambda} R(G_1, G_3) \quad \text{(Smon2)}$$

Fig. A.3: Semantics of spatial relationships. Second part.

$$QR(stG, stH) \models QR'(stG, stH) \qquad \text{where } R(G, H) \models R'(G, H) \quad \text{(STspace)}$$

$$\bigwedge_{Q \in \{\forall\forall,\forall\exists,\exists\forall,\exists\exists\}} .\forall\forall R(stG, stH) \models QR(stG, stH) \quad \text{(STallall)}$$

$$\bigwedge_{Q \in \{\forall\exists,\exists\exists\}} .\forall\exists R(stG, stH) \models QR(stG, stH) \quad \text{(STallexists)}$$

$$\bigwedge_{Q \in \{\exists\forall,\exists\exists\}} .\exists\forall R(stG, stH) \models QR(stG, stH) \quad \text{(STexistsall)}$$

$$\forall\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge GI(stG.tG, tG') \models \forall\forall R(stG', stH) \quad \text{(STrel1)}$$

$$\forall\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge SG(stG.tG, tG') \models \exists\forall R(stG', stH) \quad \text{(STrel2)}$$

$$\forall\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge FT(stG.tG, tG') \models \exists\exists R(stG', stH) \quad \text{(STrel3)}$$

$$\forall\exists R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge GI(stG.tG, tG') \models \forall\exists R(stG', stH) \quad \text{(STrel4)}$$

$$\forall\exists R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge FT(stG.tG, tG') \models \exists\exists R(stG', stH) \quad \text{(STrel5)}$$

$$\exists\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge FT(stG.tG, tG') \models \exists\exists R(stG', stH) \quad \text{(STrel6)}$$

$$\exists\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge SG(stG.tG, tG') \models \exists\forall R(stG', stH) \quad \text{(STrel7)}$$

$$\exists\exists R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge FT(stG.tG, tG') \models \exists\exists R(stG', stH) \quad \text{(STrel8)}$$

$$\forall\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge GI(tG', stG.tG) \models \exists\forall R(stG', stH) \quad \text{(STrel9)}$$

$$\forall\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge FT(tG', stG.tG) \models \forall\forall R(stG', stH) \quad \text{(STrel10)}$$

$$\forall\exists R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge GI(tG', stG.tG) \models \exists\exists R(stG', stH) \quad \text{(STrel11)}$$

$$\forall\exists R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge SG(tG', stG.tG) \models \forall\exists R(stG', stH) \quad \text{(STrel12)}$$

$$\exists\forall R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge GI(tG', stG.tG) \models \exists\forall R(stG', stH) \quad \text{(STrel13)}$$

$$\exists\exists R(stG, stH) \wedge stG' = \langle tG', stG.E \rangle \wedge GI(tG', stG.tG) \models \exists\exists R(stG', stH) \quad \text{(STrel14)}$$

Fig. A.4: Semantics of spatio-temporal relationships. First part.

$$\forall\forall R(stG, stH) \wedge GI(stG.tG, stH.tG) \vDash \forall\forall R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv1)}$$
$$\forall\forall R(stG, stH) \wedge FT(stG.tG, stH.tG) \vDash \exists\exists R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv2)}$$
$$\forall\forall R(stG, stH) \wedge SG(stG.tG, stH.tG) \vDash \exists\forall R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv3)}$$
$$\forall\exists R(stG, stH) \wedge GI(stG.tG, stH.tG) \vDash \forall\exists R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv4)}$$
$$\forall\exists R(stG, stH) \wedge FT(stG.tG, stH.tG) \vDash \exists\exists R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv5)}$$
$$\exists\forall R(stG, stH) \wedge GI(stG.tG, stH.tG) \vDash \exists\exists R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv6)}$$
$$\exists\forall R(stG, stH) \wedge FT(stG.tG, stH.tG) \vDash \exists\exists R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv7)}$$
$$\exists\forall R(stG, stH) \wedge SG(stG.tG, stH.tG) \vDash \exists\forall R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv8)}$$
$$\exists\exists R(stG, stH) \wedge GI(stG.tG, stH.tG) \vDash \exists\exists R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv9)}$$
$$\exists\exists R(stG, stH) \wedge FT(stG.tG, stH.tG) \vDash \exists\exists R'(stH, stG) \quad \text{where } R(G,H) \vDash R'(H,G) \quad \text{(STinv10)}$$

$$\forall\forall R(stG, stH) \wedge \forall\forall R'(stG, stH) \vDash \forall\forall R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair1)}$$
$$\forall\forall R(stG, stH) \wedge \forall\exists R'(stG, stH) \vDash \forall\exists R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair2)}$$
$$\forall\forall R(stG, stH) \wedge \exists\forall R'(stG, stH) \vDash \exists\forall R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair3)}$$
$$\forall\forall R(stG, stH) \wedge \exists\exists R'(stG, stH) \vDash \exists\exists R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair4)}$$
$$\forall\exists R(stG, stH) \wedge \forall\forall R'(stG, stH) \vDash \forall\exists R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair5)}$$
$$\forall\exists R(stG, stH) \wedge \exists\forall R'(stG, stH) \vDash \exists\exists R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair6)}$$
$$\exists\forall R(stG, stH) \wedge \forall\forall R'(stG, stH) \vDash \exists\forall R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair7)}$$
$$\exists\forall R(stG, stH) \wedge \forall\exists R'(stG, stH) \vDash \exists\exists R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair8)}$$
$$\exists\exists R(stG, stH) \wedge \forall\forall R'(stG, stH) \vDash \exists\exists R''(stG, stH) \quad \text{where } R(G,H) \wedge R'(G,H) \vDash R''(G,H) \quad \text{(STpair9)}$$

Fig. A.5: Semantics of spatio-temporal relationships. Second part.

$$\forall\forall R(stG, stH) \wedge \forall\forall R'(stH, stI) \vDash \forall\forall R''(stG, stI) \quad \text{where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc1}$$

$$\forall\forall R(stG, stH) \wedge \forall\exists R'(stH, stI) \wedge GI(stG.tG, stH.tG) \vDash \exists\exists R''(stG, stI) \text{ where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc2}$$

$$\forall\forall R(stG, stH) \wedge \forall\exists R'(stH, stI) \wedge SG(stG.tG, stH.tG) \vDash \forall\exists R''(stG, stI) \text{ where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc3}$$

$$\forall\exists R(stG, stH) \wedge \exists\forall R'(stH, stI) \wedge GI(stG.tG, stH.tG) \vDash \exists\forall R''(stG, stI) \text{ where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc4}$$

$$\forall\forall R(stG, stH) \wedge \exists\exists R'(stH, stI) \wedge GI(stG.tG, stH.tG) \vDash \exists\exists R''(stG, stI) \text{ where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc5}$$

$$\forall\exists R(stG, stH) \wedge \forall\forall R'(stH, stI) \vDash \forall\exists R''(stG, stI) \quad \text{where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc6}$$

$$\forall\exists R(stG, stH) \wedge \exists\forall R'(stH, stI) \wedge GI(stG.tG, stH.tG) \vDash \exists\exists R''(stG, stI) \quad \text{where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc7}$$

$$\exists\forall R(stG, stH) \wedge \forall\forall R'(stH, stI) \vDash \exists\forall R''(stG, stI) \quad \text{where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc8}$$

$$\exists\forall R(stG, stH) \wedge \forall\exists R'(stH, stI) \wedge SG(stG.tG, stH.tG) \vDash \exists\exists R''(stG, stI) \quad \text{where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc9}$$

$$\exists\exists R(stG, stH) \wedge \forall\forall R'(stH, stI) \vDash \exists\exists R''(stG, stI) \quad \text{where } R(G,H) \wedge R'(H,I) \vDash R''(G,I) \tag{STconc10}$$

Fig. A.6: Semantics of spatio-temporal relationships. Third part.

B

Inference rules

B.1 Inference rules for spatial relationships

$R(G_1, G_2)$	$?(G_1, G_2)$	
GroupsInto	GroupsInto	✓
	FinerThan	-
	Subgranularity	×
	Partition	-
	CoveredBy	-
	Disjoint	×
	Overlap	×
FinerThan	GroupsInto	-
	FinerThan	✓
	Subgranularity	-
	Partition	-
	CoveredBy	✓
	Disjoint	×
	Overlap	×
Subgranularity	GroupsInto	×
	FinerThan	✓
	Subgranularity	✓
	Partition	×
	CoveredBy	✓
	Disjoint	×
	Overlap	×
Partition	GroupsInto	✓
	FinerThan	✓
	Subgranularity	×
	Partition	✓
	CoveredBy	✓
	Disjoint	×
	Overlap	×

$R(G_1, G_2)$	$?(G_1, G_2)$	
CoveredBy	GroupsInto	-
	FinerThan	-
	Subgranularity	-
	Partition	-
	CoveredBy	✓
	Disjoint	×
	Overlap	×
Disjoint	GroupsInto	×
	FinerThan	×
	Subgranularity	×
	Partition	×
	CoveredBy	×
	Disjoint	✓
	Overlap	×
Overlap	GroupsInto	×
	FinerThan	×
	Subgranularity	×
	Partition	×
	CoveredBy	×
	Disjoint	×
	Overlap	✓

Table B.1: $\neg G_1 \approx G_2 \wedge R(G_1, G_2) \vdash ?(G_1, G_2)$

$R(G_1, G_2)$	$?(G_2, G_1)$	
GroupsInto	GroupsInto	×
	FinerThan	-
	Subgranularity	-
	Partition	×
	CoveredBy	✓
	Disjoint	×
	Overlap	×
FinerThan	GroupsInto	-
	FinerThan	×
	Subgranularity	×
	Partition	×
	CoveredBy	-
	Disjoint	×
	Overlap	×
Subgranularity	GroupsInto	✓
	FinerThan	×
	Subgranularity	×
	Partition	×
	CoveredBy	×
	Disjoint	×
	Overlap	×
Partition	GroupsInto	×
	FinerThan	-
	Subgranularity	×
	Partition	×
	CoveredBy	✓
	Disjoint	×
	Overlap	×

$R(G_1, G_2)$	$?(G_2, G_1)$	
CoveredBy	GroupsInto	-
	FinerThan	-
	Subgranularity	
	Partition	-
	CoveredBy	-
	Disjoint	×
	Overlap	×
Disjoint	GroupsInto	×
	FinerThan	×
	Subgranularity	×
	Partition	×
	CoveredBy	×
	Disjoint	✓
	Overlap	×
Overlap	GroupsInto	×
	FinerThan	×
	Subgranularity	×
	Partition	×
	CoveredBy	×
	Disjoint	×
	Overlap	✓

Table B.2: $\neg G_1 \approx G_2 \wedge R(G_1, G_2) \vdash ?(G_2, G_1)$

$R_1(G_1, G_2)$	$R_2(G_1, G_2)$	$?(G_1, G_2)$
GroupsInto	**FinerThan**	See Partition in Table B.1
	Subgranularity	$G_1 = G_2$
	Partition	See Partition in Table B.1
	CoveredBy	See GroupsInto+CoveredBy in Table B.1
	Disjoint	Impossible
	Overlap	Impossible
FinerThan	**GroupsInto**	See Partition in Table B.1
	Subgranularity	See Subgranularity in Table B.1
	Partition	See Partition in Table B.1
	CoveredBy	See FinerThan in Table B.1
	Disjoint	Impossible
	Overlap	Impossible
Subgranularity	**GroupsInto**	$G_1 = G_2$
	FinerThan	See Subgranularity in Table B.1
	Partition	$G_1 = G_2$
	CoveredBy	See Subgranularity in Table B.1
	Disjoint	Impossible
	Overlap	Impossible
Partition	**GroupsInto**	See Partition in Table B.1
	FinerThan	See Partition in Table B.1
	Subgranularity	$G_1 = G_2$
	CoveredBy	See Partition in Table B.1
	Disjoint	Impossible
	Overlap	Impossible
CoveredBy	**GroupsInto**	See GroupsInto+CoveredBy in Table B.1
	FinerThan	See FinerThan in Table B.1
	Subgranularity	See Subgranularity in Table B.1
	Partition	See Partition in Table B.1
	Disjoint	Impossible
	Overlap	Impossible
Disjoint	**GroupsInto**	Impossible
	FinerThan	Impossible
	Subgranularity	Impossible
	Partition	Impossible
	CoveredBy	Impossible
	Overlap	Impossible
Overlap	**GroupsInto**	Impossible
	FinerThan	Impossible
	Subgranularity	Impossible
	Partition	Impossible
	CoveredBy	Impossible
	Disjoint	Impossible

Table B.3: $\neg G_1 \approx G_2 \wedge R_1(G_1, G_2) \wedge R_2(G_1, G_2) \vdash ?(G_1, G_2)$

$R(G_1, G_2)$		$S(G_2, G_3)$						
	GroupsInto	**FinerThan**	**SubGranul.**	**Partition**	**CoveredBy**	**Disjoint**	**Overlap**
GroupsInto	GroupsInto ✓	GroupsInto -	GroupsInto -	GroupsInto ✓	GroupsInto -	GroupsInto -	GroupsInto -
	FinerThan -[a]	FinerThan -[b]	FinerThan -	FinerThan -[b]	FinerThan -	FinerThan ×	FinerThan ×
	SubGranul. ×	SubGranul. -	SubGranul. -	SubGranul. ×	SubGranul. -	SubGranul. ×	SubGranul. ×
	Partition -[a]	Partition -	Partition -	Partition -[b]	Partition -	Partition ×	Partition ×
	CoveredBy -	CoveredBy -	CoveredBy -	CoveredBy -[b]	CoveredBy -	CoveredBy ×	CoveredBy ×
	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint -	Disjoint ×
	Overlap ×	Overlap -	Overlap -	Overlap ×	Overlap -	Overlap -	Overlap -
FinerThan	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto ×	GroupsInto -
	FinerThan -	FinerThan ✓	FinerThan ✓	FinerThan ✓	FinerThan -	FinerThan ×	FinerThan -
	SubGranul. -	SubGranul. -	SubGranul. -	SubGranul. -	SubGranul. -	SubGranul. ×	SubGranul. -
	Partition -	Partition -	Partition -	Partition -	Partition -	Partition ×	Partition -
	CoveredBy -	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ×	CoveredBy -
	Disjoint -	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ✓	Disjoint -
	Overlap -	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap -
SubGranul.	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto ×	GroupsInto -
	FinerThan -	FinerThan ✓	FinerThan ✓	FinerThan ✓	FinerThan -	FinerThan ×	FinerThan -
	SubGranul. -	SubGranul. -	SubGranul. ✓	SubGranul. -	SubGranul. -	SubGranul. ×	SubGranul. -
	Partition -	Partition -	Partition -	Partition -	Partition -	Partition ×	Partition -
	CoveredBy -	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ×	CoveredBy -
	Disjoint -	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ✓	Disjoint -
	Overlap -	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap -

[a] ✓ if $CoveredBy(G_1, G_2)$ and $CoveredBy(G_2, G_3)$
[b] ✓ if $CoveredBy(G_1, G_2)$

Table B.4: $\neg G_1 \approx G_3 \wedge R(G_1, G_2) \wedge S(G_2, G_3) \vdash ?(G_1, G_3)$. First part.

$R(G_1,G_2)$	$S(G_2,G_3)$							
	GroupsInto	**FinerThan**	**SubGranul.**	**Partition**	**CoveredBy**	**Disjoint**	**Overlap**
Partition	GroupsInto ✓	GroupsInto -	GroupsInto -	GroupsInto ✓	GroupsInto -	GroupsInto ×	GroupsInto ×
	FinerThan -	FinerThan ✓	FinerThan ✓	FinerThan ✓	FinerThan -	FinerThan ×	FinerThan ×
	SubGranul. ×	SubGranul. -	SubGranul. -	SubGranul. ×	SubGranul. -	SubGranul. ×	SubGranul. ×
	Partition -	Partition -	Partition -	Partition ✓	Partition -	Partition ×	Partition ×
	CoveredBy -	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ×	CoveredBy ×
	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ✓	Disjoint ×
	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap ✓
CoveredBy	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto -	GroupsInto ×	GroupsInto -
	FinerThan -	FinerThan -	FinerThan -	FinerThan -	FinerThan -	FinerThan ×	FinerThan -
	SubGranul. -	SubGranul. -	SubGranul. -	SubGranul. -	SubGranul. -	SubGranul. ×	SubGranul. -
	Partition -	Partition -	Partition -	Partition -	Partition -	Partition ×	Partition -
	CoveredBy -	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ✓	CoveredBy ×	CoveredBy -
	Disjoint -	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ✓	Disjoint -
	Overlap -	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap ×	Overlap -
Disjoint	GroupsInto ×	GroupsInto ×	GroupsInto ×	GroupsInto ×	GroupsInto ×	GroupsInto -	GroupsInto ×
	FinerThan ×	FinerThan -	FinerThan -	FinerThan ×	FinerThan -	FinerThan -	FinerThan -
	SubGranul. ×	SubGranul. -	SubGranul. -	SubGranul. ×	SubGranul. -	SubGranul. -	SubGranul. -
	Partition ×	Partition ×	Partition ×	Partition ×	Partition ×	Partition -	Partition ×
	CoveredBy ×	CoveredBy -	CoveredBy -	CoveredBy ×	CoveredBy -	CoveredBy -	CoveredBy -
	Disjoint ✓	Disjoint -	Disjoint -	Disjoint ✓	Disjoint -	Disjoint -	Disjoint -
	Overlap ×	Overlap -	Overlap -	Overlap ×	Overlap -	Overlap -	Overlap -
Overlap	GroupsInto -	GroupsInto ×	GroupsInto ×	GroupsInto ×	GroupsInto ×	GroupsInto -	GroupsInto -
	FinerThan ×	FinerThan -	FinerThan -	FinerThan ×	FinerThan -	FinerThan ×	FinerThan -
	SubGranul. ×	SubGranul. -	SubGranul. -	SubGranul. ×	SubGranul. -	SubGranul. ×	SubGranul. -
	Partition ×	Partition ×	Partition ×	Partition ×	Partition ×	Partition ×	Partition -
	CoveredBy ×	CoveredBy -	CoveredBy -	CoveredBy ×	CoveredBy -	CoveredBy ×	CoveredBy -
	Disjoint -	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint ×	Disjoint -	Disjoint -
	Overlap -	Overlap -	Overlap -	Overlap ✓	Overlap -	Overlap -	Overlap -

Table B.5: $\neg G_1 \approx G_3 \wedge R(G_1,G_2) \wedge S(G_2,G_3) \vdash ?(G_1,G_3)$. Second part.

	GroupsInto	FinerThan	Subgranularity	Partition	CoveredBy	Disjoint	Overlap
$G' = Grouping(G, P, L)$	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ✓	$G' \to G$ ×	$G' \to G$ ×
	$G \to G'$ ✓	$G \to G'$ ✓	$G \to G'$ ×	$G \to G'$ ✓	$G \to G'$ ✓	$G \to G'$ ×	$G \to G'$ ×
$G' = Combine(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ×	$G' \to G_1$ ×
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ ✓	$G' \to G_2$ ×	$G' \to G_2$ ×
	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ ×	$G_1 \to G'$ ×
	$G_2 \to G'$ ✓	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ ×	$G_2 \to G'$ ×
$G' = Subset(G, S)$	$G' \to G$ ×	$G' \to G$ ✓	$G' \to G$ ✓	$G' \to G$ ×	$G' \to G$ ✓	$G' \to G$ ×	$G' \to G$ ×
	$G \to G'$ ✓	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ×
$G' = Subset(g, G, R)$	$G' \to G$ -	$G' \to G$ ✓	$G' \to G$ ✓	$G' \to G$ -	$G' \to G$ ✓	$G' \to G$ ×	$G' \to G$ ×
	$G \to G'$ ✓	$G \to G'$ -	$G \to G'$ -	$G \to G'$ -	$G \to G'$ -	$G \to G'$ ×	$G \to G'$ ×
$G' = SelectContain(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ×	$G' \to G_1$ ×
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ ×	$G' \to G_2$ -
	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ ×	$G_1 \to G'$ ×
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ ×	$G_2 \to G'$ -
$G' = SelectInside(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ×	$G' \to G_1$ ×
	$G' \to G_2$ -	$G' \to G_2$ ✓	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ ✓	$G' \to G_2$ ×	$G' \to G_2$ ×
	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ ×	$G_1 \to G'$ ×
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ ×	$G_2 \to G'$ ×

Table B.6: Rules to infer relationships between operands and result of an operation. First part.

	GroupsInto	FinerThan	Subgranularity	Partition	CoveredBy	Disjoint	Overlap
$G' = SelectIntersect(G_1, G_2)$	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ ×	$G' \rightarrow G_1$ ×
	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ✓
	$G_1 \rightarrow G'$ ✓	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ ×	$G_1 \rightarrow G'$ ×
	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ✓
$G' = Union(G_1, G_2)$	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ ×	$G' \rightarrow G_1$ ×
	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ×
	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ ✓	$G_1 \rightarrow G'$ ✓	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ ✓	$G_1 \rightarrow G'$ ×	$G_1 \rightarrow G'$ ×
	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ ✓	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ×
$G' = Intersect(G_1, G_2)$	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ ×	$G' \rightarrow G_1$ ×
	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ ✓	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ -	$G' \rightarrow G_2$ ✓	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ×
	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ ×	$G_1 \rightarrow G'$ ×
	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ -	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ×
$G' = Difference(G_1, G_2)$	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ -	$G' \rightarrow G_1$ ✓	$G' \rightarrow G_1$ ×	$G' \rightarrow G_1$ ×
	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ×	$G' \rightarrow G_2$ ✓	$G' \rightarrow G_2$ ×
	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ -	$G_1 \rightarrow G'$ ×	$G_1 \rightarrow G'$ ×
	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ×	$G_2 \rightarrow G'$ ✓	$G_2 \rightarrow G'$ ×

Table B.7: Rules to infer relationships between operands and result of an operation. Second part.

B.2 Inference rules for temporal relationships

$R(G_1, G_2)$	$?(G_1, G_2)$	
GroupsInto	GroupsInto	✓
	FinerThan	-
	Subgranularity	×
	Partition	-
	GroupsPeriodInto	-
FinerThan	GroupsInto	-
	FinerThan	✓
	Subgranularity	-
	Partition	-
	GroupsPeriodInto	-
Subgranularity	GroupsInto	×
	FinerThan	✓
	Subgranularity	✓
	Partition	×
	GroupsPeriodInto	×
Partition	GroupsInto	✓
	FinerThan	✓
	Subgranularity	×
	Partition	✓
	GroupsPeriodInto	-
GroupsPeriodInto	GroupsInto	✓
	FinerThan	-
	Subgranularity	×
	Partition	-
	GroupsPeriodInto	✓

Table B.8: $\neg G_1 \approx G_2 \wedge R(G_1, G_2) \vdash ?(G_1, G_2)$

$R(G_1, G_2)$	$?(G_2, G_1)$	
GroupsInto	GroupsInto	×
	FinerThan	-
	Subgranularity	-
	Partition	×
	GroupsPeriodInto	×
FinerThan	GroupsInto	-
	FinerThan	×
	Subgranularity	×
	Partition	×
	GroupsPeriodInto	-
Subgranularity	GroupsInto	✓
	FinerThan	×
	Subgranularity	×
	Partition	×
	GroupsPeriodInto	-
Partition	GroupsInto	×
	FinerThan	×
	Subgranularity	×
	Partition	×
	GroupsPeriodInto	×
GroupsPeriodInto	GroupsInto	×
	FinerThan	-
	Subgranularity	-
	Partition	×
	GroupsPeriodInto	×

Table B.9: $\neg G_1 \approx G_2 \wedge R(G_1, G_2) \vdash ?(G_2, G_1)$

$R_1(G_1, G_2)$	$R_2(G_1, G_2)$	$?(G_1, G_2)$
GroupsInto	**FinerThan**	See Partition in Table B.8
	Subgranularity	$G_1 = G_2$
	Partition	See Partition in Table B.8
	GroupsPeriodInto	See GroupsPeriodInto in Table B.8
FinerThan	**GroupsInto**	See Partition in Table B.8
	Subgranularity	See SubGranularity in Table B.8
	Partition	See Partition in Table B.8
	GroupsPeriodInto	See GroupsPeriodInto+Partition in Table B.8
Subgranularity	**GroupsInto**	$G_1 = G_2$
	FinerThan	See SubGranularity in Table B.8
	Partition	$G_1 = G_2$
	GroupsPeriodInto	$G_1 = G_2$
Partition	**GroupsInto**	See Partition in Table B.8
	FinerThan	See Partition in Table B.8
	Subgranularity	$G_1 = G_2$
	GroupsPeriodInto	See GroupsPeriodInto+Partition in Table B.8
GroupsPeriodInto	**GroupsInto**	See GroupsPeriodInto in Table B.8
	FinerThan	See GroupsPeriodInto+Partition in Table B.8
	Subgranularity	$G_1 = G_2$
	Partition	See GroupsPeriodInto+Partition in Table B.8

Table B.10: $\neg G_1 \approx G_2 \wedge R_1(G_1, G_2) \wedge R_2(G_1, G_2) \vdash ?(G_1, G_2)$

$R(G_1,G_2) \setminus S(G_2,G_3)$	**GroupsInto**		**FinerThan**		**Subgranularity**		**Partition**		**GroupsPeriodInto**	
GroupsInto	GroupsInto	✓	GroupsInto	-	GroupsInto	-	GroupsInto	✓	GroupsInto	✓
	FinerThan	-	FinerThan	-	FinerThan	-	FinerThan	-	FinerThan	-
	Subgranularity	×	Subgranularity	-	Subgranularity	-	Subgranularity	×	Subgranularity	×
	Partition	-	Partition	-	Partition	-	Partition	-	Partition	-
	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-
FinerThan	GroupsInto	-	GroupsInto	-	GroupsInto	-	GroupsInto	-	GroupsInto	-
	FinerThan	-	FinerThan	✓	FinerThan	✓	FinerThan	✓	FinerThan	-
	Subgranularity	-	Subgranularity	-	Subgranularity	-	Subgranularity	-	Subgranularity	-
	Partition	-	Partition	-	Partition	-	Partition	-	Partition	-
	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-
Subgranularity	GroupsInto	-	GroupsInto	-	GroupsInto	-	GroupsInto	-	GroupsInto	-
	FinerThan	-	FinerThan	✓	FinerThan	✓	FinerThan	✓	FinerThan	-
	Subgranularity	-	Subgranularity	-	Subgranularity	✓	Subgranularity	-	Subgranularity	-
	Partition	-	Partition	-	Partition	-	Partition	-	Partition	-
	GroupsPeriodInto	-	GroupsPeriodInto	×	GroupsPeriodInto	×	GroupsPeriodInto	×	GroupsPeriodInto	-
Partition	GroupsInto	✓	GroupsInto	-	GroupsInto	-	GroupsInto	✓	GroupsInto	✓
	FinerThan	-	FinerThan	✓	FinerThan	✓	FinerThan	✓	FinerThan	-
	Subgranularity	×	Subgranularity	-	Subgranularity	-	Subgranularity	×	Subgranularity	×
	Partition	-	Partition	-	Partition	-	Partition	✓	Partition	-
	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-
GroupsPeriodInto	GroupsInto	✓	GroupsInto	-	GroupsInto	-	GroupsInto	✓	GroupsInto	✓
	FinerThan	-	FinerThan	-	FinerThan	-	FinerThan	-	FinerThan	-
	Subgranularity	×	Subgranularity	-	Subgranularity	-	Subgranularity	×	Subgranularity	×
	Partition	-	Partition	-	Partition	-	Partition	-	Partition	-
	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	-	GroupsPeriodInto	✓

Table B.11: $\neg G_1 \approx G_3 \wedge R(G_1,G_2) \wedge S(G_2,G_3) \vdash ?(G_1,G_3)$

	GroupsInto	FinerThan	Subgranularity	ShiftEquiv	Partition	GroupsPeriodInto
$G' = Group_m(G)$	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ×
	$G \to G'$ ✓	$G \to G'$ ✓	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ✓	$G \to G'$ ✓ (1, m)
$G' = AlteringTick^m_{l,k}(G_1, G_2)$	$G' \to G_1$ ×	$G' \to G_1$ ×	$G' \to G_1$ ×	$G' \to G_1$ ×	$G' \to G_1$ ×	$G' \to G_1$ ×
	$G' \to G_2$ ×	$G' \to G_2$ ×	$G' \to G_2$ ×	$G' \to G_2$ ×	$G' \to G_2$ ×	$G' \to G_2$ ×
	$G_1 \to G'$ ×	$G_1 \to G'$ ×	$G_1 \to G'$ ×	$G_1 \to G'$ ×	$G_1 \to G'$ ×	$G_1 \to G'$ ×
	$G_2 \to G'$ ✓	$G_2 \to G'$ ✓	$G_2 \to G'$ ×	$G_2 \to G'$ ×	$G_2 \to G'$ ✓	$G_2 \to G'$ ✓
$G' = Shift_m(G)$	$G' \to G$ ✓	$G' \to G$ ✓	$G' \to G$ ✓	$G' \to G$ ✓	$G' \to G$ ✓	$G' \to G$ ✓ (1,1)
	$G \to G'$ ✓	$G \to G'$ ✓	$G \to G'$ ✓	$G \to G'$ ✓	$G \to G'$ ✓	$G \to G'$ ✓ (1,1)
$G' = Combine(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -
	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -
	$G_2 \to G'$ ✓	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -
$G' = AnchoredGrouping(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -
	$G_1 \to G'$ ✓	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ ✓	$G_1 \to G'$ -
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -
$G' = Subset^n_m(G)$	$G' \to G$ ×	$G' \to G$ ✓	$G' \to G$ ✓	$G' \to G$ ×	$G' \to G$ ×	$G' \to G$ ×
	$G \to G'$ ✓	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ×	$G \to G'$ ✓
$G' = SelectDown^l_k(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -
	$G' \to G_2$ -	$G' \to G_2$ ✓	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -
	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -

Table B.12: Rules to infer relationships between operands and result of an operation. First part.

	GroupsInto	FinerThan	Subgranularity	ShiftEquiv	Partition	GroupsPeriodInto
$G' = SelectUp(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -
	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -
$G' = SelectByIntersect_k^l(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -
	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -
$G' = Union(G_1, G_2)$	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -[a]
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -
	$G_1 \to G'$ -	$G_1 \to G'$ ✓	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -
$G' = Intersection(G_1, G_2)$	$G' \to G_1$ ×	$G' \to G_1$ ✓	$G' \to G_1$ ✓	$G' \to G_1$ ×	$G' \to G_1$ ×	$G' \to G_1$ ×
	$G' \to G_2$ ×	$G' \to G_2$ ✓	$G' \to G_2$ ✓	$G' \to G_2$ ×	$G' \to G_2$ ×	$G' \to G_2$ ×
	$G_1 \to G'$ ✓	$G_1 \to G'$ ×	$G_1 \to G'$ ×	$G_1 \to G'$ ×	$G_1 \to G'$ ×	$G_1 \to G'$ -[a]
	$G_2 \to G'$ ✓	$G_2 \to G'$ ×	$G_2 \to G'$ ×	$G_2 \to G'$ ×	$G_2 \to G'$ ×	$G_2 \to G'$ -[a]
$G' = Difference(G_1, G_2)$	$G' \to G_1$ -	$G' \to G_1$ ✓	$G' \to G_1$ ✓	$G' \to G_1$ -	$G' \to G_1$ -	$G' \to G_1$ -
	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -	$G' \to G_2$ -
	$G_1 \to G'$ ✓	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -	$G_1 \to G'$ -[a]
	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -	$G_2 \to G'$ -

[a] ✓ if G_1, G_2 are periodic

Table B.13: Rules to infer relationships between operands and result of an operation. Second part.

B.3 Inference rules for spatio-temporal relationships

Q	$S(tG, tG')$											
	$\approx$		**GroupsInto**		**FinerThan**		**SubGranul.**		**Partition**		**GroupsPeriodInto**	
$\forall\forall$	$\forall\forall$	✓	$\forall\forall$	✓	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	✓	$\forall\forall$	✓
	$\forall\exists$	✓	$\forall\exists$	✓	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	✓	$\forall\exists$	✓
	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	-	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	✓
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓
$\forall\exists$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	✓	$\forall\exists$	✓	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	✓	$\forall\exists$	✓
	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓
$\exists\forall$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-
	$\exists\forall$	✓	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	✓	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	-	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	-
$\exists\exists$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-
	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	-	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	-

Table B.14: $QR(stG, stH) \wedge S(tG, tG') \vdash ?R(stG', stH)$

Q	$S(tG', tG)$											
	$\approx$		**GroupsInto**		**FinerThan**		**SubGranul.**		**Partition**		**GroupsPeriodInto**	
$\forall\forall$	$\forall\forall$	✓	$\forall\forall$	-	$\forall\forall$	✓	$\forall\forall$	✓	$\forall\forall$	✓	$\forall\forall$	-
	$\forall\exists$	✓	$\forall\exists$	-	$\forall\exists$	✓	$\forall\exists$	✓	$\forall\exists$	✓	$\forall\exists$	-
	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	✓
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓
$\forall\exists$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	✓	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	✓	$\forall\exists$	-	$\forall\exists$	-
	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	-	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓
$\exists\forall$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-
	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	✓	$\exists\forall$	✓
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	-	$\exists\exists$	-	$\exists\exists$	✓	$\exists\exists$	✓
$\exists\exists$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-
	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	-	$\exists\exists$	-	$\exists\exists$	✓	$\exists\exists$	✓

Table B.15: $QR(stG, stH) \wedge S(tG', tG) \vdash ?R(stG', stH)$

Q	$S(tG', tG)$											
	$\approx$		**GroupsInto**		**FinerThan**		**SubGranul.**		**Partition**		**GroupsPeriodInto**	
$\forall\forall$	$\forall\forall$	✓	$\forall\forall$	✓	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	✓	$\forall\forall$	✓
	$\forall\exists$	✓	$\forall\exists$	✓	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	✓	$\forall\exists$	✓
	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	-	$\exists\forall$	✓	$\exists\forall$	✓	$\exists\forall$	✓
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓
$\forall\exists$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	✓	$\forall\exists$	✓	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	✓	$\forall\exists$	✓
	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓
$\exists\forall$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-
	$\exists\forall$	✓	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	✓	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓
$\exists\exists$	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-	$\forall\forall$	-
	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-	$\forall\exists$	-
	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-	$\exists\forall$	-
	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓	$\exists\exists$	✓

Table B.16: $QR(stG, stH) \wedge S(tG, tH) \vdash ?R'(stH, stG)$. R' is such that $R(sG, sH) \vdash R'(sH, sG)$

Q	Q'	?
$\forall\forall$	$\forall\forall$	$\forall\forall R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\forall\exists$	$\forall\exists R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\exists\forall$	$\exists\forall R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\exists\exists$	$\exists\exists R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
$\forall\exists$	$\forall\forall$	$\forall\exists R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\forall\exists$	-
	$\exists\forall$	$\exists\exists R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\exists\exists$	-
$\exists\forall$	$\forall\forall$	$\exists\forall R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\forall\exists$	$\exists\exists R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\exists\forall$	-
	$\exists\exists$	-
$\exists\exists$	$\forall\forall$	$\exists\exists R''$, where $R(G,H) \wedge R'(G,H) \vdash R''(G,H)$
	$\forall\exists$	-
	$\exists\forall$	-
	$\exists\exists$	-

Table B.17: $QR(stG, stH) \wedge Q'R'(stG, stH) \vdash ?R''(stG, stH)$. R'' is such that $R(sG, sH) \wedge R'(sG, sH) \vdash R''(sG, sH)$

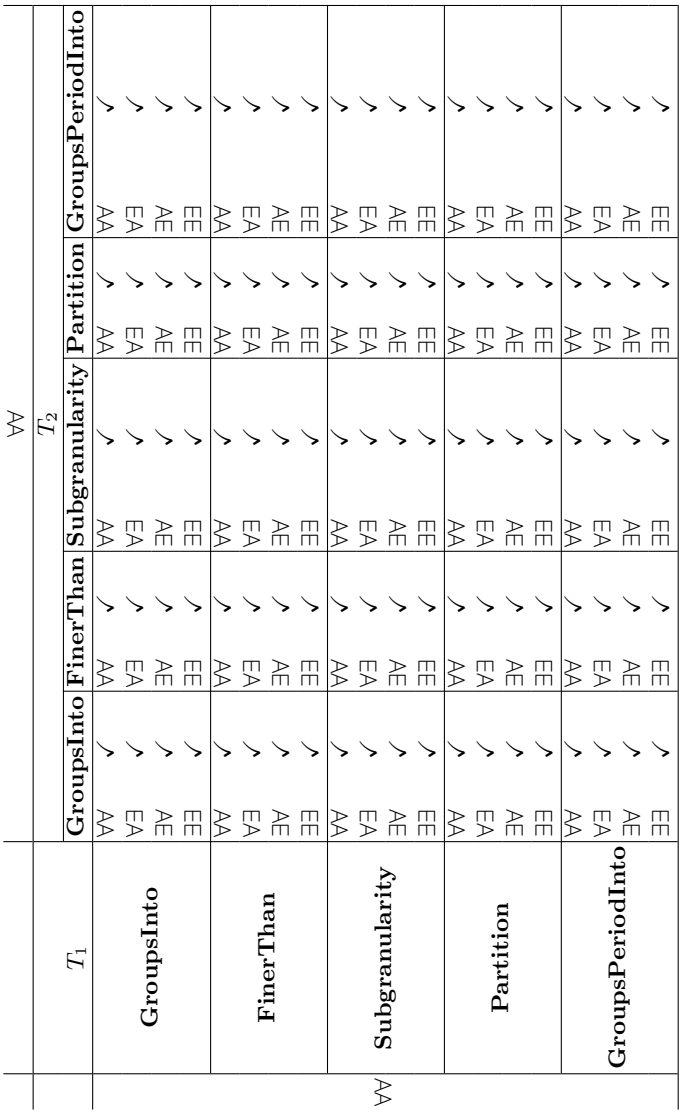


Table B.18: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

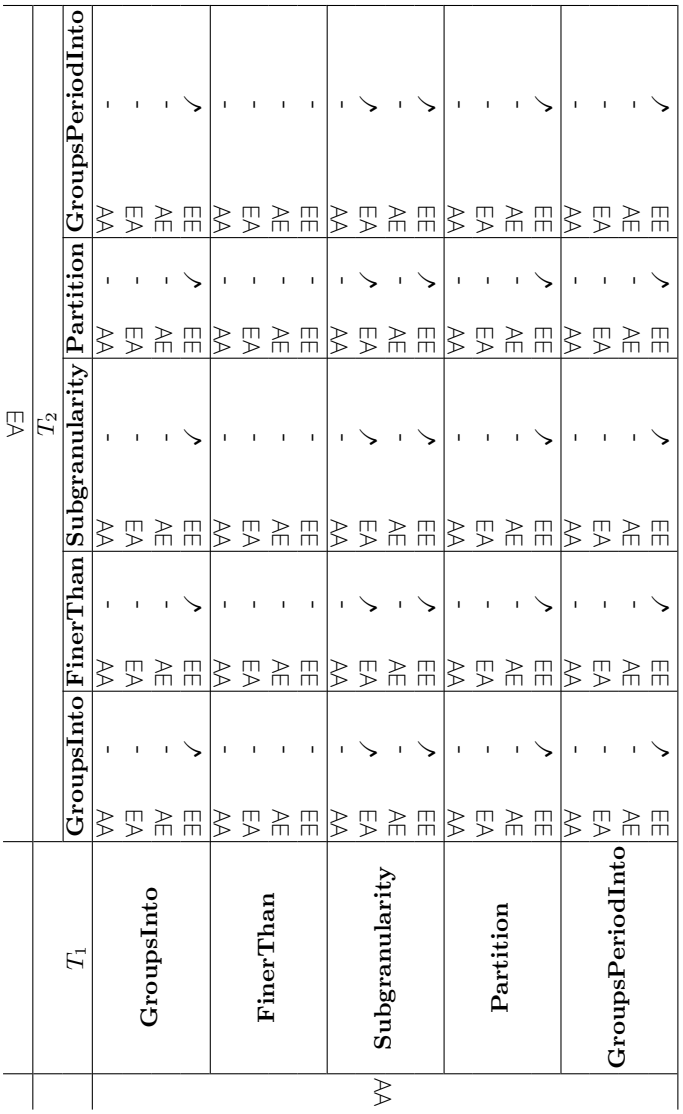


Table B.19: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

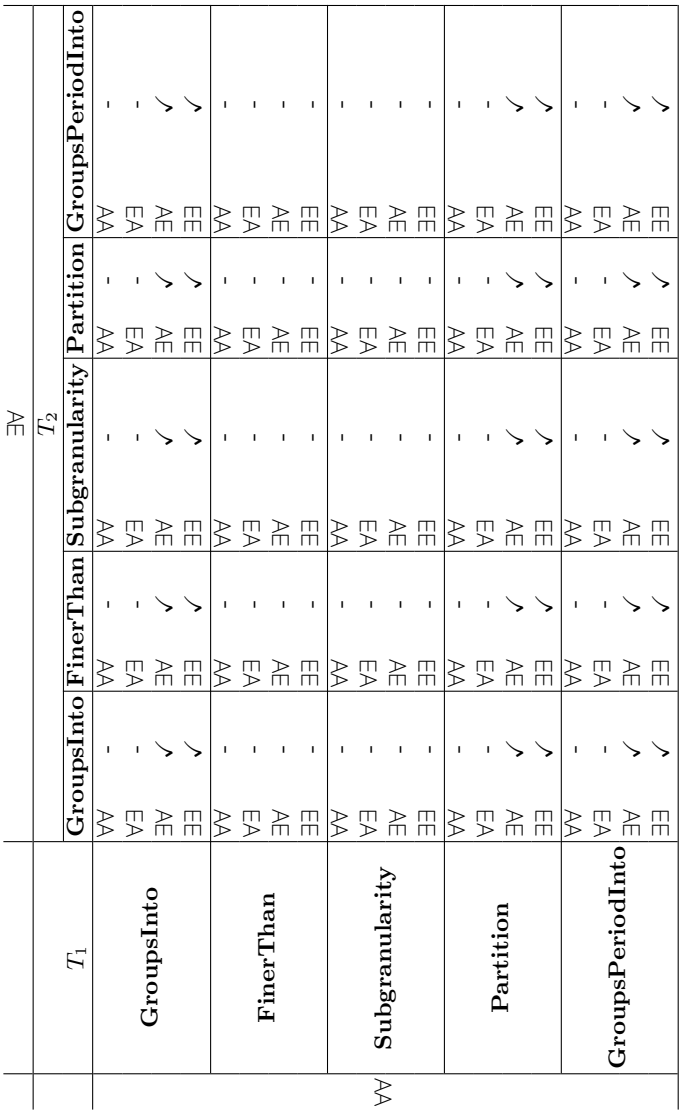


Table B.20: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

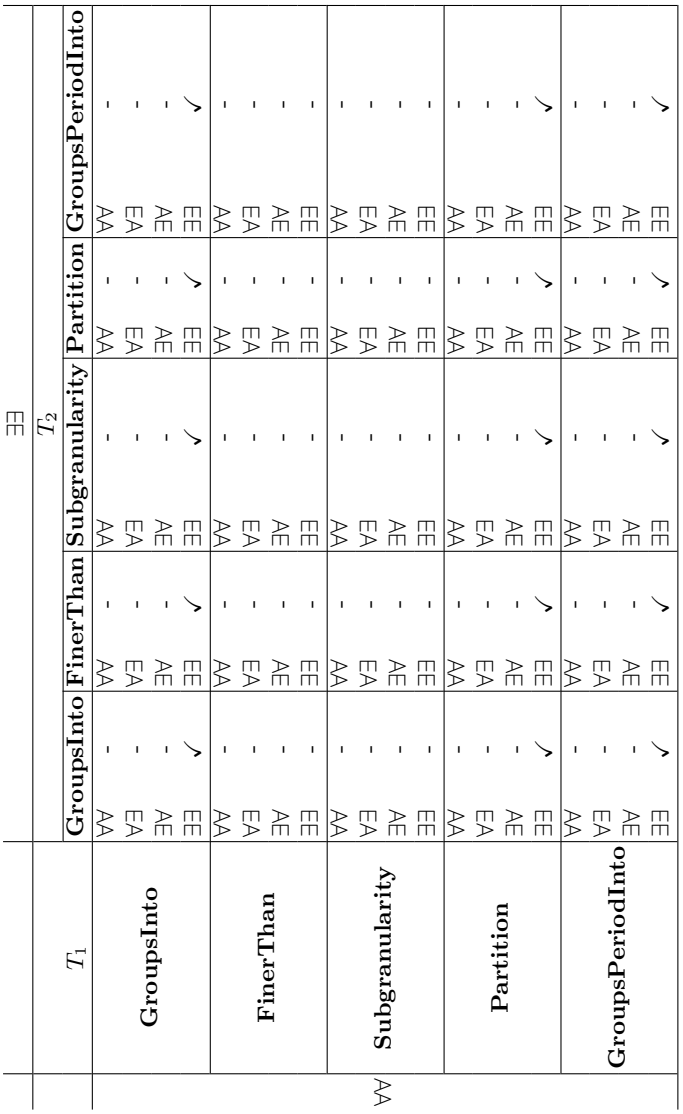


Table B.21: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

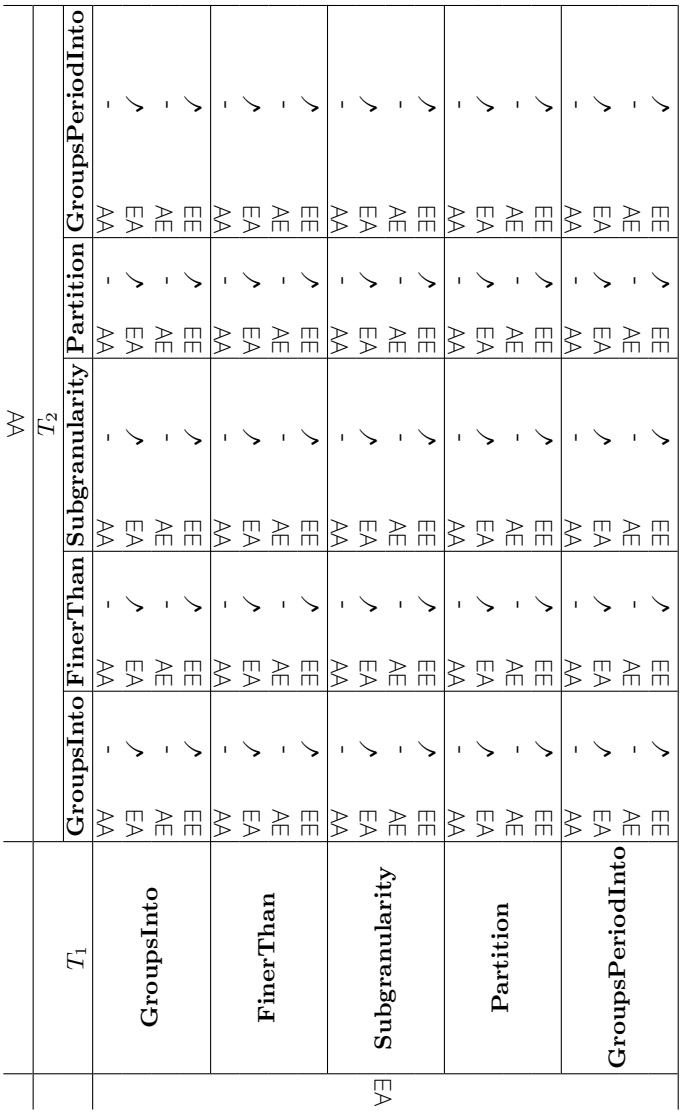


Table B.22: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

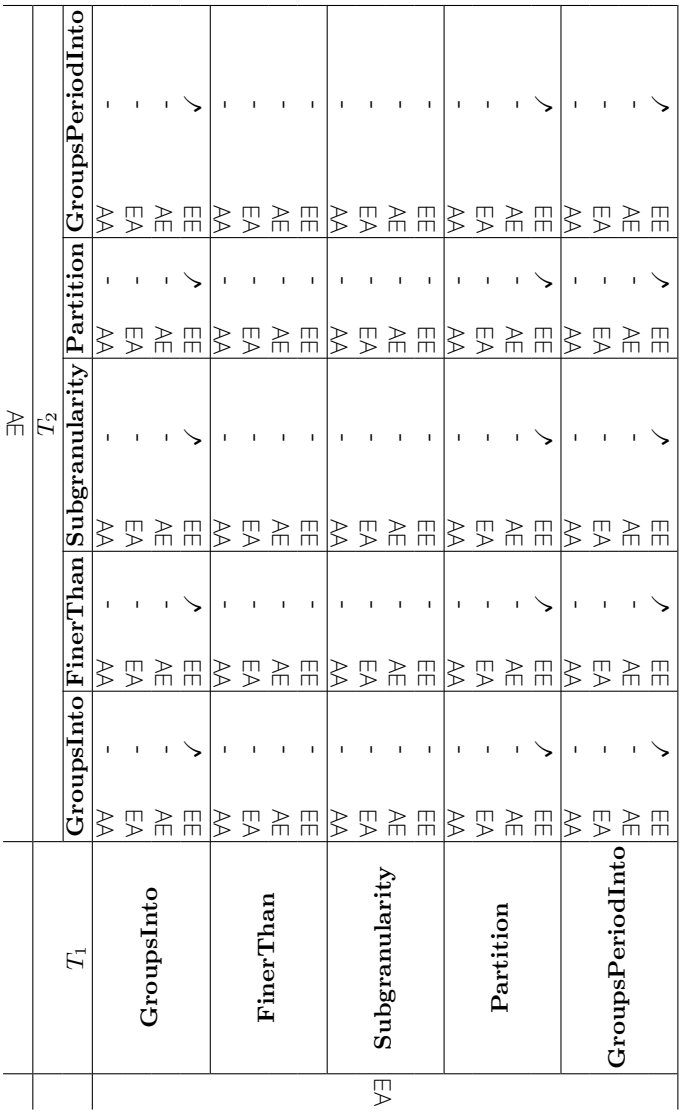


Table B.23: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

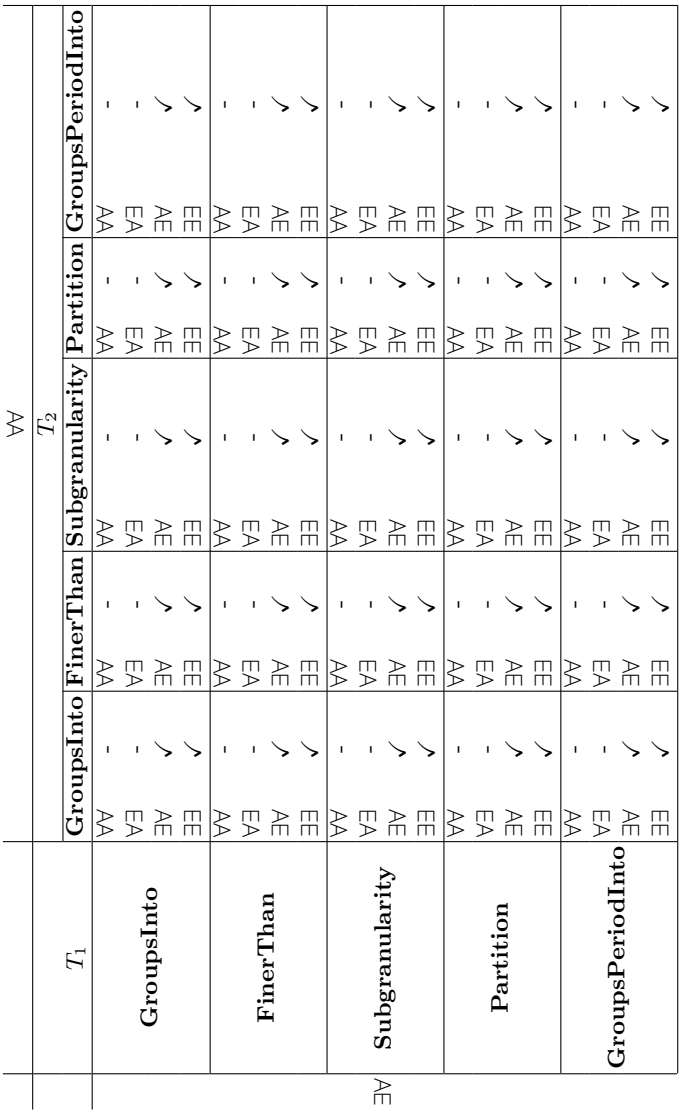


Table B.24: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

Q_1	T_1	Q_3	Q_2 = ∃∀, T_2: GroupsInto	FinerThan	Subgranularity	Partition	GroupsPeriodInto
∀∃	GroupsInto	∀∀	-	-	-	-	-
		∃∀	-	-	-	-	-
		∀∃	-	-	-	-	-
		∃∃	-	-	-	-	-
	FinerThan	∀∀	-	-	-	-	-
		∃∀	-	-	-	-	-
		∀∃	-	-	-	-	-
		∃∃	-	-	-	-	-
	Subgranularity	∀∀	-	-	-	-	-
		∃∀	-	-	-	-	-
		∀∃	-	-	-	-	-
		∃∃	✓	✓	✓	✓	✓
	Partition	∀∀	-	-	-	-	-
		∃∀	-	-	-	-	-
		∀∃	-	-	-	-	-
		∃∃	-	-	-	-	-
	GroupsPeriodInto	∀∀	-	-	-	-	-
		∃∀	-	-	-	-	-
		∀∃	-	-	-	-	-
		∃∃	-	-	-	-	-

Table B.25: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

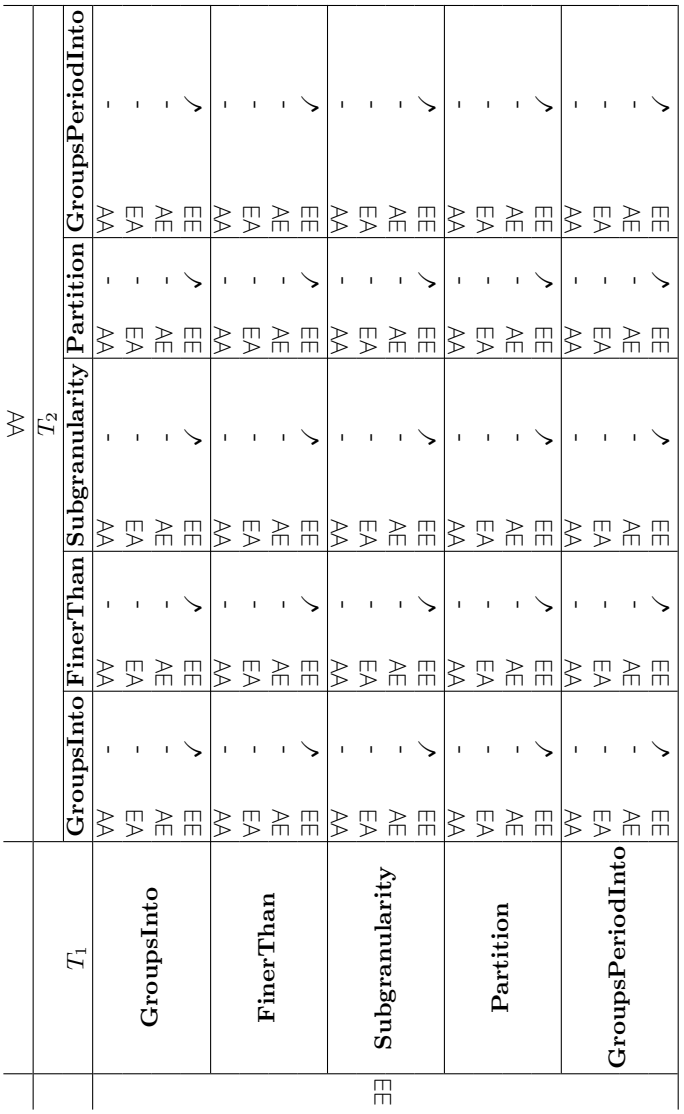


Table B.26: $Q_1R_1(stG, stH) \wedge Q_2R_2(stH, stI) \wedge T_1(tG, tH) \wedge T_2(tH, tI) \Rightarrow Q_3R_3(stG, stI)$. R_3 is such that $R_1(stG.E(t), stH.E(t)) \wedge R_2(stH.E(t), stI.E(t)) \vdash R_3(stG.E(t), stI.E(t))$, applying the concatenation rules for relationships between spatial granularities.

Q	Q'			
	$\forall\forall$	$\forall\exists$	$\exists\forall$	$\exists\exists$
$\forall\forall$	✓	✓	✓	✓
$\forall\exists$	✓	-	✓	-
$\exists\forall$	✓	✓	-	-
$\exists\exists$	✓	-	-	-

Table B.27: $QR(stG, stH) \wedge Q'\neg R(stG, stH) \vdash \bot$

Q	**Q'**
$\forall\forall$	$\exists\exists$
$\forall\exists$	$\exists\forall$
$\exists\forall$	$\forall\exists$
$\exists\exists$	$\forall\forall$

Table B.28: $QR(stG, stH) \vdash \ldots \vdash \bot \vdash Q'\neg R(stG, stH)$

C

The semantics of ST4SQL constructs

In this chapter we present the semantics of ST4SQL. As we have already mentioned, any ST4SQL query can be translated into an equivalent SQL query. Thus, the semantics of the ST4SQL language is given with respect to the one of SQL. We show how ST4SQL clauses and constructs are translated into equivalent SQL statements. In particular, we will focus only on those constructs that are new in ST4SQL.

In the following, each section focuses on just one ST4SQL construct. The semantics is presented considering a ST4SQL query containing the construct we are interested in and providing the equivalent SQL query. When more spatio-temporal constructs appear in a ST4SQL query the equivalent SQL query can be obtained combining the traslations of the different ST4SQL constructs.

C.1 The `SEMANTICS` clause

The clause `SEMANTICS` allows one to specify how the system has to manage temporal, spatial, and/or spatio-temporal dimensions for the query evaluation. How this clause is translated in SQL depends from the specified semantics and by the dimension on which it has to be applied. However, in general, the clause corresponds to some additional join and selection conditions.

Hereafter, let T_1, T_2, ..., T_n be the tables specified by the user in the `FROM` clause.

For the sake of simplicity, hereinafter we will show the ST4SQL semantics by assuming that the involved dimensions are valid time (`VALID TIME`) and valid space (`VALID SPACE`), thus we will use the `VALIDT(R)` function in order to obtain the valid time of a tuple. In other cases, the function we described in Section 7.3.2 and corresponding to the specified temporal dimension has to be used in place of the `VALIDT` function. Moreover, in the case of an interval expression `<interval>` (e.g., `VALIDT(R)`), we represent with `<interval>$start` and `<interval>$end` the start and the end time point of the interval.

C.1.1 Temporal semantics

The ATEMPORAL semantics

ATEMPORAL semantics without TIMESLICE *and* WITH TGRANULARITY *tokens*

ST4SQL :

```
SEMANTICS ATEMPORAL ON <tdim>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL:

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

ATEMPORAL semantics with the TIMESLICE *token but without the WITH TGRANULARITY token*

ST4SQL :

```
SEMANTICS ATEMPORAL ON <tdim> TIMESLICE (s,e)
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL :

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions> AND VALIDT(T1) BETWEEN s AND e AND
      ... VALIDT(Tn) BETWEEN s AND e
```

ATEMPORAL semantics with TIMESLICE *and* WITH TGRANULARITY *tokens*

ST4SQL :

```
SEMANTICS ATEMPORAL ON <tdim> WITH TGRANULARITY <tG>
          TIMESLICE <ts_exp>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL :

```
WITH granules AS (
     SELECT g.start, g.end
     FROM t-granularity AS tg, granule AS g
     WHERE g.tgranularity = tg.id AND
            tg.name = <tG> AND
            g.index BETWEEN <ts_exp>$start
                    AND <ts_exp>$end )
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions> AND
      VALIDT(T1) && ANY granules
      AND ... AND
      VALIDT(Tn) && ANY granules
```

`&&` is an operator we defined for the sake of simplicity. It tests whether two interval provided as operators intersect each other or not. `&& ANY` allows to check whether the first operand intersects any interval contained in the subquery provided as second operand.

The `CURRENT` semantics

`CURRENT` semantics without the `WITH TGRANULARITY` *token*

ST4SQL :

```
SEMANTICS CURRENT ON <tdim>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL :

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions> AND
      now() BETWEEN VALIDT(T1)$start AND VALIDT(T1)$end
      AND ... AND
      now() BETWEEN VALIDT(Tn)$start AND VALIDT(Tn)$end
```

`CURRENT` semantics with the `WITH TGRANULARITY` *token*

ST4SQL :

```
SEMANTICS CURRENT ON <tdim> WITH TGRANULARITY <tG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL :

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn,
     t-granularity AS tg, granule AS g
WHERE <conditions> AND g.tgranularity = tg.id AND
      tg.name = <tG> AND now() BETWEEN g.start AND g.end AND
      VALIDT(T1) && (g.start,g.end) AND ... AND
      VALIDT(Tn) && (g.start,g.end)
```

The SEQUENCED semantics

SEQUENCED semantics without the WITH TGRANULARITY *token*

ST4SQL :

```
SEMANTICS SEQUENCED ON <tdim>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL :

```
SELECT <sel_element_list>,
       intersection(VALIDT(T1),
            intersection(VALIDT(T2),
                 ..., VALIDT(Tn))...))
FROM T1, T2, ..., Tn
WHERE <conditions> AND
       intersection(VALIDT(T1),
            intersection(VALIDT(T2),
                 ..., VALIDT(Tn))...)) IS NOT NULL
```

Given two intervals, we defined the `intersection` function returning their intersection or `NULL` if they have an empty intersection. When the `SEQUENCED` semantics is applied to a point-based temporal dimension (e.g., the initiating and terminating event times) the `intersection` function is equivalent to the equality operator.

SEQUENCED semantics with the WITH TGRANULARITY *token*

ST4SQL :

```
SEMANTICS SEQUENCED ON <tdim> WITH TGRANULARITY <tG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL :

```
SELECT <sel_element_list>, g.start, g.end
FROM T1, T2, ..., Tn, t-granularity AS tg, granule AS g
WHERE <conditions> AND g.tgranularity = tg.id AND
     tg.name = <tG> AND VALIDT(T1) && (g.start,g.end) AND
     ... AND VALIDT(Tn) && (g.start,g.end)
```

SEQUENCED semantics with TIMESLICE *and* WITH TGRANULARITY *tokens*

ST4SQL :

```
SEMANTICS SEQUENCED ON <tdim> TIMESLICE <ts_exp>
          WITH TGRANULARITY <tG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL :

```
SELECT <sel_element_list>, g.start, g.end
FROM T1, T2, ..., Tn, t-granularity AS tg, granule AS g
WHERE <conditions> AND g.tgranularity = tg.id AND
     tg.name = <tG> AND VALIDT(T1) && (g.start,g.end) AND
     ... AND VALIDT(Tn) && (g.start,g.end) AND
     g.index BETWEEN <ts_exp>$start AND <ts_exp>$end
```

The NEXT semantics

NEXT semantics without TIMESLICE *and* WITH TGRANULARITY *tokens*

ST4SQL :

```
SEMANTICS NEXT ON <tdim> THROUGH <attr>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <join_conditions> AND <select_conditions>
```

SQL :

```
WITH from_tables AS
     (SELECT * FROM T1, T2, ..., Tn WHERE <join_conditions>)
SELECT <sel_element_list>
FROM from_tables AS T INNER JOIN from_tables AS Tnext
         ON T.<attr> = Tnext.<attr>
WHERE <select_conditions> AND
   VALIDT(T)$end < VALIDT(Tnext)$start AND
   NOT EXISTS (SELECT * FROM from_tables AS subt
   WHERE T.<attr> = subt.<attr> AND
         VALIDT(T)$end < VALIDT(subt)$start AND
         VALIDT(subt)$end < VALIDT(Tnext)$start )
```

NEXT semantics with the `duration` *option but without* `TIMESLICE` *and* `WITH TGRANULARITY` *tokens*

ST4SQL :

```
SEMANTICS NEXT(<duration>) ON <tdim> THROUGH <attr>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <join_conditions> AND <select_conditions>
```

SQL :

```
WITH from_tables AS
     (SELECT * FROM T1, T2, ..., Tn WHERE <join_conditions>)
SELECT <sel_element_list>
FROM from_tables AS T INNER JOIN from_tables AS Tnext
         ON T.<attr> = Tnext.<attr>
WHERE <select_conditions> AND
   VALIDT(T)$end < VALIDT(Tnext)$start AND
   VALIDT(Tnext)$start <= VALIDT(T)$end + <duration> + 1 AND
   NOT EXISTS (SELECT * FROM from_tables AS subt
   WHERE T.<attr> = subt.<attr> AND
         VALIDT(T)$end < VALIDT(subt)$start AND
         VALIDT(subt)$end < VALIDT(Tnext)$start )
```

NEXT semantics with the `TIMESLICE` *token but without the* `WITH TGRANULARITY` *token*

ST4SQL :

```
SEMANTICS NEXT ON <tdim> THROUGH <attr>
          TIMESLICE <ts_exp>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <join_conditions> AND <select_conditions>
```

SQL :

```
WITH from_tables AS
     (SELECT * FROM T1, T2, ..., Tn WHERE <join_conditions>)
SELECT <sel_element_list>
FROM from_tables AS T INNER JOIN from_tables AS Tnext
         ON T.<attr> = Tnext.<attr>
WHERE <select_conditions> AND
   <ts_exp>$start <= VALIDT(t)$start AND
   VALIDT(tnext)$end <= <ts_exp>$end
   VALIDT(T)$end < VALIDT(Tnext)$start AND
   NOT EXISTS (SELECT * FROM from_tables AS subt
   WHERE T.<attr> = subt.<attr> AND
         VALIDT(T)$end < VALIDT(subt)$start AND
         VALIDT(subt)$end < VALIDT(Tnext)$start )
```

NEXT semantics without the TIMESLICE *token but with the* WITH TGRANULARITY *token*

ST4SQL :

```
SEMANTICS NEXT ON <tdim> THROUGH <attr>
          WITH TGRANULARITY <tG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <join_conditions> AND <select_conditions>
```

SQL :

```
WITH from_tables AS
     (SELECT * FROM T1, T2, ..., Tn WHERE <join_conditions>)
SELECT <sel_element_list>
FROM <from_table> AS t INNER JOIN <from_table> AS tnext
         ON (t.<attr> = tnext.<attr>),
     t-granularity AS tg INNER JOIN granule AS g1
         ON g1.tgranularity = tg.id
     INNER JOIN granule AS g2
         ON g2.tgranularity = tg.id
WHERE tg.name = <tG> AND VALIDT(t) && (g1.start,g1.end) AND
      VALIDT(tnext) && (g2.start,g2.end) AND
      g2.index = g1.index + 1 AND NOT EXISTS (
         SELECT * FROM <from_table> AS subt
         WHERE VALIDT(subt) && (g2.start,g2.end) AND
               t.<attr> = subt.<attr> AND
               VALIDT(t)$end < VALIDT(subt)$start AND
               VALIDT(subt)$end < VALIDT(tnext)$start )
```

NEXT semantics with TIMESLICE *and* WITH TGRANULARITY *tokens*

ST4SQL :

```
SEMANTICS NEXT ON <tdim> THROUGH <attr>
          TIMESLICE <ts_exp> WITH TGRANULARITY <tG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <join_conditions> AND <select_conditions>
```

SQL :

```
WITH from_tables AS
     (SELECT * FROM T1, T2, ..., Tn WHERE <join_conditions>)
SELECT <sel_element_list>
FROM <from_table> AS t INNER JOIN <from_table> AS tnext
          ON (t.<attr> = tnext.<attr>),
     t-granularity AS tg INNER JOIN granule AS g1
          ON g1.tgranularity = tg.id
     INNER JOIN granule AS g2
          ON g2.tgranularity = tg.id
WHERE tg.name = <tG> AND VALIDT(t) && (g1.start,g1.end) AND
      VALIDT(tnext) && (g2.start,g2.end) AND
      g1.index >= <ts_exp>$start AND
      g2.index <= <ts_exp>$end AND
      g2.index = g1.index + 1 AND NOT EXISTS (
          SELECT * FROM <from_table> AS subt
          WHERE VALIDT(subt) && (g2.start,g2.end) AND
                t.<attr> = subt.<attr> AND
                VALIDT(t)$end < VALIDT(subt)$start AND
                VALIDT(subt)$end < VALIDT(tnext)$start )
```

where `<ts_exp>` is a pair `(gstart,gend)` representing an interval of granules. Thus, the first element (accessed with `<ts_exp>$start`) represents the first granule included in the timeslice interval while the second element (accessed with `<ts_exp>$end`) represents the last granule included in the timeslice interval.

In all cases, the user can refer, in the original ST4SQL query, to the value of an attribute `<attr>` in the successor tuple in a pair by using the `NEXT(<attr>)` function. This is translated in the SQL query as `tnext.<attr>`, while all other attributes are translated as `t.<attr>`.

C.1.2 Spatial semantics

For the geometrical functions we consider the functions implemented in PostGIS [172] and that implement the SQL/MM specification.

The SPACELESS semantics

SPACELESS semantics without SPACESLICE *and* WITH SGRANULARITY *tokens*

ST4SQL :

```
SEMANTICS SPACELESS ON <sdim>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL:

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SPACELESS semantics with both SPACESLICE *and* WITH SGRANULARITY *tokens*

ST4SQL :

```
SEMANTICS SPACELESS ON <sdim> SPACESLICE <ss_exp>
          WITH SGRANULARITY <sG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL:

```
WITH slice AS
     (SELECT ST_Union(n.geom) AS geom
     FROM sgranularity AS sg
          JOIN node AS n ON (n.granularity = sg.id)
          JOIN nodelabel AS nl ON (n.label = nl.id)
     WHERE sg.name = <sG> AND nl.label = ANY <ss_exp>)
SELECT <sel_element_list>
FROM T1, T2, ..., Tn, slice
WHERE <conditions> AND
      slice.geom.st_intersects(VALIDS(T1)) AND
      ... AND slice.geom.st_intersects(VALIDS(Tn))
```

where ST_Union is an aggregate function that calculates the geometrical union of the geometries retrieved by the query. st_intersects is a geometrical function testing whether the geometry on which it is called intersects the geometry provided as parameters.

The CURRENT semantics

ST4SQL :

```
SEMANTICS CURRENT ON <sdim> WITH SGRANULARITY <sG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL:

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn,
     sgranularity AS sg
     INNER JOIN node AS n ON (n.granularity = sg.id)
WHERE <conditions> AND sg.name = <sG> AND
      n.geom.st_intersects(VALIDS(T1)) AND
      ... AND n.geom.st_intersects(VALIDS(Tn)) AND
      n.geom.st_contains(<pos>)
```

where <pos> is the actual user position and st_contains is a geometrical function checking whether the geometry on which it is called contains the geometry provided as parameter.

The SEQUENCED semantics

ST4SQL:

```
SEMANTICS SEQUENCED ON <sdim> WITH SGRANULARITY <sG>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
```

SQL:

```
SELECT nl.label, n.geom, <sel_element_list>
FROM T1, T2, ..., Tn,
     sgranularity AS sg
     INNER JOIN node AS n ON (n.granularity = sg.id)
     INNER JOIN nodelabel AS nl ON (n.label = nl.id)
WHERE <conditions> AND sg.name = <sG> AND
      n.geom.st_intersects(VALIDS(T1)) AND
      ... AND n.geom.st_intersects(VALIDS(Tn))
```

The NEXT semantics

ST4SQL:

```
SEMANTICS NEXT ON <sdim> WITH SGRANULARITY <sG>
          ORDERED BY <sr> THROUGH <attr>
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <jconds> AND <sconds>
```

SQL:

```
WITH instance AS
     (SELECT * FROM T1, ..., Tn WHERE <jconds>)
SELECT <sel_element_list>
FROM instance AS t INNER JOIN instance AS tnext
         ON t.<attr> = tnext.<attr>,
     sgranularity AS sg
     INNER JOIN node AS n ON (n.granularity = sg.id)
     INNER JOIN node AS nnext ON (nnext.granularity = sg.id)
WHERE sg.name = <sG> AND n.geom.st_intersects(VALIDS(t)) AND
      nnext.geom.st_intersects(VALIDS(tnext)) AND
      <sconds> AND NOT EXISTS (SELECT * FROM node
            WHERE node.granularity = sg.id AND
                  evaluate(<sr>,n.geom,node.geom) AND
                  evaluate(<sr>,node.geom,nnext.geom) )
```

where `evaluate(r,g1,g2)` is a function the evaluate the spatial relationship `r` between granule `g1` and `g2`.

When the `NEXT` semantics is applied, the user can refer in the `SELECT` and `WHERE` clauses to the value of an attribute in the successor tuple by using the

NEXT(<attr>) function. That is translated in the SQL query as tnext.<attr>, while all other attributes are refered as t.<attr>.

The SELECT clause of the resulting SQL query contains all the attributes named by the user in the original ST4SQL query, where, eventually, the NEXT(<attr>) function is translated as explained.

C.1.3 Spatio-temporal semantics

Spatio-temporal semantics provide to apply the specified temporal and spatial semantics once the right spatial granularities have been selected (i.e., those spatial granularities valid at the same time of the considered tuples). Here, we do not specify the translation of spatio-temporal semantics for all possible combinations of temporal and spatial semantics. Instead, we report the SQL query, produced from the ST4SQL one, where

- conditions must be added to the WHERE clause and
- attributes must added to the SELECT clause

as provided by the given temporal and spatial semantics as we explained above in this chapter. This SQL query retrieves the right spatial granularities.

Spatio-temporal semantics without the SIMPLIFY token

ST4SQL:

```
SEMANTICS <tsem> ON <tdim> AND
          <ssem> ON <sdim> WITH STGRANULARITY <stG>
SELECT <sel_element_list>
FROM <from_list>
WHERE <conditions>
```

Let Q be this query.

SQL:

```
SELECT <sel_element_list, [[Q]]^SELECT_<ssem>, [[Q]]^SELECT_<tsem>
FROM |STGRANULARITY| AS stg
    JOIN evolution AS ev ON stg.evolution=ev.id
    JOIN validtimehistory AS vth ON vth.evolution=ev.id
    JOIN |SGRANULARITY| AS sg ON vth.|SGRANULARITY|=sg.id
    JOIN |TGRANULARITY| AS tg ON stg.|TGRANULARITY|=tg.id
    JOIN granule AS gr ON gr.granularity=tg.id,
    [[Q]]^FROM_<ssem>, [[Q]]^FROM_<tsem>
WHERE stg.name=<stG> AND
      INTERSECT((gr.start,gr.|END|),(vth.since,vth.to)) AND
      [[Q]]^WHERE_<ssem> AND [[Q]]^WHERE_<tsem>
```

where $[\![Q]\!]^{cl}_{sem}$ represents the *cl* clause in the translation of Q with respect to the temporal or spatial semantics *sem*. It is calculated by applying the translation rules presented above for the semantics *sem*.

Spatio-temporal semantics with the SIMPLIFY token

ST4SQL:

```
SEMANTICS <tsem> ON <tdim> AND <ssem> ON <sdim>
          WITH STGRANULARITY <stG>
          SIMPLIFY BY <s_aggr_funct>
SELECT <sel_element_list>
FROM <from_list>
WHERE <conditions>
```

SQL:

```
SELECT <sel_element_list, [[Q]]^SELECT_<ssem>, [[Q]]^SELECT_<tsem>
FROM |STGRANULARITY| AS stg
    JOIN |TGRANULARITY| AS tg
         ON stg.|TGRANULARITY|=tg.id
    JOIN granule AS gr ON gr.granularity=tg.id,
    eval(<s_aggr_funct>,stg.evolution,
         (SELECT sg.id
          FROM validtimehistory AS vth
               JOIN |SGRANULARITY| AS sg
                    ON vth.|SGRANULARITY|=sg.id
          WHERE vth.evolution=stg.evolution AND
                INTERSECT((gr.start,gr.|END|),
                (vth.since,vth.to)))),
    [[Q]]^FROM_<ssem>, [[Q]]^FROM_<tsem>
WHERE stg.name=<stG> AND [[Q]]^WHERE_<ssem> AND [[Q]]^WHERE_<tsem>
```

where the eval function applies the <s_aggr_funct> spatial aggregate function to spatial granularities passed as third argument and retrieved in the nested query.

C.2 The SELECT and WITH clauses

ST4SQL:

```
SELECT <sel_element_list> WITH <exp> [AS] <dim>
FROM <from_list>
```

SQL:

```
SELECT <sel_element_list>, <exp> AS <dim>
FROM <from_list>
```

C.3 The WHEN and WHEREABOUTS clauses

ST4SQL:

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions>
WHEN <temp-conditions>
WHEREABOUTS <spat-conditions>
```

SQL:

```
SELECT <sel_element_list>
FROM T1, T2, ..., Tn
WHERE <conditions> AND
      <temp-conditions> AND
      <spat-conditions>
```

C.4 The GROUP BY and HAVING clauses

C.4.1 Temporal grouping

ST4SQL:

```
SELECT TGROUP(<t_attr>), <sel_element_list>
FROM <from_list>
GROUP BY <t_attr> ON <tG>, <group_element_list>
HAVING <g_cond>
```

SQL:

```
SELECT <tG>_granules.extent, <sel_element_list>
FROM <from_list>,
     (SELECT g.index, union((g.start,g.end)) AS extent
      FROM t-granularity AS tg, granule AS g
      WHERE tg.name = <tG> AND g.tgranularity = tg.id
      GROUP BY g.index) AS <tG>_granules
WHERE intersecta(<t_attr>,<tG>_granules.extent)
GROUP BY <tG>_granules.extent, <group_element_list>
HAVING <g_cond>
```

Where `union` is an aggregate function we defined for calculating the temporal extent of a granule. It is necessary because a granule may be composed by several disconneted intervals. Thus, we use the `union` function and the subquery to calculate an array containing all the intervals (one for each disconneted part of a granule) in each granule. `intersecta(i,ia)` is a function we defined to check whether a time interval or point `i` intersects any interval in an array of intervals, `ia`.

The TGROUP(<t_attr>) function may be included in the <sel_element_list> attribute list in order to include in the output relation the group used to aggregate <t_attr>. Suppose <t_attr> is grouped using the <tG> granularity, TGROUP(<t_attr>) is translated into SQL as <tG>_granules.extent.

As usual, attributes that appear in the SELECT clause must appear also in the GROUP BY clause.

Conditions provided in the HAVING clause are translated, according to functions used in them, by applying recursively the rules introduced in the rest of the chapter.

C.4.2 Spatial grouping

ST4SQL:

```
SELECT SGROUP(<s_attr>), <sel_element_list>
FROM <from_list>
GROUP BY <s_attr> ON <sG>, <group_element_list>
HAVING <g_cond>
```

SQL:

```
SELECT n.geom, <sel_element_list>
FROM <from_list>, s-granularity AS sg, node AS n
WHERE sg.name = <sG> AND n.granularity = sg.id AND
      n.geom.st_intersects(<s_attr>)
GROUP BY n.geom, <group_element_list>
HAVING <g_cond>
```

As usual, attributes that appear in the SELECT clause must appear also in the GROUP BY clause.

Conditions provided in the HAVING clause are translated, according to functions used in them, by applying recursively the rules introduced in the rest of the chapter.

Sommario

In molti campi di ricerca, i ricercatori hanno la necessità di memorizzare, gestire e interrogare dati spazio-temporali. Tali dati sono classici dati alfanumerici arricchiti però con una o più componenti temporali, spaziali e spazio-temporali che, con diversi possibili significati, li localizzano nel tempo e/o nello spazio. Ambiti in cui tali dati spazio-temporali devono essere raccolti e gestiti sono, per esempio, la gestione del territorio o delle risorse naturali, l'epidemiologia, l'archeologia e la geografia. Più in dettaglio, per esempio nelle ricerche epidemiologiche, i dati spazio-temporali possono servire a rappresentare diversi aspetti delle malattie e delle loro caratteristiche, quali per esempio la loro origine, espansione ed evoluzione e i fattori di rischio potenzialmente connessi alle malattie e al loro sviluppo. Le componenti spazio-temporali dei dati possono essere considerate come dei "meta-dati" che possono essere sfruttati per introdurre nuovi tipi di analisi sui dati stessi. La gestione di questi "meta-dati" può avvenire all'interno di diversi framework proposti in letteratura. Uno dei concetti proposti a tal fine è quello delle granularità. In letteratura c'è ampio consenso sul concetto di granularità temporale, di cui esistono framework basati su diversi approcci. D'altro canto, non esiste invece un consenso generale sulla definizione di un framework completo, come quello delle granularità temporali, per le granularità spaziali e spazio-temporali. Questa tesi ha lo scopo di riempire questo vuoto proponendo un framework per le granularità spaziali e, basandosi su questo e su quello già presente in letteratura per le granularità temporali, un framework per le granularità spazio-temporali. I framework proposti vogliono essere completi, per questo, oltre alle definizioni dei concetti di granularità spaziale e spazio-temporale, includono anche la definizione di diversi concetti legati alle granularità, quali per esempio le relazioni e le operazioni tra granularità. Le relazioni permettono di conoscere come granularità diverse sono legate tra loro, costruendone anche una gerarchia. Tali informazioni sono poi utili al fine di conoscere se e come è possibile confrontare dati associati e rappresentati con granularità diverse. Le operazioni permettono invece di creare nuove granularità a partire da altre granularità già definite nel sistema, manipolando o selezionando alcune loro componenti.

Basandosi su questi framework, l'obiettivo della tesi si sposta poi sul mostrare come le granularità possano essere utilizzate per arricchire basi di dati spazio-temporali già esistenti al fine di una loro migliore e più ricca gestione e inter-

rogazione. A tal fine, proponiamo qui una base di dati per la gestione dei dati riguardanti le granularità temporali, spaziali e spazio-temporali. Nella base di dati proposta possono essere rappresentate tutte le componenti di una granularità come definito nei framework proposti. La base di dati può poi essere utilizzata per estendere una base di dati spazio-temporale esistente aggiungendo alle tuple di quest'ultima delle referenze alle granularità dove quei dati possono essere localizzati nel tempo e/o nel spazio.

Per dimostrare come ciò possa essere fatto, nella tesi introduciamo la base di dati sviluppata ed utilizzata dal Servizio Psichiatrico Territoriale (SPT) di Verona. Tale base di dati memorizza le informazioni su tutti i pazienti venuti in contatto con l'SPT negli ultimi 30 anni e tutte le informazioni sui loro contatti con il servizio stesso (per esempio: chiamate telefoniche, visite a domicilio, ricoveri). Parte di tali informazioni hanno una componente spazio-temporale e possono essere quindi analizzate studiandone trend e pattern nel tempo e nello spazio. Nella tesi quindi estendiamo questa base di dati psichiatrica collegandola a quella proposta per la gestione delle granularità. A questo punto i dati psichiatrici possono essere interrogati anche sulla base di vincoli spazio-temporali basati su granularità.

L'interrogazione di dati spazio-temporali associati a granularità richiede l'utilizzo di un linguaggio d'interrogazione che includa, oltre a strutture, operatori e funzioni spazio-temporali per la gestione delle componenti spazio-temporali dei dati, anche costrutti per l'utilizzo delle granularità nelle interrogazioni. Quindi, partendo da un linguaggio d'interrogazione spazio-temporale già presente in letteratura, in questa tesi proponiamo anche un linguaggio d'interrogazione che permetta ad un utente di recuperare dati da una base di dati spazio-temporale anche sulla base di vincoli basati su granularità. Il linguaggio viene introdotto fornendone la sintassi e la semantica. Inoltre per mostrare l'effettivo ruolo delle granularità nell'interrogazione di una base di dati clinica, mostreremo diversi esempi di interrogazioni, scritte con il linguaggio d'interrogazione proposto, sulla base di dati psichiatrica dell'SPT di Verona. Tali interrogazioni spazio-temporali basate su granularità possono essere utili ai ricercatori ai fini di analisi epidemiologiche dei dati psichiatrici.