JOURNAL OF SPATIAL INFORMATION SCIENCE
Number 20 (2020), pp. 167–201
doi:10.5311/JOSIS.2020.20.555

RESEARCH ARTICLE

# Ontology of core concept data types for answering geo-analytical questions

Simon Scheider[1], Rogier Meerlo[1], Vedran Kasalica[2], and Anna-Lena Lamprecht[2]

[1]Department of Human Geography and Spatial Planning, Utrecht University, the Netherlands
[2]Department of Information and Computing Sciences, Utrecht University, the Netherlands

*Received: July 29, 2019; returned: November 12, 2019; revised: January 21, 2020; accepted: March 18, 2020.*

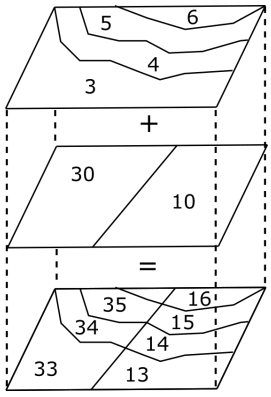
**Abstract:** In geographic information systems (GIS), analysts answer questions by designing workflows that transform a certain type of data into a certain type of goal. Semantic data types help constrain the application of computational methods to those that are meaningful for such a goal. This prevents pointless computations and helps analysts design effective workflows. Yet, to date it remains unclear which types would be needed in order to ease geo-analytical tasks. The data types and formats used in GIS still allow for huge amounts of syntactically possible but nonsensical method applications. Core concepts of spatial information and related geo-semantic distinctions have been proposed as abstractions to help analysts formulate analytic questions and to compute appropriate answers over geodata of different formats. In essence, core concepts reflect particular interpretations of data which imply that certain transformations are possible. However, core concepts usually remain implicit when operating on geodata, since a concept can be represented in a variety of forms. A central question therefore is: Which semantic types would be needed to capture this variety and its implications for geospatial analysis? In this article, we propose an ontology design pattern of *core concept data types* that help answer geo-analytical questions. Based on a scenario to compute a liveability atlas for Amsterdam, we show that diverse kinds of geo-analytical questions can be answered by this pattern in terms of valid, automatically constructible GIS workflows using standard sources.

**Keywords:** core concepts of spatial information, semantic data types, operational signatures, GIS workflow synthesis, geo-analytical question answering

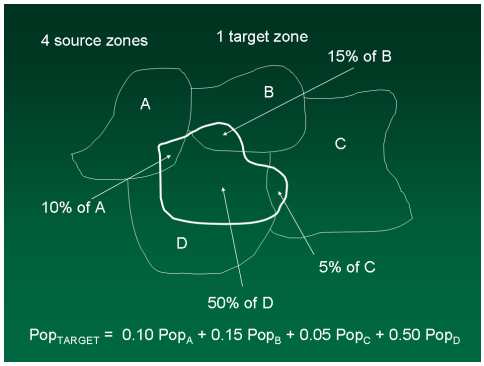
© by the author(s) Licensed under Creative Commons Attribution 3.0 License

# 1 Introduction

It is still common for geospatial analysts to capture their tasks in the language of their favorite tools, such as QGIS, ArcGIS, or R [51]. Within the languages of such a software, however, it remains very hard to distinguish essential pieces of knowledge needed to solve these analytic tasks from mere software artifacts and data formats. Suppose our task is to assess the liveability of Amsterdam. In this case, it is irrelevant whether the data is given in the form of a shapefile with vector polygons or in the form of raster cells. What we need to know is whether the data represents spatially homogeneous values of landuse or rather statistical summaries of population numbers, since only the former can be combined in overlay operations (Fig. 1a) to assess a city's liveability, while the latter requires different kinds of operations, such as areal interpolation [20], cf. Fig. 1b.




(a) Vector overlay method.




(b) Simple areal interpolation method. Image by kind permission of Michael Goodchild.

Figure 1: Methods for spatially combining polygon data. Which one is applicable for analysing a given polygon data set?

It would tremendously improve the analytic process if we could augment *data* with this kind of knowledge, instead of trying to decipher a script or interpret software formats and textual data descriptions [36, 39]. Eventually, this may allow us to translate geo-analytical questions into data sources and software tools [54] to automate the synthesis of answers[1].

Some progress has recently been made by incorporating *semantic types*[2] into scripting languages such as R[3]. Still, it remains unclear which types of spatial information would need to be distinguished in principle for geo-analytical purposes. *Core concepts of spatial information* were introduced by [44] as a simple conceptual and computational interface to geographical information system (GIS). They include[4] *field* (quality surface measurable everywhere on a metric space), *object* (spatially bounded discrete entity with qualities), *event* (temporally bounded discrete entity with qualities) and *network* (quantified re-

[1] Note the difference to ordinary question-answering (QA) systems, which are usually understood in terms of translations to queries on factoid knowledge bases [5]

[2] With the term semantic type, we mean a data type or a function type specified in a Semantic Web taxonomy or ontology.

[3] cf. chapter 6 in https://r-spatial.org/book for the incorporation of geo-analytical attribute types in R, see also [60]

[4] http://spatial.ucsb.edu/core-concepts-of-spatial-information/

lation between objects), which capture spatial information contents in an implementation-independent manner. Core concepts can be used both for formulating spatial questions [63] as well as for generating answers [45] on a higher abstraction level. Furthermore, they capture the provenance of spatial information [55] and determine the applicability of geo-analytic tools [59], e.g., in GIS workflows [53] and for geo-analytical question-answering (QA) with corresponding task descriptions [54].

However, although core concepts play an important role in determining geo-analytic tasks, they are primarily artefacts of human interpretation and measurement, and as such, do not necessarily pertain to any formal characteristics of data. Though an analyst will frequently use these concepts when interpreting and manipulating geodata, in doing so, interpretations remain largely unnoticed and may even be ambiguous. For example, when using point vector data from noise measurements, it might be implicitly assumed that the points represent a finite sample from a spatial noise field, and not a sample of locations of objects emitting noise. Yet, the data type itself does not tell us and might be interpreted in both ways. It is therefore essential not to confuse core concepts with the data types that represent them: A field lies in the eye of the beholder and is considered to be defined everywhere on its domain. Yet, there are many ways to represent such a field by the finite means of a computer, spanning both vector and raster formats [55] (cf. Sect. 5). And vice versa, a given data set might be interpreted in various ways, which has consequences for its usefulness. It is therefore crucial to find out about the different ways geodata can be interpreted in terms of core concepts, and how this can be made explicit in a semantic type system. This would allow us to add the missing conceptual detail in current data descriptions, in order to assess whether geo-analytical questions are answerable using given GIS methods, by synthesizing them into executable workflows.

In this article, we formalize the diversity of ways how core concepts can be represented with common geodata types[5], and how they can be transformed to enable the answering of geo-analytical questions in terms of automated workflows. First, we create a conceptual layered metadata model for signatures of common GIS operations. At the highest abstraction level, the model is based on human-understandable core concepts. At lower levels, core concepts are reified into combinations of machine-operable geodata and attribute types. Second, we demonstrate with a working prototype how this type system, in the form of a lightweight ontology, enables the automatic synthesis of GIS workflows answering geo-analytical questions from given data and tool sources. To the best of our knowledge, this has so far not been subject of any systematic investigation and testing. The contributions of this article are threefold:

- A systematic investigation of data type representations of core concepts of spatial information
- A Web Ontology Language (OWL) design pattern that can be used for annotating and reasoning over GIS tools and their data sources with basic distinctions relevant for geo-analytics. Note that geodata retrieval is, however, not in scope.
- A novel way of formalizing and answering geo-analytical questions through automated workflow synthesis

In addition, the article addresses also GIS practicioners and teachers interested in a systematic account of analytic tasks and GIS workflow design.

[5]With the term geodata type, we denote data types that are currently implemented in GIS.

In the following, we first introduce a design and evaluation scenario in Sect. 2 which contains simple but common examples for geo-analytical questions and corresponding datasets. We then review previous work on conceptualizing GIS, workflow synthesis, and geo-analytical problem solving (Sect. 3). We then introduce semantic distinctions relevant to answering questions in terms of workflows, including core concepts, geometric types of layers for representing these core concepts, as well as levels of measurement (Sect. 4). Using OWL class definitions, semantic distinctions are combined into an ontology design pattern which is used to add semantic type signatures to common GIS operations and data sources for workflow synthesis (Sect. 5). We finally (Sect. 6) demonstrate how the geo-analytical questions from the scenario can be expressed in a workflow synthesis language. Using a prototype we test how well our type signatures enable us to synthesize answers in terms of valid GIS workflows.

## 2 How livable is Amsterdam for elderly people?

We start with typical geo-analytical questions that can be handled within a GIS. Our scenario focuses on assessing the liveability of neighborhoods for elderly people. It was taken from a GIS course at Amsterdam University[6] involving openly available data from the City of Amsterdam[7]. The task is to assess liveability of postcode areas (PC4) within Amsterdam in terms of different urban environmental factors that make the area liveable for elderly people:

1. Since elderly people make use of particular kinds of sport facilities (such as for Pétanque), what is the amount of such facilities within each PC4 area? Sport facilities are given as *point vector* data with facility type as attribute.
2. Since elderly people may want to meet with their peers, what is the rate of elderly living in PC4 areas? Data is given on the "Buurt" (neighborhood) level as a vector polygon map by the central Bureau voor de Statistiek (CBS)[8], see Fig. 2a.
3. Since elderly people prefer parks in their neighbourhood which allow them to take a walk: What is the accessibility of green in PC4 neighborhoods? Data is given in terms of a landuse vector data set[9], see Fig. 2b.
4. Since older people are sensitive to noise, what is the average noise pollution in PC4 neighborhoods? Data is given in terms of a contour vector map with polygons denoting noise intervals, see Fig. 2c.

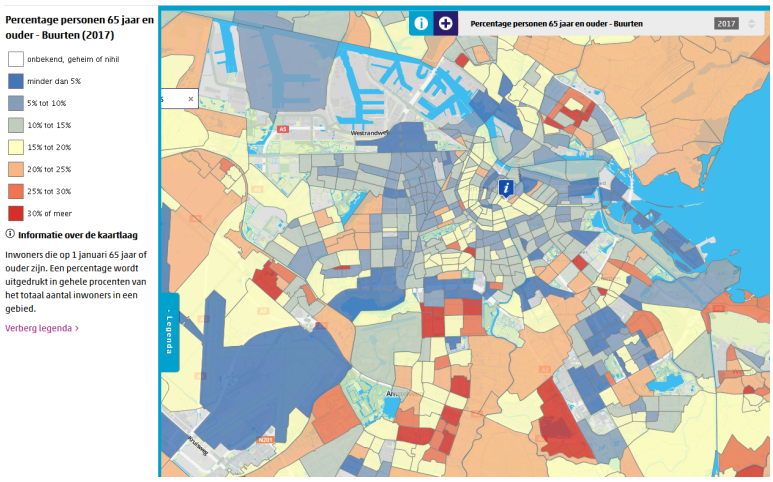
These questions are used as *competency questions* for ontology design and validation. Competency questions are widely used for two purposes in ontology engineering [26, 37], in a way similar to software engineering. First, competency questions state application requirements. If the ontology is designed based on competency questions, then it may be capable of answering equivalent kinds of questions in the application domain. Second, for the purpose of validation, the ontology is queried with competency questions to verify whether valid answers are indeed in scope. Answers may be computed in different ways, e.g., based on inference or based on data queries. In our case, note that (1) answers to

[6] https://www.vu.nl/en/study-guide/2018-2019/minor/l-r/national-geo-information/?view=module&origin=51311992x50721889&id=50855870

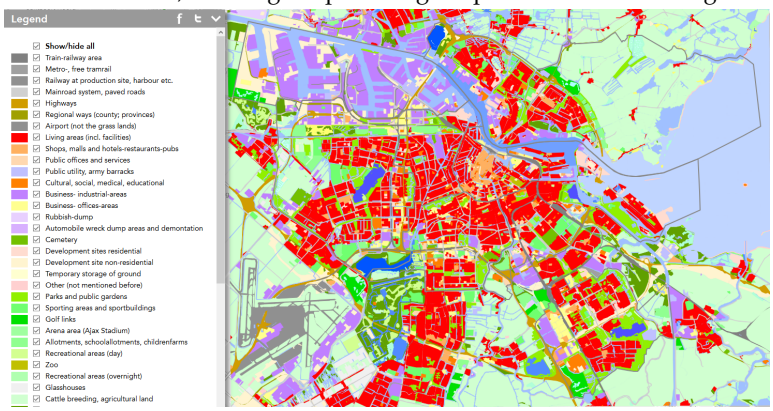
[7] https://maps.amsterdam.nl/open_geodata/

[8] https://www.cbs.nl/nl-nl/dossier/nederland-regionaal/wijk-en-buurtstatistieken

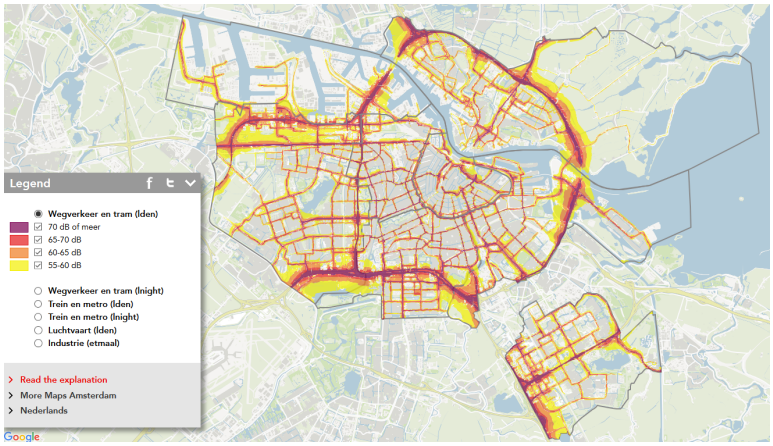
[9] https://maps.amsterdam.nl/open_geodata/?k=152




(a) CBS Buurt statistics, showing the percentage of persons over 65 in neighborhoods.




(b) Landuse map (Grondgebruikskaart) of the Amsterdam Municipality.




(c) Noise map of the Amsterdam Municipality, with intervals given in dB.

Figure 2: Map data sources used to assess liveability, cf. https://maps.amsterdam.nl/open_geodata/ and https://cbsinuwbuurt.nl/.

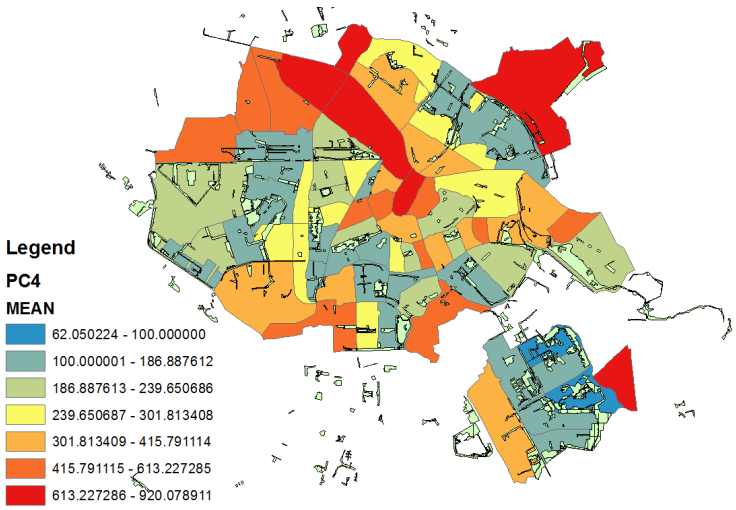



Figure 3: Liveability map showing average distance to green areas in Amsterdam within PC4 areas.

our questions need to be given *indirectly*. That is, not in terms of answers from a knowledge base, but rather in terms of analytic workflows which generate a map that contains a valid answer. The required outcome of the analysis might look similar to the map in Fig. 3, a choropleth map with liveability measured per PC4 area. We need to test our ontology therefore in terms of its *inherent capability of picking a workflow which would provide a valid answer*[10]. In addition, note that (2) the selection of suitable datasets for the task, or the modeling of domain knowledge (e.g., about urban noise or sport facilities) are out of scope[11].

Though our validation is limited in both the number of questions, as well as in the amount of synthesized workflows per question that are analyzed, our set of questions actually covers a wide range of typical GIS applications which go beyond the livability scenario used as illustration. Though the geodata types of sources look homogeneous (polygon vector data with some numeric attribute or label), they actually require very different analytic workflows to generate useful answers. This includes, e.g., vector operations such as spatial join and areal interpolation, in order to aggregate different kinds of vector data sources, as well as different raster operations like zonal and local map algebra, in order to aggregate distance information. The questions we chose thus really stand for a diverse set of more general spatial analysis problems. Below, we will explain how the included data sources actually reflect very different core concepts, and thus ask for correspondingly different treatments requiring a significant level of GIS expertise. We will illustrate that the standard geodata types *Vector* and *Raster* [29] are insufficient for this purpose. In what follows, we investigate which concepts would allow us to pick the right kind of workflow.

[10]This task was called *indirect question-answering* in [58].
[11]For these tasks, we believe that lightweight ontologies, as used in this article, are less suitable.

# 3 Preliminary work on programmatically solving geo-analytical tasks

The general task briefly sketched by the example above has precursors in the field of Geographic Information Science (GIScience) and corresponding computer languages, as well as in workflow synthesis and QA. In GIScience, authors have proposed earlier to use semantic concepts of geoinformation in order to support tool and data selection for geographic problem solving. For example, O'Brian and Gahegan [50] suggested to use backward chaining on an operational rule set in order to identify tool sources starting from a goal concept. This work was based on earlier suggestions of functional taxonomies of GIS operations [2]. However, several major problems remained, namely to decide about an ontology for GIS tool and data description, about composition procedures that capture geo-analytic constraints, as well as to handle question-based interfaces.

The problem of generalizing GIS transformations in terms of semantic concepts was addressed by Gahegan in [23] from the perspective of remote sensing, suggesting an early approach closely related to our work. Camara and others have suggested functional type systems for GIS operations [12, 13] from the perspective of fields and objects. Cova and Goodchild suggested a model integrating the two concepts in [18].

The problem reappeared in a slightly different form in Web service chaining and geoprocessing [1, 49, 64] and is open to the present day [38]. Similar to Component-based Software Engineering (CBSD, [35]) and service composition in Service Oriented Architecture (SOA, [19]), a component or a service implements a particular functionality, requires a particular input, and produces a specific output. Components are then combined to produce working software, and multiple services can be composed to create a new service. Automated discovery and chaining of components or services require knowing their interfaces, i.e., their input and output data types. In a similar way, we aim to automate the manual effort of composing GIS operations implemented in existing software.

An abstract data type (ADT) is a somewhat related concept. It is an abstraction of a type of data in terms of *behavior*, i.e., in the sense of the operations applicable to the data [47]. ADTs might be implemented in different ways and can be arranged in hierarchies, such that subtypes inherit the behavior of supertypes. This is done in order to constrain the behavior of geospatial operations in a particular computational environment. There have been efforts to conceptualize GIS notions in such a way, especially for querying spatial databases [31, 32]. For example, [31] proposed a typed model of a geometric database focusing on geometric objects and their attributes. While these types still lack semantics at the level of core concepts, examples of spatial concepts such as moving objects and trajectories were recently developed in a comparable manner [32, 33]. While we consider such type specifications a useful source for our work, our focus is less implementation centric, since we need to annotate sources across many implementations.

The related workflow composition problem is still a substantial challenge for information science [28] and plays a central role also in any geospatial information infrastructure [48]. Workflow synthesis has recently been approached using loose programming techniques [46], which provides a way of loosely specifying workflows using semantic type hierarchies that are used as constraints on workflow generation (cf. Sect. 6). This new technique was recently applied to the GIS domain [1, 40, 41].

The problem of appropriate interfaces for geo-analytic problems was approached in terms of conversational GIS [11], and recently addressed also in the context of QA systems

on geographic information [14, 65]. We believe workflow composition is a design task requiring the thorough conceptual basis of spatial questions at its core [63], and vice versa, workflow synthesis is essential to geographic QA. In a rather fundamental sense, workflows *answer* geo-analytical questions. This also motivates a novel way of looking at QA in more indirect terms [58], namely as the task of finding ways to transform data into valid answers, as opposed to simply querying over data. We call this problem *geo-analytical QA*. It forms the context of our article, in which we suggest a semantic basis for a corresponding type system.

# 4 Semantic distinctions needed for geospatial analysis

In this section, we review semantic distinctions relevant for geospatial analysis, namely how geometric layer types can represent core concepts of spatial information in terms of their attributes on different levels of measurement. We discuss the implications for geospatial analysis and suggest a lightweight formalization in the Web Ontology Language (OWL)[12], as a basis for the data type pattern[13] composed in Sect. 5. We use the Description Logic (DL) notation for OWL. A primer for the DL notation can be found in [43].

## 4.1 Types in a lightweight ontology

To capture the semantic distinctions and hierarchical dependencies of geospatial operations, we will make use of DL classes. This gives us a way to easily define new types from existing types in terms of class unions and intersections, resulting in a data taxonomy (a directed acyclic graph of subsumptions, inheriting the applicability constraints from superconcepts). It also allows us to capture arbitrary combinations of different semantic dimensions, such as geometric layer types and the core concepts they represent. For example, the ways how spatial interpolation methods are bound to the different combinations of vector or raster data with the field concept.

The semantic types proposed in this paper are metadata models specific to the GIS domain, yet they can be used across many implementations. In fact, in contrast to ADT's, our types are not meant to be implemented. Rather, they are supposed to be used for annotating methods in existing software implementations, in order to improve their reuse, much like *lightweight ontologies* [6,39]. For the same reason, our formalization lacks any specifics of the underlying concepts. We will discuss below some of the ontological implications of core concepts that are essential for understanding the operational constraints that are imposed on types. Yet, our ontology captures just the semantic distinctions needed for constraining geo-analytical tasks, without giving any formal account of these tasks or capturing general knowledge about the concepts. Correspondingly, the reasoning capabilities of OWL are used here only to constrain workflow construction via subsumption relations, as described in Sect. 6. Thus, our validation is not in terms of inferential power of the ontology with respect to the concepts, but in terms of generative power and quality of workflow synthesis.

[12] http://www.w3.org/TR/owl2-syntax/
[13] *ccd:* http://geographicknowledge.de/vocab/CoreConceptData.ttl

## 4.2 Core concepts vs. geodata types

Core concepts of spatial information were proposed by [44, 45] as generic interfaces to GIS in the sense of conceptual "lenses" through which the environment can be studied. Though they have been used in the sense of ADTs, they are considered results of human cognition and interpretation residing in an analyst's head, and thus should not be understood as data types or formats. We summarize the most important ideas in the following. Kuhn distinguishes one *base concept* (location) and two *quality concepts* (granularity and accuracy) from the following *content concepts*, on which we focus in this article:

- *Fields*: Are understood as particular kinds of functions [13, 24, 55] whose domain are locations which allow for distance assessments, and whose range may be any kind of quality. As quality values are separated by spatial distance, one can study change of a field as a function of spatial distance. Fields also offer the possibility of determining quality values at *arbitrary locations* in their domain [60]. Missing quality values can therefore be estimated by *interpolation*. Fields change in time, in the sense that the field function as a whole is only a snapshot [55] of a spatio-temporal field which changes from one moment to the next. In contrast to objects, fields do not change their locations. This concept is closely related to the notion of a field in physics [21]. Prime examples are temperature fields.
- *Objects*: Are understood in terms of functions from time to locations and qualities. Objects are prototypical examples of "endurants" [24], so they can change their location and quality in each time moment while remaining their identity. Ontologically, objects are distinct from fields and events in the sense that they have an identity and that they are fully localized in each moment of their existence, even if this location may be fuzzy. In this way objects give rise to trajectories, which are functions from time to location, and time series, which are functions from time to quality [55]. We consider geographic places as particular kinds of objects [57]. So places are not considered locations, but are rather localizable themselves. We also assume that objects include both, bona fide (perceivable) and fiat (conventional) boundaries.
- *Events*: are understood as entities that, besides having identity and having qualities like objects, *happen* during some temporal interval. Events thus correspond to particular kinds of "occurrents" or "perdurants"[14]. They usually have a start and an end, thus allowing us to determine duration, and they might have objects, fields or spatial networks as participants. In GIS, we usually assume in addition that events are localizable similar to objects[15]. Though this might not be true for all kinds of events, it is usually the case for events in GIS. Prime examples are earth quakes, having a time, a location as well as a magnitude.
- *Networks*: Are understood as functions from pairs of objects to some quality. In this way, networks are able to measure a relationship between these objects. In contrast to relations in logic or to graphs, the relationship is quantified and inheres in the objects, e.g., in terms of an amount of flow. Networks in our sense are, e.g., commuter flow matrices or traffic links in a road network. Regarding time, network functions can be considered similar to endurants, i.e., they can change their quality in time.

[14]Galton [25] suggests that events correspond to historical time, while there is also an experiential time without temporal boundaries, which he calls "process". We follow Kuhn [44] here, suggesting that processes may play a minor role in GIS.

[15]Other authors hold that events are only localizable via their participating objects [27].

Since core concepts have these properties, certain kinds of operations are naturally applied to them. For example, since fields are total functions on a metric space, their quality can be probed at every location and every distance within that space, while object qualities cannot. Objects, on the other hand, can be counted, have spatial parts (mereology) and neighbors (topology) [16], and furthermore give rise to sizes and closeness. Events can be ordered in time, e.g. by Allen's relations [3]. Finally, since networks are relations between objects, they can always be projected to sets of objects by fixing some source or destination.

Due to limited space, we adopted in this article a standard *static* view on core concept data types, though core concepts can also be used for spatio-temporal analysis [55]. Note that all core concepts (not only events) inherently have a temporal dimension as indicated above. Furthermore, while events and networks are important examples of core concepts, in this article we focused only on fields and objects for the same reason. Note also that, while the original purpose of Kuhn's core concepts was (re-) designing a novel interface for GIS, our focus here is on improving the (re-) usability and accessibility of existing tools and data sources. This is based on the following argument: When analysts use a GIS on a geo-data source, they implicitly interpret not only their analytical goal, but also the data source as well as the GIS tool in a way that is best captured by some *core concept transformation*. For example, analysts might interpret a given vector file in terms of a field representation, and search for tools that can handle such field representations with the goal of aggregating the field into a new quality of an object. Since some of the operational constraints of core concepts are retained in this interpretation, analysts are able to select appropriate tools for their purpose. For example, only field representations allow point interpolations [60]. This is what makes core concepts so important for compiling GIS workflows and for answering questions.

Note, however, that core concepts and data types are very different beasts: the former is an interpretation of an analyst, the latter is a data artefact that represents concepts in a rather indirect manner[16]. This also means analysts can switch perspective on a data type, and thus we should expect a large freedom in choosing a data type to represent a certain core concept, leading to a variety of combinations that are actually used in practice and need to be taken into account. We agree with [17] that objects and fields play autonomous roles in the geographic information process, apart from any data structures that might represent them. Therefore any approach that reduces the former to the latter misses the point. Still, data types are needed for programming workflows and geometric layers are fundamental to represent the geometry of core concepts. This conflict in abstraction might be the reason why general taxonomies of GIS functions apparently have been so hard to devise. Some authors have proposed conceptual or even formal models of fields [12] and objects [33] and their relation [18] in GIS. However, we see a lack of distinction between concepts and data types running across these works (cf. [44]).

In the following, we investigate how different geometric layer types and types of attributes are able to capture information about core concepts.

## 4.3 Geometric layer types for representing core concepts

While a *layer* is a fundamental concept for every GIS to compare and combine information based on spatial coincidence, we are used to distinguishing layers (as well as entire GIS)

[16] In particular, we suggest that mathematical properties of core concepts are *not* retained in corresponding types, in the sense that any remaining similarity is not isomorphic.

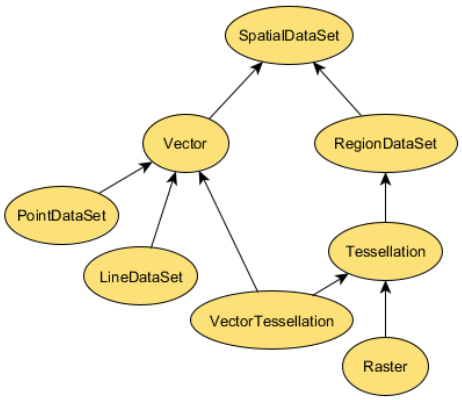



Figure 4: A concept hierarchy of layer types in the CCD ontology, based on geometric characteristics of layers. Arrows denote *rdfs:subClassOf* triples. Note that OWL classes are not mutually exclusive (if not stated otherwise). So a region data set can, e.g., also be a vector data set. See Sect. 4.3.

according to the most prominent geodata types *raster* or *vector*. The latter are subject to famous methodological debates [9,17] and relevant for computational efficiency considerations (such as runlength encoding and spatial indexing) [29]. Still, dividing analysis into raster and vector is often extraneous, not only because there exist trivial translations from raster to vector formats which retain raster principles[17] and vice versa, but also because ignoring these types is often important for geographic analysis. For example, in order to select data for noise pollution assessment in a city, it is less relevant whether noise levels are expressed in vector (contours) or raster format.

We therefore suggest to focus our data types on the *geometric properties* of layers instead, drawing a distinction between layers that are *Tessellations* or not, as well as between point, line or region datasets. A tessellation is a tiling of the plane into regions which are *jointly covering* the plane and *mutually non-overlapping* [8]. Examples are irregular tessellations, such as Choropleths, TINs and Voronoi tilings, as well as tilings with regular polygons, such as regular squares. Tessellations enable us to describe an area without any gaps and without redundancy, and thus to represent the core concept *field*, as we will argue below. In addition, tessellations have an *extent* in the sense of a finite area that is fully covered, which is essential for GIS processing. If we reorder geodata layers according to these principles, we obtain a type hierarchy where what we normally call raster cell data is a subtype of region data (see Fig.4), since the former reflects the special case of a tiling with squares[18].

[17]For example, the highest resolution level of the official statistics of the Dutch Centraal Bureau voor de statistiek (CBS) are CBS "vierkant" vectors (http://www.cbsinuwbuurt.nl/#vierkant500m_aantal_inwoners_2017), which is nothing else than a raster in vector format.

[18]In this article, we consider raster layers to consist of cells, which are squared regions. There is also a raster format that cannot be considered a tessellation because it represents cell centre or cell corner points, as implied by GDAL's 'AREA_OR_POINT' flag in GeoTIFFs (https://gdal.org/drivers/raster/gtiff.html) or NetCDF's CF conventions. This is often used in digital elevation data. We suggest to categorize this latter type differently, namely as a regularized point data set (e.g., a PointMeasure).

Building on the OWL based *analysis data (ada) design pattern*[19] [59], we capture a layer in terms of a *SpatialDataSet*[20], which is a collection of data items which have geometries of the same type as spatial references[21]. As with a table row, different attribute values can be linked to a single data item, which in turn is element of some data set. We distinguish *RegionData*, *LineData* and *PointData* depending on whether these geometry values are points, lines or regions[22]. For example, a *RegionDataSet* is a spatial data set with only RegionData elements:

**Ontological definition 1.** $RegionDataSet \equiv SpatialDataSet \sqcap \forall hasElement.RegionData$

In our model, (cell-) Raster is a subclass of Tessellation, which is a subclass of RegionDataSet:

**Axiom 1.** $Raster \sqsubset Tessellation \sqsubset RegionDataSet$

The distinction between vector and raster reflected in incompatible data formats can be retained by redefining vector layers as *not raster*:

**Ontological definition 2.** $Vector \equiv SpatialDataSet \sqcap \neg Raster$

A vector tessellation can be expressed as the intersection $Vector \sqcap Tessellation$. *PointDataSet* and *LineDataSet* are special types of Vector data, being distinct from each other and from RegionDataSets:

**Ontological definition 3.** $LineDataSet \equiv Vector \sqcap \forall hasElement.LineData$

**Ontological definition 4.** $PointDataSet \equiv Vector \sqcap \forall hasElement.PointData$

**Axiom 2.** $RegionDataSet \sqcap LineDataSet \sqcap PointDataSet \sqsubseteq \bot$

This captures the main intuition that tessellations are bound to be built by either raster or vector regions, yet not both at the same time. It also keeps our types open for further geometric realizations of tessellations, such as TINs or Thiessen polygons.

## 4.4 Attribute types for representing core concepts on different levels of measurement

How can these layer types represent core concepts? To approach this question, first, note that core concepts are basically about *thematic contents* of a layer. Fields as well as objects and events have qualities (such as noise level, building heights and event durations), and our core concept types need to be able to reflect these in terms of attributes. We therefore consider CCD types to be *types of attributes*, and not types of spatial data sets. Following the *ada* ontology, a (spatial data) *Attribute* is described as a value list of some *SpatiaDataSet* (namely the list of values of one of its attributes):

**Ontological definition 5.** $Attribute \equiv ValueList \sqcap \exists ofDataSet.SpatialDataSet$

[19] http://geographicknowledge.de/vocab/AnalysisData.rdf

[20] We use the terms "layer" and "spatial data set" interchangeably here, knowing that there might be a difference that is not captured by our scheme.

[21] That is, we make the assumption that a layer does not mix points, lines and regions.

[22] These types are not necessarily restricted to 2D geometries. However, we focus here on 2D examples.

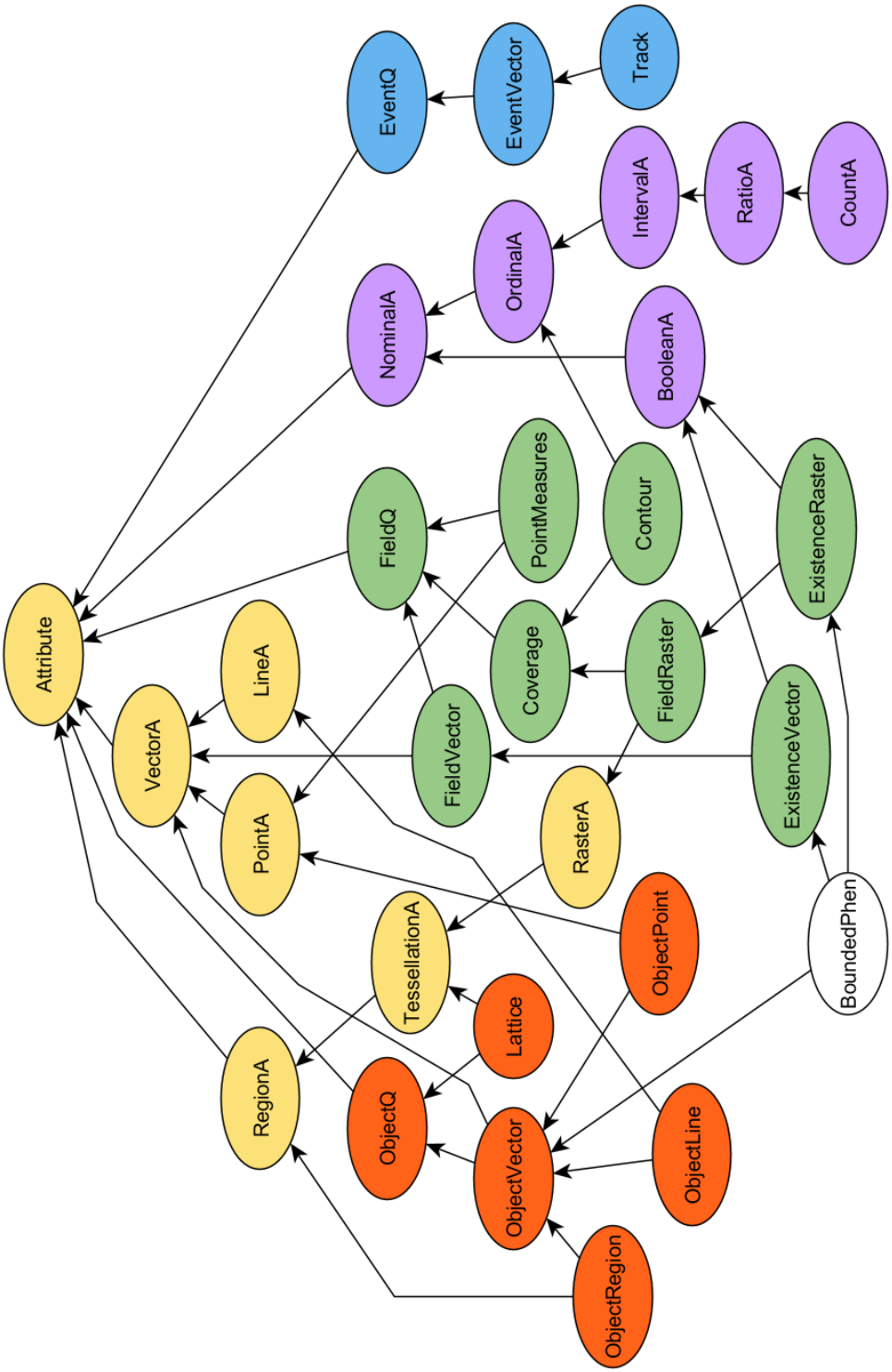
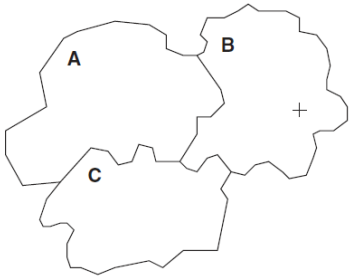



Figure 5: A concept hierarchy of attribute types in the CCD ontology (excerpt), combining geometric layer types, core concepts and measurement levels. The colors green, blue and red stand for the core concepts field, event and object, resp. Yellow classes denote attribute layer types, and violet ones denote levels of measurement. Compare Sect. 4.4.




| | land cover type | average elevation (m) |
|---|---|---|
| A | Forest | 631 |
| B | Urban | 220 |
| C | Water | 42 |
| + | Urban | |

Figure 6: The difference between a field (Coverage) and an object (Lattice) tessellation in terms of self-similarity. Land cover is an example for a coverage, and average elevation is an example for a lattice. The attribute located at the cross is determinable for the coverage, but not for the lattice. Cf. [55].

Each geometric layer type gives rise to a corresponding attribute type. Geometric subtypes of attributes, e.g., a *LineA* attribute (cf. Figs. 4 and 5), are defined as:

**Ontological definition 6.** $LineA \equiv Attribute \sqcap \exists ofDataSet.LineDataSet$

Second, note that such attributes have their own mathematical properties that only indirectly reflect the properties of the corresponding concept. For example, in order to represent a field with irregular tiles, we can use a tessellated representation called *Coverage*[23]. In contrast to the field it represents, a coverage has the property that *parts of its regions have the same attribute value as the entire region*. For instance, within any polygon of a land use coverage, any location has the same landuse value as the attribute of the containing polygon (Fig. 6). We call this attribute property *self-similarity* in the following[24]. This property is precisely what allows us to reconstruct the values of the field at arbitrary locations simply by subdividing geometries. Object or event representations, in contrast, do not have such a property. For example, a corresponding tessellated object representation (called in this article *Lattice*), such as "average elevation within Amsterdam", is not self-similar (Fig. 6). Note that this distinction is *not* captured by saying that objects are discrete and fields are not, since *all* mentioned layer data models are bound to be discrete.

In addition to core concept qualities, we need to take into account further attribute distinctions that are relevant for geo-analytic operations. This primarily includes *measurement levels*, since they largely determine the type of numerical GIS operation that can be applied to data [15], as well as the distinction between *extensive and intensive attributes* [56], which denotes whether attribute values are dependent on the size of their support regions (extensive attributes) or not (intensive attribute). In this article, we do not further focus on the latter distinction, yet our ontology inherits these classes.

Based on the arguments given above, we introduce *classes of spatial data attributes* (see Fig. 5) that represent core concept qualities (ending with a "Q"). A *FieldQ*, e.g., is a spatial data attribute which directly represents the quality values of a field, and similarly for *EventQ* and *ObjectQ*:

[23]See [55]. The term resembles the notion introduced by the Open Geospatial Consortium (OGC) in http://docs.opengeospatial.org/is/09-146r6/09-146r6.html. However, the OGC term does not distinguish core concepts from data types and is practically used only as a supertype of raster formats.

[24]The property is also called *homeomerosity* in the ontology literature [30].

**Axiom 3.** $FieldQ \sqsubseteq Attribute; EventQ \sqsubseteq Attribute; ObjectQ \sqsubseteq Attribute$

An example of a field quality would be temperature, and an example of an object quality would be the size of a building or the average income of a municipality. Note that since our core concept attribute classes are defined orthogonal to geometric layer types, we are free to mix them. This enables us to represent, e.g., a temperature field by point data as well as region data.

We finally capture *levels of measurement* for spatial attributes as a chain of subsumptions, since lower measurement levels are special cases of higher levels and thus imply these higher levels [15, 59]:

**Axiom 4.** $CountA \sqsubseteq RatioA \sqsubseteq IntervalA \sqsubseteq OrdinalA \sqsubseteq NominalA \sqsubseteq Attribute$

Note that count scales are a special case of ratio scales, where not only the meaning of 0 but also of 1 is fixed. We consider a Boolean (True/False) attribute as a special case of a (bi)nominal attribute, as an ordering is not implied:

**Axiom 5.** $BooleanA \sqsubseteq NominalA$

# 5 Core concept data types (CCD) and operational signatures

Each of the distinctions introduced above is relevant for some form of geospatial analysis, and thus for determining a corresponding data type. In principle, the proposed three semantic dimensions could be freely combined for this purpose, resulting in a large number of types, i.e., 6 (Point, Line, Region, Tessellation, Vector, Raster) * 3 (FieldQ, ObjectQ, EventQ) * 6 (NominalA, OrdinalA, IntervalA, RatioA, CountA, BooleanA) = 108. However, not all combinations seem meaningful and practically relevant. In this section, we discuss what we think are the most useful combinations in terms of the *ccd* design pattern[25], and then introduce corresponding signatures for common GIS operations. Cf. the concept hierarchy in Fig. 5. The CCD ontology and all annotation data in this article was checked for consistency using standard OWL reasoners.

## 5.1 CCD ontology pattern

A simplified overview is given in terms of the matrix in Fig. 7, and we define and explain these type combinations in the following based on examples. The borders of layer types in this figure are dotted, reminding us that classes are not mutually exclusive, and thus can overlap each other.

### 5.1.1 Field based types

We start with data representations of a field. The simplest way of representing a field is to represent it in terms of a *pointwise sample of measurements*. We define this data type *PointMeasure* as a field quality represented by the attribute of some point data set:

**Ontological definition 7.** $PointMeasures \equiv FieldQ \sqcap \exists ofDataSet.PointDataSet$

[25] *ccd:* http://geographicknowledge.de/vocab/CoreConceptData.ttl

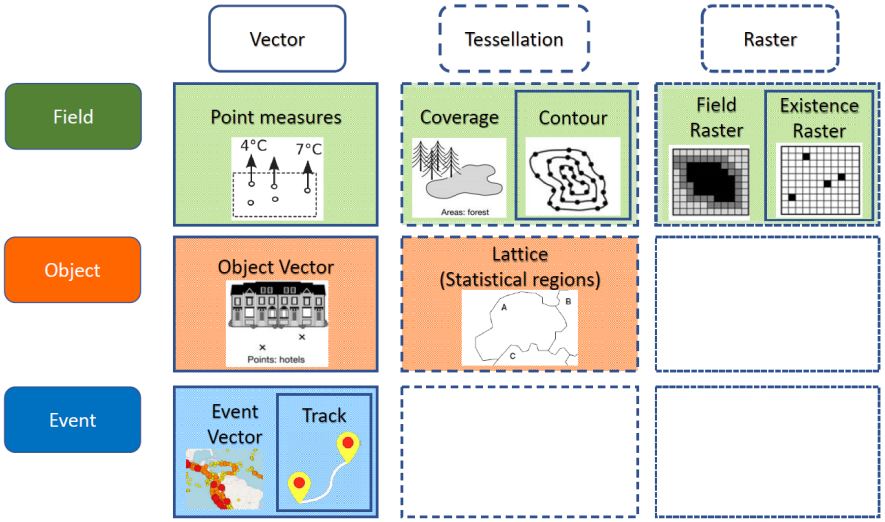



Figure 7: Matrix of data types based on combining geometric layer types with spatial core concepts. Combinations are formalized with DL class constructors, and empty cells are deprecated combinations, as explained in the text. Note that the class *tessellation* intersects with both raster and vector, so matrix cells are not mutually exclusive.

Since it is provable that the underlying data set is not a tessellation (by Axioms 1 and 2), this type of data is incapable of providing an estimate for the field quality for each location in the covered area. Still, the representation is trivially self-similar, since parts of points are always identical. Prime examples of point measures are pointwise representations of air quality or noise measurements. A *LineMeasures* attribute maps lines of homogeneous values of a surface, such as in an *isoline* map:

**Ontological definition 8.** $LineMeasures \equiv FieldQ \sqcap \exists ofDataSet.LineDataSet$

Each line may represent, e.g., a particular height in a digital terrain model. A *FieldRaster*, in contrast, is a typical field representation in remote sensing. It enables spatially continuous analysis since a (cell) raster is a type of tessellation (by Axiom 1):

**Ontological definition 9.** $FieldRaster \equiv FieldQ \sqcap \exists ofDataSet.Raster$

A more general way of representing a field in terms of a tessellation is a *Coverage*, which is defined as a tessellated representation of some field quality:

**Ontological definition 10.** $Coverage \equiv FieldQ \sqcap \exists ofDataSet.Tessellation$

It follows by Axiom 1 that field rasters are special kinds of coverages. Coverages include also irregular tessellations used, e.g., to represent landuse data in terms of nominal classes. Depending on the level of measurement, we can further distinguish *Contours*, which are kinds of coverages whose values are ordinally scaled, denoting quality intervals of fields[26]:

[26]Since contour maps go beyond ordinally scaled coverages, this is not a definition. A formal model of contours was recently suggested by Hahmann and Usery [34]. Note also that we make a distinction between *contour* (region) and *isoline* maps. Isoline maps are of type $LineMeasures$, and are not tessellated. This distinction can be made for Amsterdam datasets that we use for our scenario.

**Axiom 6.** $Contour \sqsubseteq Coverage \sqcap OrdinalA$

Examples for contours are vector based digital elevation models, where each irregular polygon maps some height interval of the terrain. Since contours are coverages and coverages are field representations, they are self-similar[27]. Correspondingly, all coverages provide an estimate of the field quality for each location within their extent. For example, an estimate of height or landuse for each point on the terrain.

An important subtype of field representation is a field raster with Boolean value ranges. We call this type of raster *ExistenceRaster*:

**Ontological definition 11.** $ExistenceRaster \equiv FieldRaster \sqcap BooleanA$

Such field rasters are used to show the existence of diverse kinds of phenomena (including all core concepts) for each location in space. For example, an existence raster may be used to show where a certain type of landuse is located, where a certain height is, or where a certain set of houses is located[28]. The value *True* asserts the presence of the phenomenon at the cell location[29]. Following the same kind of logic, alternatively, we can use Boolean Vector Fields to represent the existence of such phenomena. In this case, irregular vector geometries ("patches") show the existence of a phenomenon under consideration at all locations within the geometry:

**Ontological definition 12.** $ExistenceVector \equiv FieldVector \sqcap BooleanA$

Note that existence rasters and existence vectors can both play the role of an *individuation criterion* to construct new kinds of objects, defined as the whole of all locations for which the existence field is true, such as "the green territory in Amsterdam". In this way, they give rise to new spatial boundaries of a synthetic object that encompasses those parts of the underlying field that correspond to the selection of green landuse.

### 5.1.2 Object based types

In contrast to field based types, object based attribute types are not self-similar, so we cannot assume that parts of their geometries have the same attribute values as the whole. Object vectors are attributes of vector data which directly represent object qualities:

**Ontological definition 13.** $ObjectVector \equiv ObjectQ \sqcap \exists ofDataSet.Vector$

Examples of object vectors are building prizes or heights in a cadastral data set, emission values of industry buildings, or tagged Points of Interest (POI) such as in Open Street Map (OSM). These datasets are not tessellated, and so we can expect spatial gaps where no object (and thus no quality) is located. Think about a dataset of power plant emissions, vs. a dataset of emission concentrations. The former is an object vector, the latter being an example for a point measures representation of a field.

In addition, objects are *countable* and have *boundaries* denoting, e.g., physical borders such as walls or legal boundaries. Note that object vectors as well as existence

[27] Cf. also the discussion of coverages ("inverted fields") in [55].

[28] For this reason, one could further specify existence rasters according to their data origins, as was done in [53]. This is considered future work.

[29] Fuzzifying boolean values would result in an existence raster which points at the vague location of a phenomenon [10].

rasters/vectors afford the determination of a well defined spatial boundary[30]. We therefore summarize these types into a bounded phenomenon supertype:

**Ontological definition 14.** $BoundedPhen \equiv ObjectVector \sqcup ExistenceRaster \sqcup ExistenceVector$

This type is useful to overload proximity and topological operations which make use of boundaries (see examples below).

A *Lattice*[31] is defined as a tessellated representation of an object quality, implying that object boundaries form a tessellation:

**Ontological definition 15.** $Lattice \equiv ObjectQ \sqcap \exists ofDataSet.Tessellation$

The prototype of a lattice is *statistical data* summarized by pre-defined statistical regions. This type of data represents the qualities of particular kinds of objects, such as the mean income of the municipalities in the Netherlands. Note that also in this case, administrative units have a distinctive identity, and attributes are not self-similar. Furthermore, since these regions extensively cover a spatial area without overlap, it becomes possible to aggregate measures of their attributes (e.g., the mean income of the Netherlands). Note that this is not possible with any other kind of data in our schema.

In principle, a possible subtype of lattice would be an object raster, where each raster cell denotes the location of an object. However, such a representation is hardly practical, as it enforces squared boundaries on objects. Indirect raster representations of objects are more common, such as object density rasters or object existence rasters. Yet, the latter are indeed special types of *field rasters* in our scheme[32], and therefore appear as deprecated types in our matrix.

## 5.2 Geocomputational signatures for spatial analysis

Based on CCD data types, we introduce *type signatures* to describe input and output types of GIS operations and to check validity constraints on the automatic composition of GIS operations into a workflow. Signatures of GIS operations are gathered in a *tool description file* which links CCD classes as inputs and outputs to tool identifiers[33], an overview of which is given in Table 1. Operations will be discussed now in their order of appearance, using implementation examples from ArcGIS Pro [34]. However, we expect that an equivalent set of operations is available in other GIS software. Signatures are summarized in a computational diagram in Fig. 8.

### 5.2.1 Operations on Field representations

Prominent FieldRaster operations are Tomlin's map algebra functions [62]. *Local and focal map algebra* (such as local sums or focal averages, and many others) take some FieldRaster as input and generate a new FieldRaster, thus sharing the same signature and operating

[30]Even though some objects (such as mountains) have vague boundaries [9], they are still "bounded" in the sense that we can determine a minimal region containing their location.

[31]Used here in the sense of lattice data in spatial statistics (cf. [52]).

[32]The idea being that spatial distance/existence creates a spatial (ratio-scaled/Boolean) field. This field can be represented as a raster, for example.

[33]https://github.com/simonscheider/SemanticPipelines/blob/master/ToolDescription.ttl

[34]https://pro.arcgis.com/en/pro-app/help/analysis/geoprocessing/basics/geoprocessing-quick-tour.htm

| On | Operation type | Input type | 2nd Input type | Output type |
|---|---|---|---|---|
| Field | Local Map Algebra | FieldRaster | FieldRaster | FieldRaster |
| | Focal Statistics* | FieldRaster | | FieldRaster |
| | Zonal Statistics** | FieldRaster | VectorTessellation | Lattice |
| | Boolean Reclassify | FieldRaster | | ExistenceRaster |
| | Select by attribute | Coverage | | ExistenceVector |
| | Raster to Polygon | NominalA⊓FieldRaster | | Coverage |
| | Create Contours | IntervalA⊓FieldRaster | | Contour |
| | Polygon to Raster | Coverage | | NominalA⊓FieldRaster |
| | Polygon to Raster | Contour | | OrdinalA⊓FieldRaster |
| | Overlay | Coverage | Coverage | Coverage |
| | Point Interpolation | IntervalA⊓PointMeasures | | FieldRaster |
| | Extract Values to Points | FieldRaster | PointA | PointMeasures |
| | Thiessen Polygons | PointMeasures | | Coverage |
| | Least Cost Distance/Path | IntervalA⊓FieldRaster | BoundedPhen | ExistenceRaster |
| Object | Areal Interpolation | Lattice | | Lattice |
| | Select by Location | ObjectVector | | ObjectVector |
| | Spatial Join*** | ObjectVector | VectorTessellation | Lattice |
| | Raster to Vector | ExistenceRaster | | ObjectVector |
| | Feature to Raster | ObjectVector | | ExistenceRaster |
| | Merge | ExistenceVector | | ObjectVector |
| | Euclidean Distance | BoundedPhen | | RatioA⊓FieldRaster |
| | Focal Density | BoundedPhen | | RatioA⊓FieldRaster |

Table 1: Signatures for major types of GIS operations. Some operations are overloaded and some have the same signature. Some require additional function parameters, which can change the type of input and output attributes: *with Mean, Variety, Sum function **with Mean, Majority, Sum function, ***with $Join\_one\_to\_one$ parameter and Count, Mean, Sum functions. In effect, this is only a coarse overview of operational types described in our annotation file.

exclusively on field representations. Yet, *zonal operations* have a different role in the computational diagram, namely as one of the few possibilities to generate lattice data from field representations. We can distinguish zonal functions according to their required scale level, e.g., *Zonal Mean* requires interval scales, and *Zonal Median* requires ordinal scales, while *Zonal Variety*[35] requires only nominal scales. As rasters are tessellations, the geometric layer type is hardly affected by this operation[36], however the underlying semantics change considerably from field rasters to lattices. Everyone who practically works with rasters learns early on that *Reclassify* is an indispensable tool. Our signatures (Fig. 8) show a possible reason for this: it enables us to turn a field raster into an existence raster, by selecting and "highlighting" a specific field quality value using a Boolean value. For example, to turn a landuse raster into a one telling us "here is forest" and "here is no forest", which can then be used in further analysis. *Select by Location* allows us to do the same thing with vector field representations. *Create Contours* takes an interval scaled field raster and generates a contour map. Every contour map is a coverage, and every coverage can be turned into a

[35]Computes the number of unique values in each zone.

[36]As a matter of fact, there are zonal map algebra implementations for both vector tessellation as well as raster outputs, compare "Zonal statistics" and "Zonal statistics as Table".

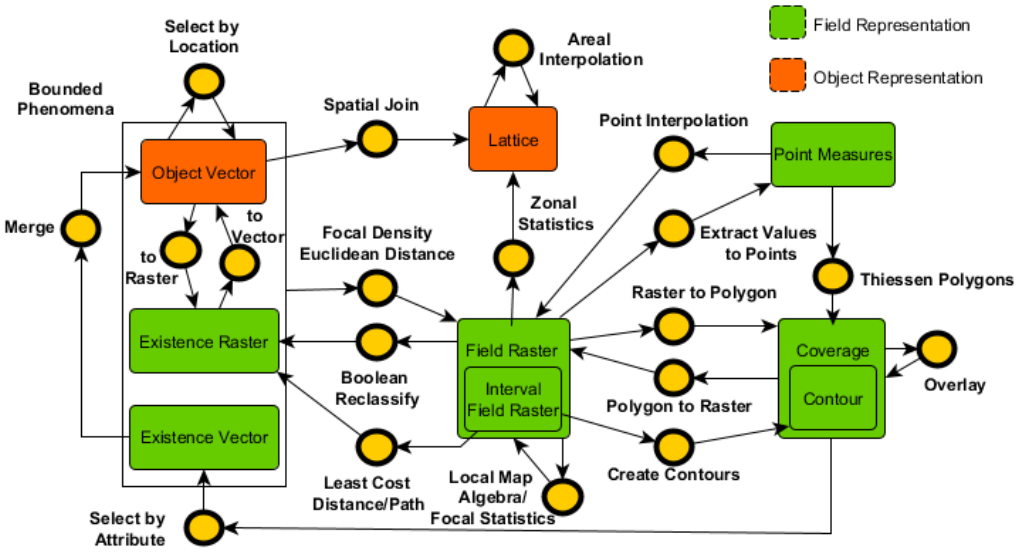



Figure 8: Computational diagram of types of geoanalytic operations applicable to semantic data types. Boxes denote semantic data types, and bold circles denote operation types.

field raster simply by *Polygon to Raster*, because coverages are self-similar. The "converse" operation, *Raster to Polygon*[37], selects subsets of cells with the same attribute value to generate polygons, making it the prototypical operation for generating coverages. Because of self-similarity, coverages can be combined in a straightforward manner via *Overlay* (including Identity, Intersect and Union), since the newly generated polygons can simply inherit the attribute values of their source layers. Our Overlay signature is therefore exclusively on coverages, though ordinary GIS would allow analysts to unjustifiably apply them to all kinds of vector data sets[38]. A different set of operations is used to deal with field vectors. *Point Interpolation* (such as Kriging or Inverse Distance Weighting (IDW)) exclusively works with Point measures to generate a field raster. The inverse operation is *Extract values to points*. Similarly, interpolating with *Thiessen polygons* requires that the input points represent a field, such as liters of rainfall. *Least Cost Distance/Path*[39] takes an interval scaled field raster which represents a cost surface, and computes an existence raster that denotes the location of the least cost path.

### 5.2.2 Operations on Object representations

Combining layers is entirely different for lattices, which are tessellated object representations. The default operation for combining and turning lattices into each other is *Areal*

[37]Using the "raster_field" parameter to indicate the attribute.

[38]Note that intersecting the geometries of objects is still possible with other operations. Remember that the listed operations are on spatial attributes, not on geometry values.

[39]We treat here both, Least Cost Distance and Least Cost Path, as a single operation, though they appear as separate (but chained) operations in ArcGIS.

*Interpolation* (including simple area weighted averages as well as Kriging based approaches and others) [20]. Simple overlay is inappropriate here due to missing self-similarity[40].

A more general way of representing objects is an object vector, which misses the tessellation requirement. We argue that this is the data type for which *location queries* (*Select by location*) are meaningful, due to the fact that *spatial relations* (such as "overlaps", "contains", "within distance"), which are used in these queries, require countable, bounded entities which can overlap. Due to the very same reason, a *Spatial Join* is restricted to object vectors, effectively turning them into lattices when used together with a vector tessellation. For example, by counting the number of objects within each region or by summarizing their qualities.

The *Raster to Point/Line* and *Feature To Raster* toolsets in ArcGIS[41] can be used to translate from existence rasters to object vectors and back for all geometric types. Note that their signature therefore differs from "Polygon to Raster" discussed above, which operates exclusively on field representations.

Operations for analysing bounded phenomena are *Euclidean Distance* and *Focal Density*. These methods use boundaries to compute a field of distances to the nearest/densities of neighboring phenomena. Focal Density subsumes Kernel Density estimation, simple Point Density, as well as Areal Density (computing the percentage of areal coverage in a moving window around each cell).

# 6 Workflow synthesis and validation

In order to test our ontology, we translate the competency questions of Sect. 2 into queries of a logic called Semantic Linear Time Logic (SLTL) [46,61], an extension of the well-known Linear Time Logic (LTL). This logic can be used to specify and synthesize programs in terms of sequences of typed operation applications over data. For workflow synthesis we are using the APE framework [40][42]. Since data retrieval is beyond the scope of this study, two simplifications were applied to workflow construction: 1) Only relevant data sources are available at the start; 2) At any moment during workflow construction, relevant input data can be unambiguously identified by its type. We tested our ontology by checking whether it generates valid workflows for each question and corresponding data source, with the capacity of answering this question. Quality was assessed by expert judgment, based on exploring examples of answer workflows[43]. Note that a more systematic test (including IR precision gains and broader scenarios) is planned in the near future.

## 6.1 Semantic Linear Time Logic

The domain of discourse of SLTL are computational pipelines (workflows), which we model as alternating sequences (paths) of sets of concrete data types $t_i \in T_c$ and concrete

[40]This is the same for raster based lattices, which is why raster *resampling* is similar to vector-based areal interpolation [54]. In our scheme, we subsume both under Areal Interpolation. We are aware that in practice, such raster lattices may often be re-interpreted as field rasters instead.

[41]Used with a boolean "raster_field" and "value_field" parameter. Unfortunately, boolean values are only implicit in ArcGIS, encoded, e.g., in "NoData" values.

[42]https://github.com/sanctuuary/APE

[43]The synthesis software together with all configurations, input and output files is available under https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT.

tools $s_i \in S_c$, with length *k+1*, $k \in \mathbb{N}_0$:

$$p = (t_0, s_1, t_1, s_2, t_2, ...s_k, t_k) \tag{1}$$

*Concrete* types/tools refer to specific data types and tools/operations from the domain, i.e., each concrete type represents an actual data instance (e.g., FieldRaster), while a concrete tool represents an executable operation (e.g., EuclideanDistance(Vector)). These concrete elements are depicted as leaves in a larger semantic hierarchy of types $T = T_c \cup T_a$ and tools $S = S_c \cup S_a$, where $T_a$ and $S_a$ represent abstractions over the concrete elements.

**Auxiliary definition 1.** *Let $T = T_c \cup T_a$ be a semantic hierarchy of types, where $T_c$ represents the leaves of the hierarchy and $T_a$ represents the non-leaf elements in the hierarchy. Similarly, let $S = S_c \cup S_a$ be a semantic hierarchy of tools. Then $\forall t_i \in T_c$ and $\forall s_i \in S_c$, the sets of the corresponding abstract types and tools are defined as follows:*

$$TTax(t_i) = \{t_i\} \cup \{t \in T_c | \, t_i \textit{ subsumed by t in T}\} \ ,$$

$$STax(s_i) = \{s_i\} \cup \{s \in T_c | \, s_i \textit{ subsumed by s in S}\}$$

**Auxiliary definition 2.** *Let $p$ be a path of length $k+1$, where $k \in N_0$ and let $i \in [1, k]$ be an index. A subpath $p_i$ represents a part of the path $p$ obtained by removing the first $i$ pairs of types and tools (2n elements in total) and is defined as follows:*

$$p_i = (t_i, s_{i+1}, ...s_k, t_k)$$

*with a special cases of $p_i = p$ when $i = 0$ and $p_i = (t_k)$ when $i = k$.*

We now introduce formulas in SLTL:

**Auxiliary definition 3.** *Formulas of SLTL are described by the following BNF:*

$$\phi \ ::= \ true \mid t \mid \neg\, \phi \mid \phi \wedge \phi \mid \ < s > \ \phi \mid G\, \phi \mid \phi\, U\, \phi$$

*where $t \in T$ and $s \in S$.*

For a path to satisfy such a formula, the alternating sequence of types and tools has to satisfy the following definition:

**Auxiliary definition 4.** *Whether a path p satisfies an SLTL formula $\phi$ (written as $p_0 \vdash \phi$) is defined inductively as:*

$$p_n \vdash true \quad \textit{is satisfied by every path (true applies to any path)} \tag{2}$$

$$p_n \vdash t \quad \textit{iff } t \in TTax(t_n) \textit{ (t applies to the first type in the sequence)} \tag{3}$$

$$p_n \vdash \neg\phi \quad \textit{iff not } p_n \vdash \phi \textit{ ($\phi$ does not hold)} \tag{4}$$

$$p_n \vdash \phi_1 \wedge \phi_2 \quad \textit{iff } p_n \vdash \phi_1 \textit{ and } p_n \vdash \phi_2 \textit{ ($\phi_1$ and $\phi_2$ hold)} \tag{5}$$

$$p_n \vdash < s > \phi \quad \textit{iff } s \in STax(s_{n+1}) \textit{ and } p_{n+1} \vdash \phi \textit{ (s is the next tool in the sequence and $\phi$ holds in the following subpath)} \tag{6}$$

$$p_n \vdash G\phi \quad \textit{iff } p_n \vdash \phi \wedge p_{n+1} \vdash G\phi \textit{ ($\phi$ holds generally, i.e., in each subpath of the path)} \tag{7}$$

$$p_n \vdash \phi_1 U \phi_2 \quad \textit{iff } p_n \vdash \phi_2 \textit{ or } p_n \vdash \phi_a \wedge p_{n+1} \vdash \phi_1 U \phi_2 \textit{ ($\phi_1$ holds until $\phi_2$)} \tag{8}$$

Notice that operators such as $\vee$ (logical or) and $F\phi$ (logical finally) can be easily computed using the presented operators, as follows: $p_n \vdash \phi_1 \vee \phi_2$ **iff** $p_n \vdash \neg(\phi_1 \wedge \phi_2)$ and $p_n \vdash F\phi$ **iff** $p_n \vdash true\, U \phi$. More useful operators can be defined over chained operations, such as *out* and *in*:

**Auxiliary definition 5.** *The workflow produces a given output data type t if the last type in the sequence is of type t. Formally, we have to express that as saying that every subpath of the path p contains the type t (including the subpath $p_k$ where k is the length of the path):*

$$p_0 \vdash \textit{out } t \textit{ iff } p_0 \vdash G(Ft)$$

**Auxiliary definition 6.** *The workflow includes a set of given input data types $t^1, ..., t^n$ if the first types in the sequence are the types $t^1, t^2, ..., t^n$. Formally, we have to express that as saying that the first type in the path $p_0$ satisfies the types $t^1, ..., t^n$:*

$$p_0 \vdash \textit{in } t^1,...,t^n \textit{ iff } p \vdash t^1 \wedge ... \wedge t^n$$

Additionally, we would like to have operators expressing a need for a certain data type (*gen*) or a tool (*use*) to be used in the sequence. For that purpose we are defining two additional operators.

**Auxiliary definition 7.** *A workflow generates a data type t when a type element in the path eventually represents the type t:*

$$p_0 \vdash \textit{gen } t := p_n \vdash Ft$$

**Auxiliary definition 8.** *A workflow uses a tool s when one of the steps in the path represents the tool s:*

$$p_n \vdash \textit{use } s := p_n \vdash F(< s > true)$$

Although workflows are represented as sequences, tools can take as input not only the direct predecessor type. Rather, the set of available types is expanded in each step, and therefore types generated previously can be reused further down the path. This means we can rewrite sequences as directed acyclic graphs (DAG), which is the way how GIS workflows are usually conceptualized. The *synthesis algorithm* [40, 61] used in our examples generates all compositions that satisfy the given specification with a given maximal length, ordered by length. The specification consists of an SLTL formula, together with the synthesis universe $S, T, T_{sig}$, where $T = T_c \cup T_a$, $S = S_c \cup S_a$ and $T_{sig}$ contains the set of tool signatures $(t^{in}, s_c, t^{out})$ denoting the input and output types of each concrete tool, where $t_c \in T_c$.

The workflow synthesis sketched above integrates with *reasoning* on the CCD ontology and the tool annotations in two phases:

1. In a first phase, we applied OWL 2 RL[44] and RDFS reasoning to expand the ontology with inferred subsumption statements for defined classed. We then extracted the subsumption graph corresponding to $T$. Together with the tool annotation file $T_{sig}$, it was fed into the synthesis process.
2. In the second reasoning phase, the synthesis process starts from a given SLTL formula (the query specification) and searches the space of all possible tool (in $T_{sig}$) sequences in order to satisfy the formula. The tools are semantically extended using the ontology over inputs and outputs. In order to exclude ambiguous classifications and to speed up search, we assume that the subclass relation is jointly exhaustive, and leaf classes are pairwise disjoint. This assumption can be satisfied for CCD types by introducing proper subclasses as leafs and by type combinations in a preprocessing step. We then can use implications of the form

$$A \Rightarrow A1 \vee A2$$

where A is a superclass of A1 and A2, to find A1 and A2 as possible inputs for A.

In this way, OWL subsumption reasoning expands the space of sensible operations for workflow synthesis reasoning. In practice, this means that we can specify a constraint over an abstract type, for example by choosing some field (FieldQ) as a goal. This constraint would be satisfied by a tool that generates, e.g., a field raster. Using the workflow synthesis algorithm, we synthesized up to 50 possible workflows (ordered by size) answering Questions 1,2,3,4 of a maximum length of 4 tool applications.

## 6.2 Geo-analytical queries in SLTL

We can now rephrase our competency questions in terms of SLTL formulas exploiting the CCD ontology and the input data types for each question (cf. complete specification files in footnotes):

**Question 1.** *What is the number of sport facilities (ObjectPoint) of each PC4 area (VectorTessellation) in Amsterdam?*
***in*** *ccd:ObjectPoint, ccd:VectorTessellation* $\wedge$ ***out*** *ccd:CountALattice*

[44] https://www.w3.org/TR/owl2-profiles

[45] Note that in this definition, we make use of a measurement level (a count lattice) to specify what we need as output. Alternatively, one could also leave the measurement level unspecified. We tried out both versions.

**Question 2.** *What is the rate of elderly people of each PC4 area in Amsterdam?*
***in*** *ccd:Lattice, ccd:VectorTessellation* $\wedge$ ***out*** *ccd:Lattice*

[46] In this specification, we ask for translating a lattice of rates on neighborhood level into one on PC4 level.

**Question 3.** *What is the average distance to the nearest green of each PC4 area in Amsterdam?*
***in*** *ccd:CoverageNominalA, ccd:VectorTessellation* $\wedge$ ***out*** *ccd:Lattice* $\wedge$ ***use*** *tool:Distance*

[47] [48] In this specification, we ask about the accessibility of green areas, which is translated in terms of a workflow from a nominal coverage to a lattice which makes use of some distance function. The *use* constraint enforces workflows to apply some distance function[49].

**Question 4.** *What is the average noise level of each PC4 area in Amsterdam?*
***in*** *ccd:Contour, ccd:VectorTessellation* $\wedge$ ***out*** *ccd:Lattice* $\wedge$ ***use*** *tool:ZonalStatistics*

[50] [51] In this specification, we ask for a lattice of average noise levels, derived from a contour map. The term "average" is translated here in terms of the Zonal Statistics method (which can make use of any aggregation method dependent on the measurement level). Of course, other translations may be possible.

Note that in order to improve the precision of workflow construction, further constraints may be added to each translated question, concerning, e.g., the types of functionality or the order of tool application. However, since our validation is preliminary, precision (making sure workflows have high quality) is less relevant than recall (useful workflows are captured by the ontology). We therefore leave more sophisticated translations and evaluations to future work.

## 6.3 Constructing a liveability atlas of Amsterdam

In this section we illustrate how our data types can be exploited together with the operation signatures in Table 1 to give meaningful answers to questions specified in the last section. The goal is a liveability atlas on the level of PC4 regions (see example Fig. 3), which is of type *Lattice*. The synthesized workflows are sequences of the form *type, type,... - tool - type* as introduced above, and we indicate parametrized versions of tools by writing *[toolname]([parameter])*. Once introduced, types can be reused along the pipeline. We picked in

[45] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/ape_sports.configuration

[46] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/ape_cbs.configuration

[47] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/ape_bbg.configuration

[48] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/constraints_bbg.csv

[49] We use a tool type to illustrate the possibility for different kinds of spatial distance tools that might be used here, e.g., Euclidean versus network distance.

[50] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/ape_noise.configuration

[51] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/constraints_noise.csv

the following only representative examples of such workflows, both of better and worse quality for answering the given question. The complete set of answer workflows is linked in a footnote under each question.

**Number of sport facilities of PC4 areas in Amsterdam** [52]

**Workflow 1.** *NominalA⊓ObjectPoint, TessellationA⊓VectorA - SpatialJoin(Count) - CountA⊓Lattice*

This occurred not only as the shortest possible workflow in the set, it also corresponds to the shortest meaningful answer: using a spatial join with the count parameter on a nominal ObjectPoint and a vector tessellation, to produce a lattice of counts of these objects. When relaxing the goal type to Lattice, the synthesis algorithm produced also two more sophisticated ways to reach an equally valid result:

**Workflow 2.** *NominalA⊓ObjectPoint, TessellationA⊓VectorA - EuclideanDistance(Vector) - FieldRaster⊓RatioA - ZonalStatistics(Mean) - Lattice⊓RatioA*

In this workflow, we do not count the number of sport facilities, but rather compute a mean distance to the nearest facility for each PC4 area, using the Euclidean Distance tool with vector data. This gives a measure of accessibility of sport facilities.

**Workflow 3.** *NominalA⊓ObjectPoint, TessellationA⊓VectorA - FocalStatistics(Vector) FieldRaster⊓RatioA - ZonalStatistics(Mean) - Lattice⊓RatioA*

In this workflow we apply focal statistics to first compute the density of facilities, which is then averaged into PC4 areas.

Since our specification of the question in the last section is still very coarse, also many less useful and redundant workflows for the given task are produced. For example:

**Workflow 4.** *NominalA⊓ObjectPoint, TessellationA⊓VectorA - SpatialJoin(Count) - CountA⊓Lattice - ArealInterpolation(Rate) - CountA⊓Lattice*

In this workflow, we add an unnecessary interpolation step that turns the lattice into an equivalent lattice. While this workflow is not wrong, it would not be considered for practical reasons.

**Rate of elderly people of PC4 areas in Amsterdam** [53] This question produced only a single workflow result (ignoring redundant mutations):

**Workflow 5.** *Lattice, TessellationA⊓VectorA - ArealInterpolation(Average) - Lattice*

The workflow suggests using an areal interpolation technique to transform rates of elderly from CBS neighborhoods to PC4 regions.

[52] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/sat_solutions_sports.txt

[53] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/sat_solutions_cbs.txt

**Distance to the nearest green of PC4 areas in Amsterdam** [54]

**Workflow 6.** *Coverage⊓NominalA, TessellationA⊓VectorA - VectorToRaster - FieldRaster⊓NominalA - BooleanReclassify - BooleanA⊓ExistenceRaster - EuclideanDistance(Raster) - FieldRaster⊓RatioA - ZonalStatistics(Mean) - Lattice⊓RatioA*

In this workflow, the BBG landuse coverage is first turned into a nominal field raster, which is then reclassified (selecting those landuse values which correspond to "green") into an existence raster, before computing a distance field to "green", which is then aggregated with zonal statistics into a lattice. The workflow set contains also another interesting and equally valid result:

**Workflow 7.** *Coverage⊓NominalA, TessellationA⊓VectorA - SelectbyAttribute - ExistenceVector - Merge - CountA⊓ObjectVector - EuclideanDistance(Vector) - FieldRaster⊓RatioA - ZonalStatistics(Mean) - Lattice⊓RatioA*

Here, the green areas are not selected by raster reclassification, but rather by selection of vector attributes, which are then merged into a single object, based on which distances are computed and aggregated into the lattice.

**Average noise level of PC4 areas in Amsterdam** [55] We obtained a meaningful answer based on the following workflow:

**Workflow 8.** *Contour, TessellationA⊓VectorA - VectorToRaster - FieldRaster⊓OrdinalA - ZonalStatistics(Median) - Lattice⊓OrdinalA*

In this workflow, an ordinal lattice (of noise levels) is produced by first generating an ordinal field raster, and then computing the median of the noise levels within each PC4 area. This is a valid solution. However, in addition, we also got many that were less useful for the given purpose:

**Workflow 9.** *Contour, TessellationA⊓VectorA - VectorToRaster - FieldRaster⊓OrdinalA - FocalStatistics(Variety) - CountA⊓FieldRaster - ZonalStatistics(Mean) - Lattice⊓RatioA*

In this workflow, we compute the variety (the number of different noise values) in a focal window around each raster cell, which is aggregated into PC4 areas. This workflow does not measure a central value, but rather the spread of noise values.

**Workflow 10.** *Contour, TessellationA⊓VectorA - VectorToRaster - FieldRaster⊓OrdinalA - BooleanReclassify - BooleanA⊓ExistenceRaster - ZonalStatistics(Majority) - BooleanA⊓Lattice*

In this workflow, we pick out a particular noise level in terms of an existence raster and compute whether the majority of cells within a PC4 area has that level or not. While this workflow is meaningful, it is less useful because the answer it gives is unnecessarily coarse, and thus would not be selected by a GIS analyst.

The workflows discussed above are still rather abstract, because they are on the type level. For each of the four questions, Fig. 9 summarizes the most useful answer in terms of concrete data sources. These paths in fact correspond to GIS workflows that a pro-

[54] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/sat_solutions_bbg.txt

[55] https://github.com/simonscheider/SemanticPipelines/blob/master/SatSolutionsCCDT/sat_solutions_noise.txt

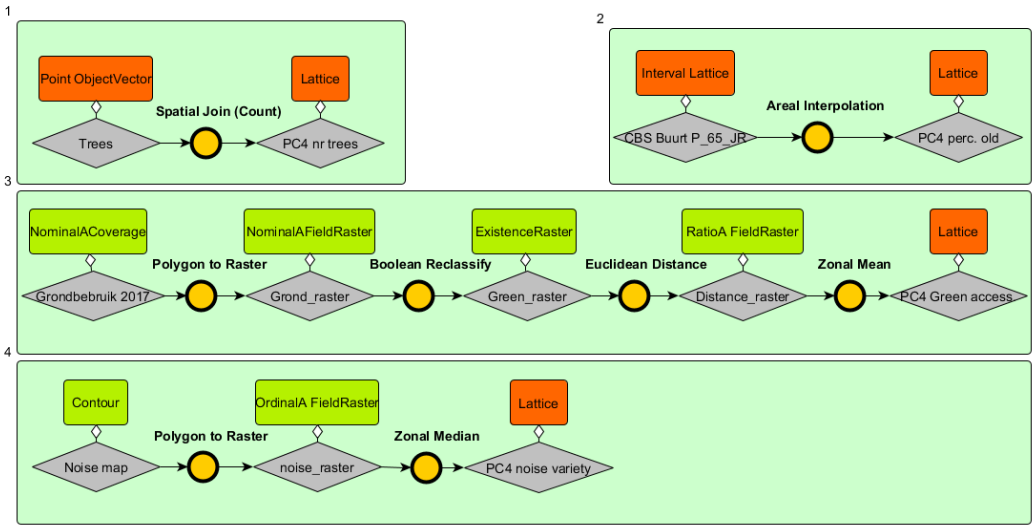



Figure 9: Meaningful workflows suggested by our computational signatures. Red and green boxes denote object and field representations, respectively. Grey diamonds denote geodata attributes.

fessional GIS analyst might implement in order to generate the atlas: ObjectVectors are aggregated via Spatial Join (Fig. 9.1). Lattices can be converted to lattices only via areal interpolation (Fig. 9.2). The landuse map is a nominal coverage. Turned into a FieldRaster, it becomes nominal, too. We can select one of its classes by reclassification to generate an ExistenceRaster, which can then be used to generate a distance based raster (which is ratio scaled). The latter can be aggregated via Zonal Mean into a lattice (Fig. 9.3). Since a noise map is a contour map, it is bound to be ordinal, so the generated field raster is ordinal, too, and thus can only be aggregated to a lattice via a Zonal Median function (Fig. 9.4).

Finally, our tests demonstrated that CCD type signatures at the same time prevent loads of meaningless artifacts of bad GIS design, which result as soon as we remove the core concept data types and only keep the standard types vector and raster. For example:

**Workflow 11.** *Sport facilities (PointVector) - PointInterpolation - Raster*

is meaningless because sport facilities are objects, not fields;

**Workflow 12.** *Grondgebruik 2017 (RegionVector) - SpatialJoin(Count) - RegionVector*

is meaningless because landuse coverages represent fields, so areas do not have an identity and thus are not countable;

**Workflow 13.** *CBS Buurt (RegionVector) - PolygontoRaster - Raster*

is meaningless because CBS data are lattices, and thus overlay results in a Modifiable area unit problem (MAUP) error [22];

**Workflow 14.** *Noise Map 2017 (RegionVector) - ArealInterpolation - RegionVector*

is meaningless since noise map is a field, not a lattice, and thus requires point interpolation.

# 7 Conclusion, discussion and future work

In this article, we systematically investigated the different ways how core concepts of spatial information can be represented by geodata, as a basis for a type system that helps answering geo-analytical questions by constructing appropriate workflows. We have suggested a lightweight ontology pattern in OWL, including geometric layer types, measurement levels and core concepts of spatial information. The CCD ontology can not only be used to specify geo-analytical questions, but also to reason with data sources across data formats and to add type signatures to GIS operations in order to constrain meaningful applications and to automate the construction of answers. The basic idea is that core concept interpretations of data sources can be represented in direct and more indirect ways in terms of different geometric layer types with different measurement scale levels. We have introduced a DL formalization of such types as combinations of basic classes, which can be used to reason about geodata and GIS operations. Furthermore, using a scenario for computing a liveability index for Amsterdam, we demonstrated that our type signatures are effective in synthesizing useful answers and in preventing computational nonsense. Our test shows that not only valid workflows are produced, but also that various equally valid answers can be discovered this way.

However, our study also illustrates a number of challenges which need to be met in order to turn this into a useful method of geo-analytical question-answering. One challenge is the *problem of ambiguity* of core concept interpretation. For example, analysts frequently re-interpret data, "ignoring" their properties in order to simplify analysis. Events are re-interpreted as objects, lattices may be re-interpreted as field rasters to simplify overlay, and rasters may be re-interpreted as regularized point measures using the cell center points. In the CCD annotation of data sources, we might in the future incorporate this possibility with systematic (probabilistic?) re-annotations. Another problem is *automating annotations*. How can CCD annotations be scaled up across large amounts of data sources and tool sources on the web, such as the different GIS software tools [4]? This problem might be partially solved using data mining, as done in [56], or might draw on existing metada, as given by the OGC. However, we expect that a substantial amount of manual annotations is unavoidable. For this purpose, we are planning extensive data source annotation studies. Based on this, future work should investigate the role of the CCD ontology in mitigating *interoperability* problems [7], based on matching GIS tools to annotated data sources.

The proposed type system should be extended by the core concepts network and event, allowing us to capture *accessibility* and *trajectory* as concepts within data sources and workflows. Also, a genuine ontological study of the formal properties of core concepts in a more expressive logic is still lacking. Highlighting how the CCD ontology deals with temporal information is an important part of this. In case time is a constraint on data, one might use temporal functions that are separate from any ontology. In case we want to describe temporal GIS functions (which have time as input or output), temporal values of core concepts might be handled with OWL Time[56]. Beyond this, one might take the stance that processes should be considered a separate category. However, if all core concepts come with some temporal dimension, we wonder to what extent this covers the role of processes. Furthermore, it is apparent that our current type system does not distinguish different *kinds of transformations* between the same core concept data types. To define GIS functionality beyond this level, we need a *transformation language* that generalizes tools similar to the data

[56]https://www.w3.org/TR/owl-time/

types in terms of their behavior. To this end, we are currently working on a transformation algebra which is based on core concepts.

In this study, we tested GIS workflow synthesis only on a limited basis. In the future, we plan to extend our test by measuring the precision gain of the approach with respect to a benchmark of commonly used geodata types, in terms of meaningful vs. non-meaningful, redundant, or erroneous workflows. For this purpose, we work on extending the test scenarios to cover more diverse workflow queries. Regarding reasoning, we plan to use SLTL workflow reasoning to further constrain workflows in terms of sequences of operations for retaining data quality [1, 42]. In this way, we can prevent information loss in terms of (spatial and semantic) resolution. Furthermore, we are working on a method to reason with classes projected to separable semantic dimensions, which makes the approach independent from ad-hoc class combinations.

Finally, we believe that core concepts and their transformations can play a central role in realizing question-based interactions with GIS [54, 63], since type transformations can express analytical goals, which capture precisely the intent of corresponding questions. For example, the task of aggregating green space into statistical units corresponds to a question about aggregated qualities of these units, e.g., "how accessible is green space within Amsterdam?". For this purpose, it is necessary to analyse the structure of geo-analytical questions in greater depth. In the future, a geo-analytical grammar might be devised that helps constructing and translating such questions into queries over workflows.

## Acknowledgments

This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 803498). We would like to thank the reviewers as well as Enkhbold Nyamsuren for their support in the revision process. Furthermore, the article owes a lot to the comments by Frans Rip, Werner Kuhn, Sara Lafia and the research group at UCSB.

## References

[1] AL-AREQI, S., LAMPRECHT, A.-L., AND MARGARIA, T. Constraints-driven automatic geospatial service composition: Workflows for the analysis of sea-level rise impacts. In *International Conference on Computational Science and Its Applications* (2016), Springer, pp. 134–150. doi:10.1007/978-3-319-42111-7_12.

[2] ALBRECHT, J. Universal analytical GIS operations: A task-oriented systematization of data structure-independent GIS functionality. *Geographic information research: Transatlantic perspectives* (1998), 577–591.

[3] ALLEN, J. F. Maintaining knowledge about temporal intervals. In *Readings in qualitative reasoning about physical systems*. Elsevier, 1990, pp. 361–372. doi:10.1145/182.358434.

[4] BALLATORE, A., SCHEIDER, S., AND LEMMENS, R. Patterns of consumption and connectedness in GIS web sources. In *The Annual International Conference on Geographic Information Science* (2018), Springer, pp. 129–148. doi:10.1007/978-3-319-78208-9_7.

[5] BAO, J., DUAN, N., ZHOU, M., AND ZHAO, T. Knowledge-based question answering as machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2014), vol. 1, pp. 967–976.

[6] BERNSTEIN, A., HENDLER, J., AND NOY, N. A new look at the semantic web. *Communications of the ACM 59*, 9 (2016), 35–37. doi:10.1145/2890489.

[7] BISHR, Y. Overcoming the semantic and other barriers to GIS interoperability. *International journal of geographical information science 12*, 4 (1998), 299–314. doi:10.1080/136588198241806.

[8] BOOTS, B., OKABE, A., AND SUGIHARA, K. Spatial tessellations. *Geographical information systems 1* (1999), 503–526.

[9] BURROUGH, P. A., AND FRANK, A. U. Concepts and paradigms in spatial information: are current geographical information systems truly generic? *International Journal of Geographical Information Systems 9*, 2 (1995), 101–116. doi:10.1080/02693799508902028.

[10] BURROUGH, P. A., AND FRANK, A. U. *Geographic objects with indeterminate boundaries*, vol. 2. CRC Press, 1996.

[11] CAI, G., WANG, H., MACEACHREN, A. M., AND FUHRMANN, S. Natural conversational interfaces to geospatial databases. *Transactions in GIS 9*, 2 (2005), 199–221. doi:10.1111/j.1467-9671.2005.00213.x.

[12] CAMARA, G., EGENHOFER, M. J., FERREIRA, K., ANDRADE, P., QUEIROZ, G., SANCHEZ, A., JONES, J., AND VINHAS, L. Fields as a generic data type for big spatial data. In *International Conference on Geographic Information Science* (2014), Springer, pp. 159–172. doi:10.1007/978-3-319-11593-1_11.

[13] CÂMARA, G., FREITAS, U., AND CASANOVA, M. A. Fields and objects algebras for GIS operations. In *Proceedings of III Brazilian Symposium on GIS* (1995), pp. 407–424.

[14] CHEN, W. *Developing a Framework for Geographic Question Answering Systems Using GIS, Natural Language Processing, Machine Learning, and Ontologies*. PhD thesis, Ohio State University, 2014.

[15] CHRISMAN, N. *Exploring Geographic Information Systems, 2nd Edition*. Wiley, 2002, ch. Reference systems for measurement, pp. 15–35.

[16] COHN, A. G., BENNETT, B., GOODAY, J., AND GOTTS, N. M. Representing and reasoning with qualitative spatial relations about regions. In *Spatial and temporal reasoning*. Springer, 1997, pp. 97–134. doi:10.1007/978-0-585-28322-7_4.

[17] COUCLELIS, H. People manipulate objects (but cultivate fields): beyond the raster-vector debate in GIS. In *Theories and methods of spatio-temporal reasoning in geographic space*. Springer, 1992, pp. 65–77. doi:10.1007/3-540-55966-3_3.

[18] COVA, T. J., AND GOODCHILD, M. F. Extending geographical representation to include fields of spatial objects. *International Journal of geographical information science 16*, 6 (2002), 509–532. doi:10.1080/13658810210137040.

[19] CURBERA, F., DUFTLER, M., KHALAF, R., NAGY, W., MUKHI, N., AND WEERAWARANA, S. Unraveling the web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet computing 6*, 2 (2002), 86–93. doi:10.1109/4236.991449.

[20] DE SMITH, M. J., GOODCHILD, M. F., AND LONGLEY, P. *Geospatial analysis: a comprehensive guide to principles, techniques and software tools*. Troubador Publishing Ltd, 2007.

[21] EINSTEIN, A. On the method of theoretical physics. *Philosophy of science 1*, 2 (1934), 163–169. doi:10.1086/286316.

[22] FOTHERINGHAM, A. S., AND WONG, D. W. The modifiable areal unit problem in multivariate statistical analysis. *Environment and planning A 23*, 7 (1991), 1025–1044. doi:10.1068/a231025.

[23] GAHEGAN, M. Specifying the transformations within and between geographic data models. *Transactions in GIS 1*, 2 (1996), 137–152. doi:10.1111/j.1467-9671.1996.tb00040.x.

[24] GALTON, A. Fields and objects in space, time, and space-time. *Spatial cognition and computation 4*, 1 (2004), 39–68. doi:10.1207/s15427633scc0401_4.

[25] GALTON, A. Outline of a formal theory of processes and events, and why GIScience needs one. In *International Conference on Spatial Information Theory* (2015), Springer, pp. 3–22. doi:10.1007/978-3-319-23374-1_1.

[26] GANGEMI, A. Ontology design patterns for semantic web content. In *International semantic web conference* (2005), Springer, pp. 262–276. doi:10.1007/11574620_21.

[27] GANGEMI, A., GUARINO, N., MASOLO, C., OLTRAMARI, A., AND SCHNEIDER, L. Sweetening ontologies with DOLCE. In *International Conference on Knowledge Engineering and Knowledge Management* (2002), Springer, pp. 166–181. doi:10.1007/3-540-45810-7_18.

[28] GIL, Y., DEELMAN, E., ELLISMAN, M., FAHRINGER, T., FOX, G., GANNON, D., GOBLE, C., LIVNY, M., MOREAU, L., AND MYERS, J. Examining the challenges of scientific workflows. *Computer 40*, 12 (2007), 24–32. doi:10.1109/MC.2007.421.

[29] GOODCHILD, M. F. A spatial analytical perspective on geographical information systems. *International Journal of Geographical Information Systems 1*, 4 (1987), 327–334. doi:10.1080/02693798708927820.

[30] GUIZZARDI, G. On the representation of quantities and their parts in conceptual modeling. In *FOIS* (2010), pp. 103–116. doi:10.5555/1804715.1804728.

[31] GÜTING, R. H. Geo-relational algebra: A model and query language for geometric database systems. In *International Conference on Extending Database Technology* (1988), Springer, pp. 506–527. doi:10.1007/3-540-19074-0_70.

[32] GÜTING, R. H., BEHR, T., DÜNTGEN, C., ET AL. SECONDO: A platform for moving objects database research and for publishing and integrating research implementations. *IEEE Data Eng. Bull. 33*, 2 (2010), 56–63.

[33] GÜTING, R. H., BÖHLEN, M. H., ERWIG, M., JENSEN, C. S., LORENTZOS, N. A., SCHNEIDER, M., AND VAZIRGIANNIS, M. A foundation for representing and querying moving objects. *ACM Transactions on Database Systems (TODS) 25*, 1 (2000), 1–42. doi:10.1145/352958.352963.

[34] HAHMANN, T., AND USERY, E. L. What is in a contour map? In *International Conference on Spatial Information Theory* (2015), Springer, pp. 375–399. doi:10.1007/978-3-319-23374-1_18.

[35] HEINEMAN, G. T., AND COUNCILL, W. T. Component-based software engineering. Putting the pieces together. *Addison-wesley* (2001), 5. doi:10.1126/stke.2001.103.tw372.

[36] HINSEN, K. Computational science: shifting the focus from tools to models. *F1000Research 3* (2014). doi:10.12688/f1000research.3978.2.

[37] HITZLER, P., GANGEMI, A., AND JANOWICZ, K. *Ontology engineering with ontology design patterns: Foundations and applications*, vol. 25. IOS Press, 2016.

[38] HOFER, B., MÄS, S., BRAUNER, J., AND BERNARD, L. Towards a knowledge base to support geoprocessing workflow development. *International Journal of Geographical Information Science 31*, 4 (2017), 694–716. doi:10.1080/13658816.2016.1227441.

[39] JANOWICZ, K., VAN HARMELEN, F., HENDLER, J. A., AND HITZLER, P. Why the data train needs semantic rails. *AI Magazine* (2014). doi:10.1609/aimag.v36i1.2560.

[40] KASALICA, V., AND LAMPRECHT, A.-L. Workflow Discovery with Semantic Constraints: A SAT-Based Implementation. Under review.

[41] KASALICA, V., AND LAMPRECHT, A.-L. Automated composition of scientific workflows: A case study on geographic data manipulation. In *2018 IEEE 14th International Conference on e-Science (e-Science)* (2018), IEEE, pp. 362–363. doi:10.1109/eScience.2018.00099.

[42] KASALICA, V., AND LAMPRECHT, A.-L. Workflow discovery through semantic constraints: A geovisualization case study. In *International Conference on Computational Science and Its Applications* (2019), Springer, pp. 473–488. doi:10.1007/978-3-030-24302-9_34.

[43] KRÖTZSCH, M., SIMANCIK, F., AND HORROCKS, I. A description logic primer. *arXiv preprint arXiv:1201.4089* (2012).

[44] KUHN, W. Core concepts of spatial information for transdisciplinary research. *International Journal of Geographical Information Science 26*, 12 (2012), 2267–2276. doi:10.1080/13658816.2012.722637.

[45] KUHN, W., AND BALLATORE, A. Designing a language for spatial computing. In *AGILE 2015*. Springer, 2015, pp. 309–326. doi:10.1007/978-3-319-16787-9_18.

[46] LAMPRECHT, A.-L., NAUJOKAT, S., MARGARIA, T., AND STEFFEN, B. Synthesis-based loose programming. In *2010 Seventh International Conference on the Quality of Information and Communications Technology* (2010), IEEE, pp. 262–267. doi:10.1109/QUATIC.2010.53.

[47] LISKOV, B., AND ZILLES, S. Programming with abstract data types. In *ACM Sigplan Notices* (1974), vol. 9(4), ACM, pp. 50–59. doi:10.1145/942572.807045.

[48] LUDÄSCHER, B., LIN, K., BOWERS, S., JAEGER-FRANK, E., BRODARIC, B., AND BARU, C. Managing scientific data: From data integration to scientific workflows. *Geoinformatics: Data to knowledge 397* (2006), 109. doi:10.1130/2006.2397(08).

[49] LUTZ, M. Ontology-based descriptions for semantic discovery and composition of geoprocessing services. *Geoinformatica 11*, 1 (2007), 1–36. doi:10.1007/s10707-006-7635-9.

[50] O'BRIEN, J., AND GAHEGAN, M. Representing, manipulating and reasoning with geographic semantics within a knowledge framework. In *Developments in Spatial Data Handling* (Berlin, Heidelberg, 2005), Springer Berlin Heidelberg, pp. 585–603. doi:10.1007/3-540-26772-7_44.

[51] REY, S. J. Show me the code: spatial analysis and open source. *Journal of Geographical Systems 11*, 2 (2009), 191–207. doi:10.1007/s10109-009-0086-8.

[52] SAIN, S. R., AND CRESSIE, N. A spatial model for multivariate lattice data. *Journal of Econometrics 140*, 1 (2007), 226–259. doi:10.1016/j.jeconom.2006.09.010.

[53] SCHEIDER, S., AND BALLATORE, A. Semantic typing of linked geoprocessing workflows. *International Journal of Digital Earth 11*, 1 (2018), 113–138. doi:10.1080/17538947.2017.1305457.

[54] SCHEIDER, S., BALLATORE, A., AND LEMMENS, R. Finding and sharing GIS methods based on the questions they answer. *International Journal of Digital Earth* (2018), 1–20. doi:10.1080/17538947.2018.1470688.

[55] SCHEIDER, S., GRÄLER, B., PEBESMA, E., AND STASCH, C. Modeling spatiotemporal information generation. *International Journal of Geographical Information Science 30*, 10 (2016), 1980–2008. doi:10.1080/13658816.2016.1151520.

[56] SCHEIDER, S., AND HUISJES, M. D. Distinguishing extensive and intensive properties for meaningful geocomputation and mapping. *International Journal of Geographical Information Science* (2018), 1–27. doi:10.1080/13658816.2018.1514120.

[57] SCHEIDER, S., AND JANOWICZ, K. Place reference systems. *Applied Ontology 9*, 2 (2014), 97–127. doi:10.3233/AO-140134.

[58] SCHEIDER, S., OSTERMANN, F. O., AND ADAMS, B. Why good data analysts need to be critical synthesists. Determining the role of semantics in data analysis. *Future Generation Computer Systems 72* (2017), 11–22. doi:10.1016/j.future.2017.02.046.

[59] SCHEIDER, S., AND TOMKO, M. Knowing whether spatio-temporal analysis procedures are applicable to datasets. In *FOIS* (2016), pp. 67–80. doi:10.3929/ethz-b-000118709.

[60] STASCH, C., SCHEIDER, S., PEBESMA, E., AND KUHN, W. Meaningful spatial prediction and aggregation. *Environmental modelling & software 51* (2014), 149–165. doi:10.1016/j.envsoft.2013.09.006.

[61] STEFFEN, B., MARGARIA, T., AND FREITAG, B. Module configuration by minimal model construction. Tech. Rep. MIP-9313, Fakultät für Mathematik und Informatik, Universität Passau, 1993.

[62] TOMLIN, C. D. *GIS and cartographic modeling*, vol. 380. Esri Press Redlands, CA, 2013.

[63] VAHEDI, B., KUHN, W., AND BALLATORE, A. Question-based spatial computing—a case study. In *Geospatial Data in a Changing World*. Springer, 2016, pp. 37–50. doi:10.1007/978-3-319-33783-8_3.

[64] YUE, P., DI, L., YANG, W., YU, G., AND ZHAO, P. Semantics-based automatic composition of geospatial web service chains. *Computers & Geosciences 33*, 5 (2007), 649–665. doi:10.1016/j.cageo.2006.09.003.

[65] ZHANG, Z., ZHANG, L., ZHANG, H., HE, W., SUN, Z., CHENG, G., LIU, Q., DAI, X., AND QU, Y. Towards answering geography questions in Gaokao: A hybrid approach. In *China Conference on Knowledge Graph and Semantic Computing* (2018), Springer, pp. 1–13. doi:10.1007/978-981-13-3146-6_1.