International Journal of Geo-Information

MDPI

Article

Topological Access Methods for Spatial and Spatiotemporal Data

Markus Wilhelm Jahn [1] and Patrick Erik Bradley [2,*]

[1] Geodetic Institute, Karlsruhe Institute of Technology (KIT), Englerstr. 7, 76131 Karlsruhe, Germany
[2] Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology (KIT), Englerstr. 7, 76131 Karlsruhe, Germany
* Correspondence: bradley@kit.edu

Abstract: In order to perform topological queries on geographic data, it is necessary to first develop a topological access method (TOAM). Using the fact that any (incidence or other binary) relation produces a topology which includes the common usage of topology for spatial or spatiotemporal data, here, such a TOAM is developed on the basis of the previously applied concept of Property Graph used in order to manage topological information in data of any dimension, whether time dependent or not. As a matter of fact, it is necessary to have a TOAM in order to query such a graph, and also to have data which are topologically consistent in a certain sense. While the rendering of topological consistency was the concern of previous work, here, the aim is to develop a methodology which builds on this concept. In the end, an experimental test of this approach on a small city model is performed. It turned out that the Euler characteristic, a well-known topological invariant, can be helpful for the initial data validation. Practically, this present theoretical work is seen to be necessary in view of future innovative applications, e.g., in the context of city model simulations, including distributed geo-processing.

Keywords: geometrically induced topology; simplicial complexes; topological algorithms; watertight volumetric model; topological access method

Citation: Jahn, M.W.; Bradley, P.E. Topological Access Methods for Spatial and Spatiotemporal Data. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 533. https://doi.org/10.3390/ijgi11100533

Academic Editors: Peng Yue, Danielle Ziebelin and Yaxing Wei

Received: 20 July 2022
Accepted: 16 October 2022
Published: 20 October 2022

1. Introduction

In [1], the desire for a topological access method was expressed in the context of CityGML data. If the topology and the geometry of a spatial or spatiotemporal data model are treated separately, then one can rely on topological queries for the underlying topological model only. The existence of a special access method that can be used to perform the search within the topological model would facilitate this task. However, the access methods available so far, such as Octree or *R*-tree, are geometry related and necessitate the embedding of the model in Euclidean space. The desired access method would solely use the topological boundary structure of the model in order to find objects which are nearby in the sense of this incidence relation. Other objects, even if geometrically nearby, could be ignored without any harm. This approach exploits the assumption that the topological and geometric model are topologically consistent, i.e., no two atomic objects overlap.

This idea is reflected in our previous definition of *topological consistency*, reviewed below. In order to understand the importance of our notion of topological consistency, we now first illustrate its definition, as well as how it is related to the notion of topology of geographic data, and in general. A situation such as the one in Figure 1 (left) is topologically inconsistent (in our sense), because the topological model sees only two disjointed segments $x \leftarrow \ell_1 \rightarrow y$ and $a \leftarrow \ell_2 \rightarrow b$, whereas the geometrical model also sees the intersection of the vertical line ℓ_1 and the horizontal line ℓ_2. The overlay space in Figure 1 (right) is topologically consistent through the introduction of the extra object $\circ$ in the boundary of ℓ_1 and ℓ_2. The incidence graph of the overlay space is shown in Figure 2.

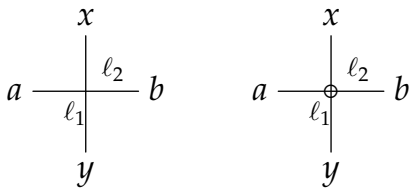

Figure 1. Topologically inconsistent situation (**left**) and overlay space (**right**).

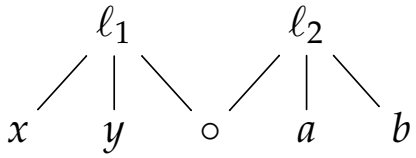

Figure 2. The incidence graph of the overlay space of Figure 1.

This space is topologically connected, and we can see, e.g., that the lines ℓ_1 and ℓ_2 are nearby with respect to the *incidence* relation, whereas in the original topological model, they are not. Observe that in this example, the overlay space is a hyper-graph whose hyper-edges ℓ_1 and ℓ_2 have boundary vertices $x, y, \circ$ and $a, b, \circ$, respectively.

The most general form of a finite topological model is a finite topological space X. The topology of these spaces is generated by the reflexive and transitive closure of a binary relation [2]. An important special case is given when the relation is antisymmetric. In that case, the finite topological space is called a T_0-space, and the topology is generated by a partial order $\leq$. A convenient interpretation of $\leq$ is that of an *incidence* relation. However, other interpretations are also used. In any case, $a \leq b$ means that a is in the closure of b [2].

For example, in Figure 3, the boundary of A consists of the outer rectangle, the inner triangle and the puncture P. This could be, e.g., a cartographic situation in which a region consists of a large metropolitan area (the triangle), a smaller centre (P) and the remainder (A).

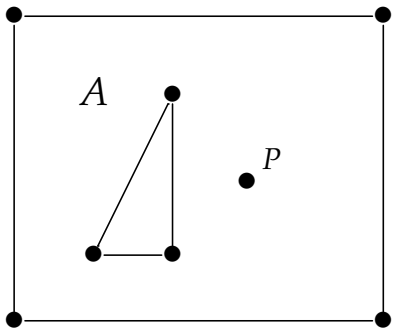

Figure 3. A rectangle with a triangular hole and a puncture.

Assume that some spatiotemporal data are modeled as a finite T_0-space X using an *incidence relation* $\leq$. The Hasse diagram $\mathfrak{H}(X)$ is nothing but the incidence graph: its vertex set is X, and an edge (x, y) is given precisely when $x < y$. Furthermore, $x < z \leq y$ implies $z = y$. The purpose of the Hasse diagram is to have a minimal model for the topology, as an edge indicates a direct relationship. Alexandrov's result [2] states that any partial ordering comes from a topology, even if it is not always intuitively visible at first glance. Since the relationships used in this article are directed and acyclic, they can often be interpreted as *bordered by*, i.e., they have a semantic indicating an underlying topology from the user's viewpoint. Later, we will apply Alexandrov's observation to relationships not usually deemed topological by users in order to have their convenient encoding in Hasse diagrams. Notice that any acyclic-directed graph can be extended to a partial ordering inducing the same T_0 topology, and the Hasse diagram is the unique minimal representation of this

topology, cf. [3]. In the case of this article, the topology (or, in other words, the Hasse diagram) in turn represents certain relationships between geo-objects of any dimension.

If a data model explicitly models the underlying topology, then care has to be taken that there is no contradiction with the underlying geometry. This means that there are in fact two models: the abstract topological model X with partial order $\leq$, and the geometric model. Now, the geometric model has a derived topological model, and topological consistency is given if and only if those two topological models coincide. Otherwise, there will be a topological inconsistency, such as that in the situation of Figure 1 discussed above.

In order to obtain topological consistency, an overlay can be derived by calculating all intersections of atomic objects, as in [4].

It is clear that if a search is to be performed on a topological model, then the correctness of the result is guaranteed if and only if the model is topologically consistent (provided the geometry is correct). Hence, the topological access method (TOAM) assumes the topological consistency of a data model. In reality, the issues of computational geometry pose limits on the possibility of finding and correctly repairing all topological inconsistencies before invoking the TOAM.

Assume two not necessarily convex solids, modeled as polyhedra in 3D with many boundary triangles as spatial boundary representations (BREP) of the solids, touching each other in a few points. The topological query is to find these points. Any spatial access method (*SAM*) such as the Octree [5], the *R*-tree [6] or the *R**-tree [7] can be used to solve this query.

The *SAM* method would go through the following steps:

1. Create a *SAM* for each BREP point of solid A.
2. For each boundary triangle b of solid B, use the *SAM* to retrieve all BREP points C of A which intersect the bounding box of b.
3. For each BREP point c of C, check if b contains c (true: add to the result set).
4. Repeat the last steps to find all BREP points of solid B, respectively.

To calculate the complexity of this method, we assume that the first step is a preprocessing step and does not count towards the complexity. As an example, the search efficiency of the *R*-tree method depends on the segmentation of the *R* tree, which lies on average at $\mathcal{O}(\log_M n_a)$, where M denotes the maximal number of entries per *R* tree node and n_a denotes the number of BREP points of solid A. The worst case reaches to $\mathcal{O}(n_a)$ for one search. Since every boundary triangle of the solid B has to be checked, the complexity extends to $\mathcal{O}(n_b \log_M n_a)$, where n_b denotes the number of boundary triangles.

Now, assume two not necessarily convex moving and morphing solids, modeled as polyhedra by a snapshot model for some time steps in spatiotemporal space with many triangles as spatial boundary representations of the solids at each time step, touching each other in a few BREP points at certain unknown time steps. The topological query is to find these BREP points.

Theoretically, the last pseudo algorithm needs to run for each time step of the temporal overlapping interval of both moving and morphing solids to find every touching point. Therefore, this method adds a factor n_c to the complexity (when using the *R*-tree) as $\mathcal{O}(n_c n_b \log_M n_a)$ for the number of time steps which have to be checked. It is obvious that the precision of the data type used for the temporal coordinate has a significant impact on the complexity. On the other hand, if the spatial change is small enough, the spatial situation might be equal for some time steps. Further research needs to be applied to deal more efficiently with this situation than can be achieved by iterating over each possible time.

Another method includes the tetrahedralizations of the moving and morphing boundary faces of both solids. The tetrahedralizations are the spatiotemporal boundary representations of the solids in spatiotemporal space. The previously described steps can be applied by checking the intersections for each boundary tetrahedron of one solid with the moving points of the other boundary representation. This method has the same complexity as the *SAM* method with $\mathcal{O}(n_b \log_M n_a)$, where n_a denotes the number of moving BREP points of solid A which are spatiotemporal curves, and n_b denotes the number of boundary

tetrahedra of solid *B*. However, the *SAM* turns to a spatiotemporal access method *STAM* and the complexity of calculating the intersections of spatiotemporal curves and tetrahedra differs from the complexity of calculating the intersections of spatial points and triangles. However, the example yields only two intersection types, since the two solids only touch each other sometimes. If the intersection is a point, then the solids touch each other at one point at one time step. If the intersection is a curve, then the touching point moves through time and maybe through space too. However, all possible intersections of both spatiotemporal boundary representations do not need to be calculated in this example.

The last two examples illustrate how geometrically induced topology is queried. Using spatial access methods (SAM) or spatiotemporal access methods (STAM) only implies an implicit topology model. The downside of an implicit topological model is recalculation of intersections, etc., which may also lead to different results if the used geometries are transformed in some ways due to the problems of computational geometry, data precision and arithmetic (floating point or integer).

Another way is to define an explicit topological model for any geometry type used in geographic sciences. Geometrically induced topology is only a small part of topology, since any relation between some entities already defines a topology [2]. Therefore, it is necessary to ask how a general topology model for geo-objects may be designed and how topological access methods can be modeled in order to access the general topological model for geo-objects and to answer topological queries efficiently. The first question has been addressed in [4,8] with a suitable application of Property Graphs, whereas the second question is dealt with in this article.

The main contribution of this article builds on the *Property Graphs* which manage the possible topologies on geo-objects. These graphs are obtained with the methods of [4,8]. The contribution can be summarized as follows:

- The use of topological invariants (here: Euler characteristic) in the initial data validation (pre)process.
- The development of a topological access method (TOAM) for more efficiently querying nodes with topological properties.
- An experimental test of this approach on a small city model.

2. Related Work

Topology now has a long history in geographic and building information. After reviewing some important contributions to the issue of obtaining topological information from data models in general, we will explain in more detail the aspects of topology most relevant to our previous and present work.

Since [9], the topology of city models can be captured in an xml-type data structure named CityGML. It is now a de facto standard in the administration of municipal databases worldwide, and the object of attention of much research. A known issue with CityGML is its topological and also geometric validation, most importantly since the actual models are often rendered from point clouds. The authors of [10] proposed a two-level topological model for 3D features in CityGML, which captures semantics and geometry with extended topological consistency rules to guarantee the validation of CityGML primitives. A quite limited usability of the existing topology mechanism of CityGML in order to extract topological information was found in [11].

The models developed in [12,13] provide geometric information based on CityGML, and the topological relationships of indoor-adjacent spaces in this model are represented by CityGML XLinks. These XLinks form an explicit way of modeling connectivities which can conflict with possible underlying connectivities between polygons in the model. The latter connectivities are revealed through computational geometry calculations.

In the BIM world, there is recent research on information retrieval in the context of smart cities, as well as that which focuses on enabling simulation models. All of this exploits the underlying topology, cf., e.g., [14–17] in order to name a few articles in this

domain. As the two worlds of BIM and GIS have already begun to interact, the issue of topological consistency becomes more and more relevant in both domains.

According to our observation, the potential of the mathematical discipline of topology for an efficient information retrieval using existing topological data structures has not been sufficiently exploited yet, and the existing data structures do not facilitate this task, either. For this reason, the research of our group has concentrated on finding refined methods which enable (mathematical) topology to serve the geographic community in the task of topological information management and retrieval. The following paragraphs review the different concepts related to this work.

Searching through spatial or spatiotemporal data can be carried out efficiently by using a suitable access method. For Euclidean n-space, this is e.g., the Octree [5], the R-tree [6], or the R^*-tree [7]. The reason for mentioning these is that certain existing geometrical access methods are also used in this article. The remainder of this section deals with topological methods.

Topology is a mathematical discipline which formalizes the concept of nearness. It is natural to consider topological models for spatial and spatiotemporal data. An early example of a data structure for such models is given by G-maps [18]. These are used, e.g., in *DB4GeO* [19]. However, G-maps are verbose and have exponential storage complexity with increasing dimension [20]. Considering $3D$ plus time leads to a $4D$ model [21]. The $3D$ plus scale leads to another kind of higher-dimensional model [22]. Of course, space, time, version and scale can be combined to a unifying nD model [23]. Simplicial and polytope complexes were suggested for $3D$ and $4D$ [19].

In [3], it was shown that finite topological spaces—which were first studied by Alexandrov [2]—allow the efficient modeling of any topological situation which can be stored on a computer. Special cases are T_0 spaces (also known as posets) which should be sufficiently general models for spatiotemporal data. The topology of posets is given by the reflexive and transitive closure of a binary relation which is acyclic, i.e., it has no directed cycles.

On the one hand, the separation of geometry and topology leads to efficient topological queries, cf., e.g., [24], where a tree structure is used to navigate through the rooms of a building. Additionally, in [25], efficient spatial and topological queries on large tetrahedral meshes with arbitrary topology and complex boundaries are performed.

On the other hand, this separation also leads to a consistency issue [26]. Namely, one has two models: an abstract topological model and a geometric model, which are obtained by assigning coordinates to objects of the abstract topological model. The underlying topology of the geometric model in general differs from that of the abstract topological model. If they do not, then one can call the models topologically consistent. Observe that the literature contains various differing notions of topological consistency [27–30]. The present notion of topological consistency is based on the idea in [31] which is the first to relate the geometry and the incidence graph of a topological model, at least as far as the authors are aware.

In order to develop efficient topological access methods, which lead to efficient topological query processing, we assume in the following that the model is topologically consistent. That is precisely the situation when any topological query on the topological model yields a correct answer. However, since the raw data are not topologically consistent, they first need to be properly preprocessed, e.g., by using the overlay methods introduced in [4]. It turned out that the Euler characteristic was helpful in this stage.

2.1. Graph Model

The implementation of the topological access method here extends the *DB4GeOGraphS* core framework which had been introduced in [8,26], extended to compute watertight volumetric models from boundary representations in [4,32], and further extended by a p-adic Gray-Hilbert curve index for point clouds in [33] all of which (including this paper) are parts of the dissertation [34].

In this section, we reintroduce the graph model used by the *DB4GeOGraphS* core framework for the self-containment purpose of this paper. The predecessor of the *DB4GeOGraphS* core framework is *DB4GeO*, developed by the working group of Martin Breunig [19], an object-oriented research database for spatial and spatiotemporal data. The *DB4GeOGraphS* core framework is written in JAVA and implements basic algorithms to compare spatial objects and to calculate spatial intersections and differences. The *DB4GeOGraphS* core framework can be seen as a research framework to test data structures, access methods and algorithms which are able to manage and process spatial and spatiotemporal data.

The spatial data types are based on simplicial complexes. There are three different levels of aggregations: the element level, the component level and the net level, where a d-dimensional element is a d-dimensional simplex, a d-dimensional component is a collection of d-dimensional simplices forming a connected d-dimensional manifold or, in other words, a connected d-dimensional simplicial complex. A d-dimensional net is a collection of d-dimensional components in the form of a topological sum. The class names for the different geometry types follow the convention of $A +$ "3D"$+B$, where A denotes the dimension of the simplicial object (*Point*, *Segment*, *Triangle* and *Tetrahedron*), and B denotes the level of aggregation (*Element*, *Component* and *Net*).

The spatiotemporal model is based on the Polthier and Rumpf model [35] and adds four different aggregation levels (element, sequence, component and net). A d-dimensional element is a $d+1$-dimensional polytope in four-dimensional space, where the first three coordinates define the spatial coordinates and the fourth coordinate defines the time step. Each d-dimensional element is defined by two d-dimensional simplices in a fixed 3-dimensional Euclidean space two different times to model a moving and morphing d-dimensional simplex. A d-dimensional sequence is defined by chaining those d-dimensional elements at their d-dimensional border simplices. A d-dimensional component is a collection of d-dimensional sequences forming a connected $d+1$-dimensional manifold or. in other words, a connected $d+1$-dimensional polytope complex. A d-dimensional net is defined as a collection of d-dimensional components in the form of a topological sum. The class names for the different geometry types follow the convention of $A +$ "4D"$+B$, where A denotes the dimension of the simplicial object (*Point*, *Segment*, *Triangle* and *Tetrahedron*) which moves and morphs over some time interval, and B denotes the level of aggregation (*Element*, *Component*, *Sequence* and *Net*).

The *DB4GeOGraphS* core framework combines the *Property Graph Model* with the OGC's *Feature Model* design patterns for node definitions. The *Property Graph Model* is a paradigm well suited to deal with unstructured data. This is often the case when different kinds of geo-information need to be combined for analysis. Simplicial complexes or the described spatiotemporal polytope complexes also belong to the class of unstructured spatial or spatiotemporal dataset types, since the topology is not regular as to be seen in voxel or raster data. This also makes it a bit more difficult to efficiently handle the computation and persistence of simplicial complexes, since the topology needs to be memorized explicitly. However, for example, if neighbors of some d-dimensional simplex are defined by any simplex which shares a $d-1$-dimensional border simplex, then each d-dimensional simplex has a maximal number of d-dimensional neighbors a_{max} with $a_{max} = d+1$ within a d-dimensional simplicial complex. So there is some "regular" topological structure which can be used to implement robust spatial or spatiotemporal algorithms, even if the topology is not as regular as to be seen in voxel or raster data. The use of unstructured data enriches the range of applications. In that sense, regular topology (meaning that the number of direct neighbours is constant) is a special case of the more general unstructured form, and it is due to the research on and the implementation of suitable spatial or spatiotemporal databases to classify the data driven cases and to apply the most efficient way of dealing with those different cases. These are technical problems and should be hidden from the user.

Topology in the *DB4GeOGraphS* core framework is understood in a wider sense than only focusing on *incidence relations*. In the context of complex inter and intra-related spatial or spatiotemporal geo-objects, common relation types of those geo-objects are *incidence relations* (*border-of*, *inner-of*), *aggregation relations* (*part-of*, *composite-of*) and *abstraction relations* (*generalisation-of*, *specialisation-of*). Each relation type may be included bidirectionally or unidirectionally. Unidirectional relation types generate T_0-spaces. A set of T_0-spaces can be combined to a graph, which in turn can be analysed. The *Property Graph Model* replaces the former object-oriented data model of *DB4GeO* which enables the representation of complex geo-information by the definition of *Property Graphs* with spatial or spatiotemporal attributes/properties, temporal attributes/properties and arbitrary thematic attributes/properties on each node and the studies of those graphs.

With all of this in mind, it is possible to model a complex graph which represents a multi-scaled spatiotemporal model with different LoDs. As an example, we could picture a truck driving through a city. A common question would be: "Does the truck interfere with the city?" in the sense that it touches buildings or is blocked by too-narrow passages. A truck could be modelled precisely as a spatiotemporal tetrahedron net or just as a spatiotemporal segment element. The truck could also be modeled as a spatial tetrahedron net which moves through the city by the definition of a spatiotemporal trajectory and directions. It is easy to see that this example will produce different results for each type of simplification. However, the main problem is still computational geometry when dealing with different scales in one model, which will lead to incorrect results in the geometrical computation of topologically consistent models. If the truck is modeled with relatively small simplices and the city model uses much greater simplices, wrong intersections may be found or mathematically true intersections may not be found. Only exact computational geometry and its overhead may deal with this problem more accurately.

Another interesting example is the revolving door at the entrance of some shopping malls. We then ask how to navigate into the mall. The problem is not as trivial as it seems, since a revolving door always blocks the inner of the mall and the outer of the mall. Using topologically consistent spatiotemporal geometries to model this example reveals a topological space which interconnects the outer and inner of the mall, even by the selection of only one thematic space (air).

The most complex way to model this situation is to use spatiotemporal tetrahedron complexes (moving and morphing tetrahedron complexes), five tetrahedron complexes for the door, which includes the definition of four disjointed moving complexes beside the door itself as a glass complex, and two tetrahedron complexes for the inner and the outer mall as air complexes, which are divided by two walls designed as stone complexes. The top row of Figure 4 illustrates the example geometrically flattened to the floor at a certain time step (left) and the spatiotemporal *Property Graph* for the spatiotemporal boundary representations of the air, stone or glass complexes (right). The complexes are connected by aggregation relations, which connect to their shared spatiotemporal geometries (the smallest nodes). Four different spatial *Property Graphs* can be realized for four different time steps by the use of spatial geometries and removing either the red and magenta nodes, or the green and the yellow nodes, or the magenta and green nodes or the yellow and red nodes. The geometrically flattened spatial example from the top of Figure 4 shows the case when removing the red and magenta nodes. However, removing these nodes prevents the possibility of moving from the outdoor complex to the indoor complex by only using air complexes. The thematic of the spatiotemporal geometries does not necessarily need to change temporally in this example.

A very common type of simplification is to use spatiotemporal triangle complexes (moving and morphing triangle complexes) by transforming the 3-dimensional shapes into representing 2-dimensional shapes used within cadastres or some other traditionally 2-dimensional geographic application, which flatten the world. This type of simplification does not necessarily simplify the topology, and spatiotemporal geometries are also needed

to solve this problem. However, this type of abstraction, transforming the spatial or spatiotemporal objects into another d-dimensional space, is not of further interest here.

A less complex model of 3-dimensional shapes within the 3-dimensional space is to reduce their dimension. The walls, doors, roofs, windows and floors may be modeled as surfaces and not as solids. In this case, the walls, doors, roofs, windows and floors of the mall are part of the border of the indoor complex. The middle row of Figure 4 illustrates this type of simplification. The model may consist of only a spatial model together with a temporal interval of existence. To model the opening and closing of the mall, the thematic of the door may change temporally (air to glass and vice versa).

Another very common type of simplification is to reduce the number of entities of interest. In case of the different *LoDs* of CityGML, the wall, door, roof, window, floor and air complexes of the mall are aggregated to reduce the complexity of the model. The border of the mall complex then represents the wall, door, roof, window and floor of the mall. The bottom row of Figure 4 illustrates this simplification. As you can see, this type of simplification blocks the indoors from the outdoors. To model the opening and closing of the mall, the thematic of the wall may change temporally (air to stone and vice versa).

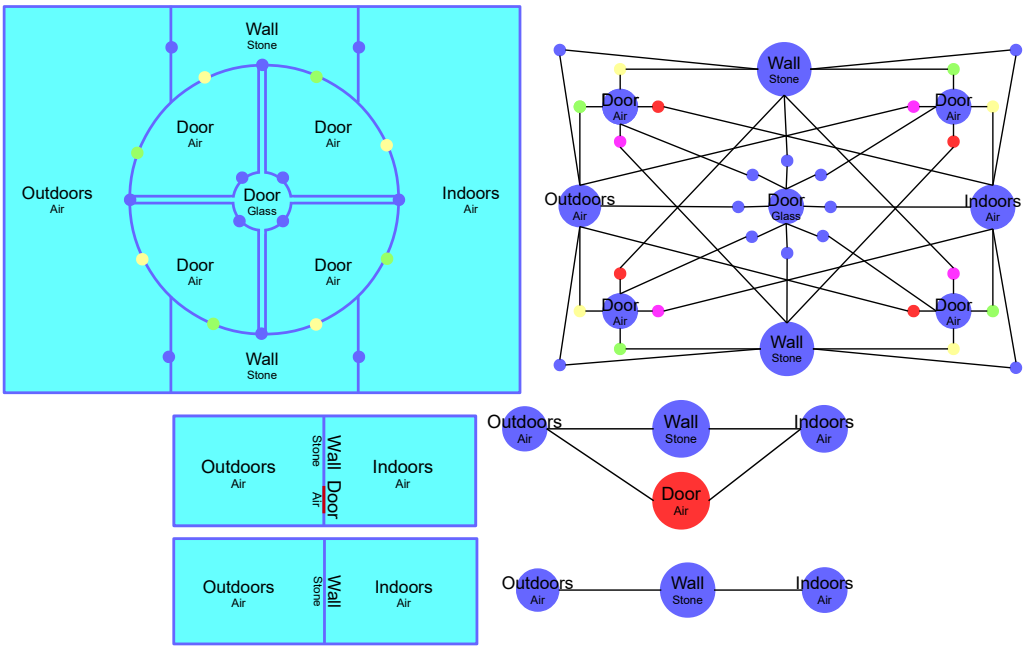

Figure 4. Top: *Property Graph* of the most complex geometry model, which uses spatiotemporal tetrahedron complexes for each abstract entity of interest. The small nodes represent the spatiotemporal intersections of the big nodes. **Middle**: *Property Graph* of the less complex geometry model, which reduces the dimension of some mall geometries. **Bottom**: *Property Graph* of the less complex geometry model, which reduces the mall geometries to one spatiotemporal tetrahedron complex.

The last three examples can be interconnected through the *abstraction relations*, (simplification interpreted as generalization). The three models use internally *aggregation relations* and *incidence relations* only. The advantage is that redundant modeling (opening and closing of the mall) does not necessarily need to be performed, and a query could be parsed to clusters which carry the adequate *LoD*. If, for performance reasons such as reducing the data transfer within a multi database system, redundant modeling is necessary, then automated generation of simplified models and/or cross-checking of the redundant models may be possible.

A common way of querying *Property Graphs* is by the use of a graph traversal language (e.g., Gremlin). Querying a *Property Graph* is achieved by applying step-by-step atomic steps onto a *Property Graph*. Three different kinds of steps exist: (a) a map step (transforming the objects in the stream), (b) a filter step (removing objects from the stream), or (c) a

side-effect-step (computing statistics about the stream) [36]. The atomic steps are provided by the used graph database. *DB4GeOGraphS* core framework implements atomic steps to handle certain features which a spatial or spatiotemporal *Property Graph* database should not miss in order to maintain the topology based on the described relation types. This includes operations to build (a) the border nodes of a given node, (b) the inner of a *BREP* (boundary representation) node or even the inner of a d-dimensional net if the contained d-dimensional components bound some space, (c) the decomposition nodes of an aggregation node, or (d) the interior overlay of a given node with a set of other nodes.

Notice that, although we usually speak of "spatial" and "spatiotemporal" Property Graphs, its construction actually does not depend on the dimensionality of the underlying geo-objects, which in turn are only special properties of the graph nodes.

3. Methodology

The methodology described in the following subsections consists of the development of a topological access method (TOAM) with two parts. First, an access method for topological queries using coordinates is built, and secondly, coordinates for any topology in the form of a T_0-space (mathematically) and *Property Graph* (technically) are defined, giving an encoding of the topology. Figure 5 schematically shows the breaking up of the TOAM into these two constituent parts.

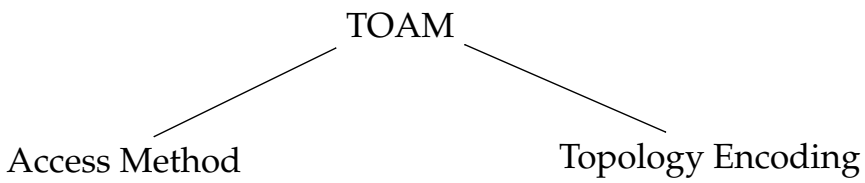

Figure 5. The Topological Access Method consists of an Access Method and a Topology Encoding.

3.1. Access Method

As mentioned at the end of the previous section, we will speak of either "spatial" or "spatio-temporal" Property Graphs, when the managed geo-objects either do or do not have time-changing properties. Notice that the dimension number of these objects can even be higher than three or four without changing the methodology presented below. In addition to the definition of atomic steps for a spatial or spatiotemporal *Property Graph* traversal language, it is also important to integrate suitable access methods which integrate the properties of spatial or spatiotemporal data in order to support big geo-scientific data analysis based on those spatial or spatiotemporal *Property Graphs* efficiently. It is not a problem to apply a spatial access method such as the *R*-Tree* on a set of spatial nodes. This is similar to applying a *R*-Tree* on the spatial column of an object-relational database such as PostGreSQL with PostGIS extension. However, topological access methods refer to the topology of spatial or spatiotemporal objects, to the T_0-spaces they are part of or span, not to the geometrical spaces they live in. In fact, the geometrical space may be of any dimension, and the geo-objects are endowed with a well-defined topology for each relation type, as introduced in the previous sections generally, and also for the spatial and spatiotemporal case. So, the geometrical space is spanned by the spatial or spatiotemporal metric space and some other metric thematic attributes. The geo-objects are related by n basic relation types to describe n topologies managed by the *Property Graph*. In the following, the concepts are illustrated with spatial examples only to ease the understanding. Spatiotemporal or even higher-dimensional examples (e.g., as in Figure 4) can be treated as described similarly because the relation types do not change.

DB4GeOGraphS core framework *indizes.api* package contains a *TOAM* interface for topological access methods (TOAM), now. Our prototype, the first implementation of a topological access method within *DB4GeOGraphS* core framework, is based on a discrete *R*-Tree* which manages multidimensional discrete points. Any other implementation of access

methods which are able to manage multidimensional discrete point clouds are conceivable for our approach. As a matter of fact, some other implementations (e.g., a space-filling curve index) or even combinations of access methods (e.g., one one-dimensional access method per coordinate) may work more efficiently. This is a work in progress.

3.2. Topology Encoding

We have tested a few different ways of defining each coordinate. One of the simplest definitions is to count the number of relations of each node. As described previously, three different kinds of bidirectional relation types exist, which are only usable as six different unidirectional relation types within *DB4GeOGraphS* core framework. So the definition can be extended to define a multidimensional key by counting per relation type. The main argument for this particular definition is that it is possible to retrieve geo-objects with certain local topological properties (e.g., retrieve all geo-objects which are *part-of* 2 to 4 other geo-objects). It is also possible to take other node properties into account e.g., some thematic attributes, its spatial dimension or its aggregation level or even the spatial or spatiotemporal property itself, which would lead to a hybrid spatial or spatiotemporal topological access method.

If we concentrate on access methods which organize the structure of the *Property Graph* in certain ways, another definition for the coordinates of a multidimensional key to identify a node is given by the number of steps taken along the shortest path to a pivot node within the *Property Graph* using a Dijkstra algorithm. Trivially, the pivot node needs to be reachable, which implies that the *Property Graph* is a coherent component. An argument to take this definition could rely on the importance of a pivot element (e.g., retrieve all geo-objects which are in a certain topological range to some pivot geo-object). This definition is extendable by analyzing the shortest path to the pivot node. A multidimensional version is to count the steps of the shortest path per relation type separately. The sum over each coordinate is equal to the number of steps of the shortest path to reach the pivot node. A practical use is the ability to query for nodes which connect to the pivot node within a topological range of certain relation types (e.g., retrieve all nodes which connect to a pivot node of interest by at least one *border-of* relation step and 2 to 4 *part-of* relation steps). The relation type may also be used to add different weights to the steps of a Dijkstra algorithm. The practical use of the last idea is to have more control over the Dijkstra algorithm. However, the complexity to calculate the coordinates by those definitions as a preprocessing step depends on the graph structure.

Another interesting definition which leads to a rather complex preprocessing step to calculate each coordinate of some node within the *Property Graph* is the sum of the shortest path lengths of a node to every other node. For example, if every node is without detours connected with every other node within the *Property Graph*, each node has the same shortest path length to each other node which is equal to one. The coordinate would be equal to $n - 1$ for every of the n nodes. If we picture a p-adic tree as a second example, the root node would be in the centre and gets the lowest sum of shortest path lengths and the leaf nodes get the highest sum of shortest path lengths since the paths over some parent node to another leaf node will always add some extra steps instead of going directly to each of the leaf nodes from the parent node. So this definition of a coordinate is a measure for how close to the centre of the *Property Graph* the location of the node actually is. This definition can also be extended by the use of the different relation types.

Some Examples of the previously described definitions are given in Figures 6 and 7. The previously described definitions are given in the form of three vectors below each example graph. Each example graph depicts only the relations on the shortest path when starting at the node with value 0. The first vector on the left shows the number of relations per relation type of the node with value 0. The ordering of the vector entries is *part-of*, *composite-of*, *border-of* and *inner-of*. The second vector shows the number of steps taken to reach the pivot node (red bounded node) where the first coordinate ignores the relation type followed by the number of steps per relation type in the same order as shown previously.

The third vector shows the sum of each shortest path lengths from the node with value 0 to each of the other nodes, where the first coordinate ignores the relation type and the other coordinates sum the steps taken by certain relation types along the shortest paths in the same order as shown previously.

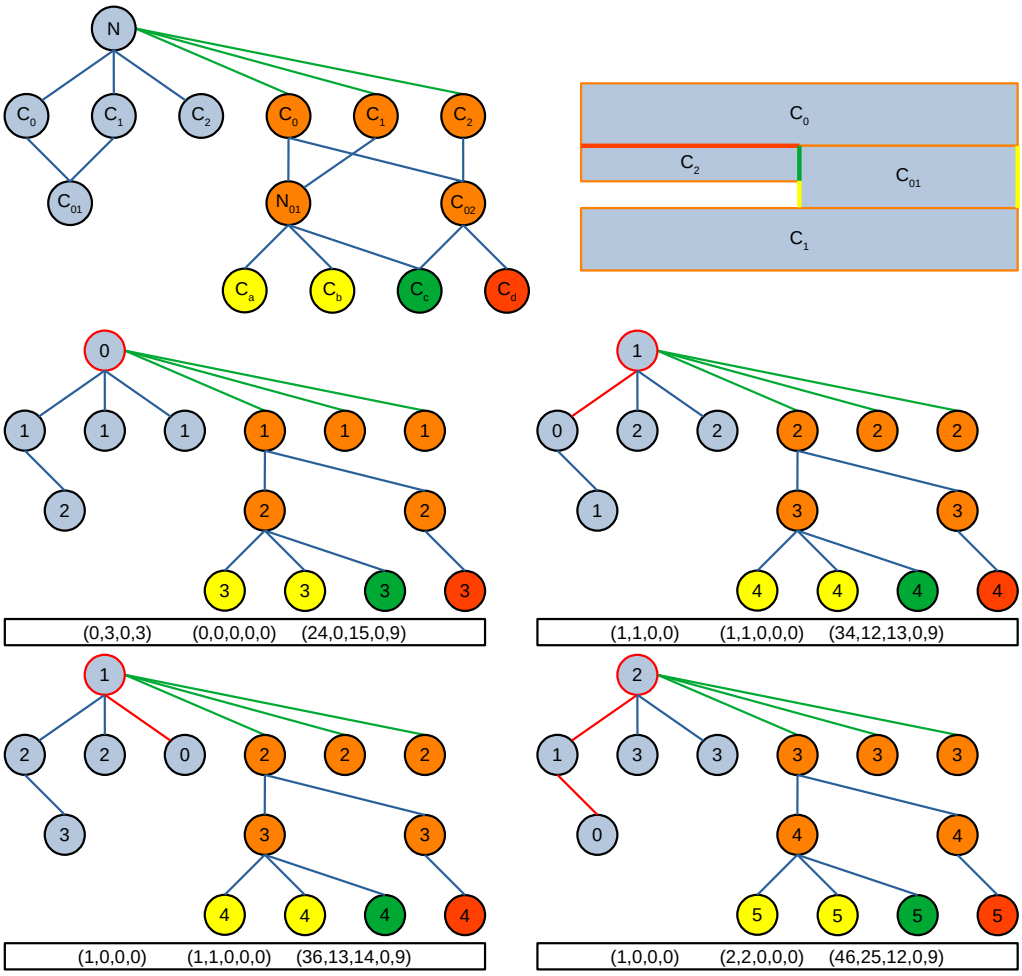

Figure 6. The top-left figure shows the *Property Graph* of the topologies from the shapes shown in the top-right figure. The relations in the *Property Graph* are *part-of* (red), *composite-of* (blue), *border-of* (orange) and *inner-of* (green). The figures in the second and third row show the shortest paths of the Dijkstra algorithm when starting at the node with value 0. The values of the other nodes represent the number of steps taken to the node with value 0. The vectors below represent the topological properties of the node with value 0 (see last paragraph of Section 3).

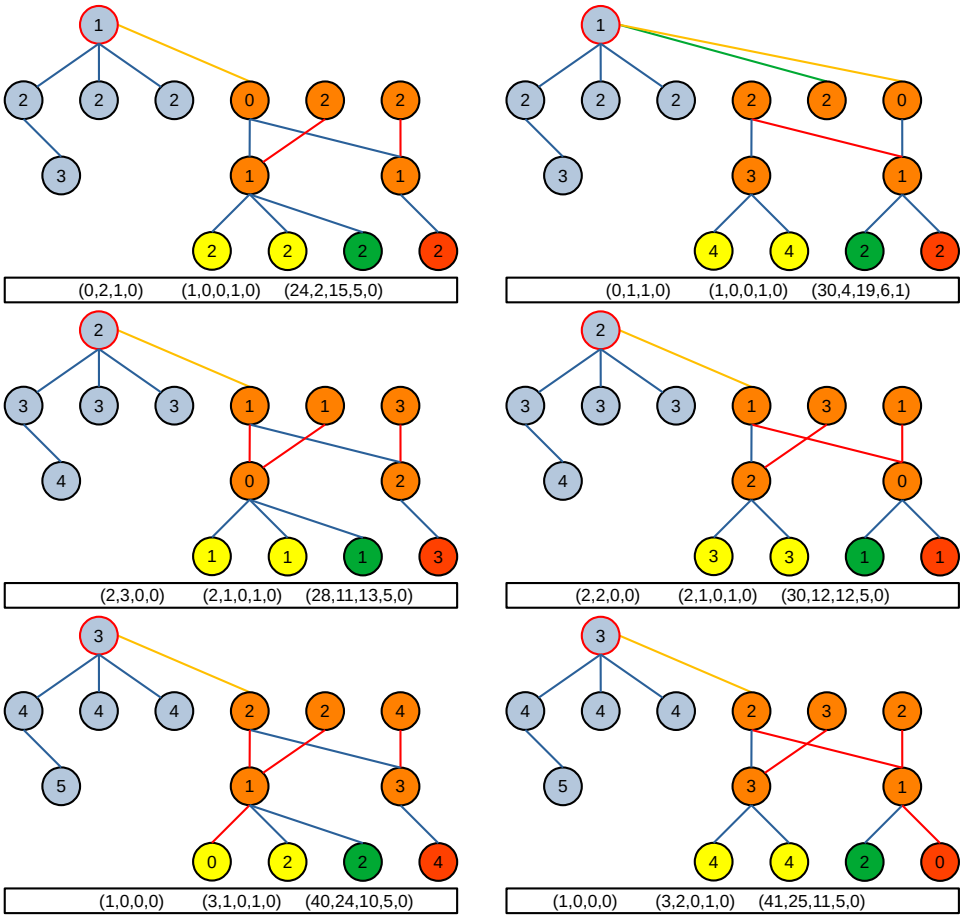

Figure 7. The figures show the relation types which the Dijkstra algorithm had chosen when starting at the node with the value 0 from the example given in Figure 6 (top right). The values of the other nodes are the shortest distances to the node with value 0. The vectors below represent the topological properties of the node with value 0 (see last paragraph of Section 3).

4. Experiments

In the previous section, we introduced a topological access method (*TOAM*) for general spatiotemporal data with possibly further metric attributes together with encoded directed relationships between geo-objects. At the moment, the implemented topological access method of *DB4GeOGraphS* can only handle spatial geometries, but an extension to higher-dimensional geometries is a work in progress.

The topological access method is exemplarily calculated for a tetrahedralized city model of a historical part of Salvador (Brazil) which had been modeled by the working group of Arivaldo Leão de Amorim. The data were produced by the Laboratory of advanced studies on City Architecture and Digital technologies (LCAD) at Federal University of Bahia (UFBA) during the research project "Establishing requirements for City Information Modelling", supported by the Brazilian federal government agency *Coordination for the Improvement of Higher Education Personnel* (CAPES) and the *German Academic Exchange Service* (DAAD). The city model used for this work is not a final version. It was used within a workshop in Salvador in the year 2019 as part of the exchange program. The key data are summarized as follows:

1. Zero: 462.851043701172 376.195983886719 71.6225051879883.
2. *CityGML* tree nodes: 22386.
3. Number of groups: 56.
4. Polygons: 925.
5. Distinct polygons: 924.

6. Decimal places: 15.
7. Min. point distance [m]: 6.216820473225959E-6.
8. Segments: 4446.
9. Min. length [m]: 3.0517578125E-5.
10. Average length [m]: 5.0552051851945565.
11. Max. length [m]: 32.06260853268529.

As can be seen, the geometric scale varies considerably, which will produce some numerical issues in the intersection calculations with the effect that the topology is not well defined, as the focus is not on the intricacies of computational geometry, but on the calculated overlay space. We ignore this issue here, even if the graph does not reflect the true topology. The main point here is to have topologically consistent data (as far as the computational geometry algorithms allow) to work with.

The city model shows an accuracy with maximal 15 decimal places. The sensitivity analysis as described in [32] or [34] showed promising results with positional precision value $\epsilon = 10^{-2}$ and angular precision value $\zeta = 10^{-14}$, even if some corner points of some polygons were ignored during import. This means that the calculated overlay space may also lack consistency with the geometric model. Significant topological inconsistencies are possible, as further analysis might reveal. However, this is not the focus of this article. In total, 50 out of 56 buildings could be tetrahedralized with one tetrahedron complex per building and a overall volume of 68916 m^3 (see top of Figure 8).

In the preprocessing step, we analyzed the distribution of the Euler characteristic which here needed to be given in its combinatorial form, as the data are initially topologically inconsistent. The distributions of the combinatorial Euler characteristics show that the choice of the precision parameters could be in order in view of the expected topologies. The combinatorial Euler characteristics are a quick way to analyze the topological quality of geo-objects. Therefore, if the combinatorial Euler characteristics are plausible, the calculated interior overlays and the resulting *Property Graph* (overlay space) where each interior overlay node is well integrated will likely be nearly correct as far as the used computational geometry algorithms find all correct intersections. In the present case, there are three tetrahedron complex created by the interior overlay with a Euler characteristic of one. There are 52 triangle complexes found by the border interior overlay, one of which shows a combinatorial Euler characteristic equal to zero, the others equal to one (see Table 1 left). There seems to be no 2-dimensional sphere involved, since a combinatorial Euler characteristic equal to two is not contained. This means that the dataset does not seem to contain closed shells. Seven *Triangle3DNet* objects consist of a minimum of two triangle complexes and three *Triangle3DNet* objects consist of a minimum of three triangle complexes (see Table 1 right). Those *Triangle3DNet* objects need to be observed for further investigations. The border interior overlays resulting in one and zero dimensional intersections are provided in Tables 2 and 3. As can be seen in Table 3, there is one point cloud of 19 points found as the intersection between two buildings. This is unlikely to be found in reality. Therefore, the intersection calculations showing higher Euler characteristic values should be manually checked for plausibility or topological errors.

Figure 8 (top) shows the tetrahedralized *CityGML* file and their interior overlays. Figure 8 (bottom) shows the corresponding graph, which includes the *CityGML* tree with the nodes which carry the "planar" polygons, their aggregations (grouped by building), the triangulation and the tetrahedralization of each building, the border of each building, and finally the interior intersections of each building and the border to create an overlay space for the buildings. Each *Triangle3DNet* object (building) is tetrahedralized by the *buildCores*-Algorithm (see [4,32] or [34]) separately. This figure also illustrates the border *Triangle3DComponent* objects of the *Tetrahedron3DComponent* objects. The whole graph has been created by using only the node operations mentioned in Section 2.1 and a *CityGML* importer, which is able to group *CityGML* polygons by some XML-Tag.

Table 1. Distributions of combinatorial Euler characteristics of triangle complexes (left) and *Triangle3DNet* objects (right) from interior overlay.

Euler	0	1	2	3
Count	1	51	7	3

Table 2. Distributions of combinatorial Euler characteristics of segment complexes (left) and *Segment3DNet* objects (right) from border interior overlay.

Euler	1	2
Count	21	4

Table 3. Distribution of combinatorial Euler characteristics of point complexes from border interior overlay.

Euler	3	4	6	7	8	12	15	19
Count	1	1	2	2	2	2	1	1

We tested a combination of 23 different attributes, including not only coordinates derived from the topological setting of each node, but also those derived from the geometric properties (dimension and aggregation level). This has the benefit of querying geometric properties together with the topological setting (e.g., return every surface which is part of at least two geometries). The coordinate definitions are as follows:

1. TONode ID.
2. TONode dimension.
3. TONode aggregation level.
4. Relation count.
5. Relation count of SPECIALISATION_OF.
6. Relation count of GENERALISATION_OF.
7. Relation count of PART_OF.
8. Relation count of COMPOSITE_OF.
9. Relation count of BORDER_OF.
10. Relation count of INNER_OF.
11. Distance to root.
12. Distance to root by SPECIALISATION_OF.
13. Distance to root by GENERALISATION_OF.
14. Distance to root by PART_OF.
15. Distance to root by COMPOSITE_OF.
16. Distance to root by BORDER_OF.
17. Distance to root by INNER_OF.
18. Accumulated distances.
19. Accumulated distances by SPECIALISATION_OF.
20. Accumulated distances by GENERALISATION_OF.
21. Accumulated distances by PART_OF.
22. Accumulated distances by COMPOSITE_OF.
23. Accumulated distances by BORDER_OF.
24. Accumulated distances by INNER_OF.

We used a discrete *R*-Tree* to create the TOAM which manages this 23-dimensional set of points (excluding the TONode ID). As mentioned in Section 3, any other access method based on metric attributes would work. The *R*-Tree* consists of seven hyper-cuboid levels to be able to manage all the 23570 nodes (from the *CityGML* tree, the surface triangulations, the solid tetrahedralizations and the new nodes resulting from the inner overlays). Figure 9 illustrates 3-dimensional projections of the *R*-Tree*. The top row focuses on topological coordinates only, where the X-Axis explains the COMPOSITE_OF relation, the Y-Axis the

PART_OF relation and the *Z*-Axis ignores the relation type. The bottom row focuses on the two geometric properties, the aggregation level on the *X*-Axis and the dimension on *Y*-Axis. The *Z*-Axis on the left column shows the overall relation counts, the *Z*-Axis on the middle column shows the distances to the root node, and the *Z*-Axis on the right column shows the accumulated distances.

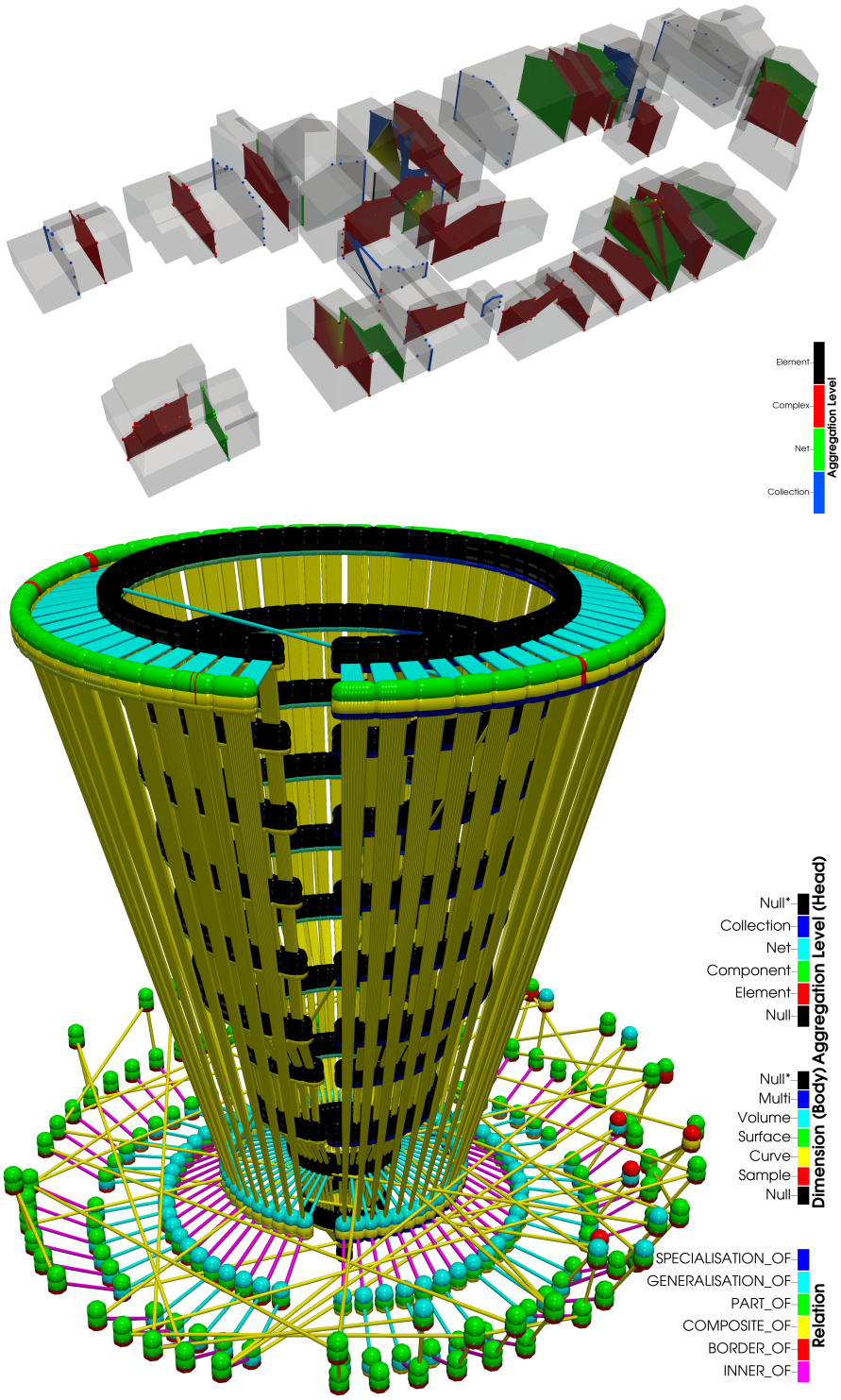

Figure 8. **TOP**: Salvador's historical city center overlays of tetrahedron complexes colored by aggregation level. **BOTTOM**: Salvador's historical city center inner overlays (colored by aggregation level and dimension) of the *Tetrahedron3DComponent* objects (green head and cyan body) and their borders as closed (without boundary) *Triangle3DComponent* objects (green head and green body).

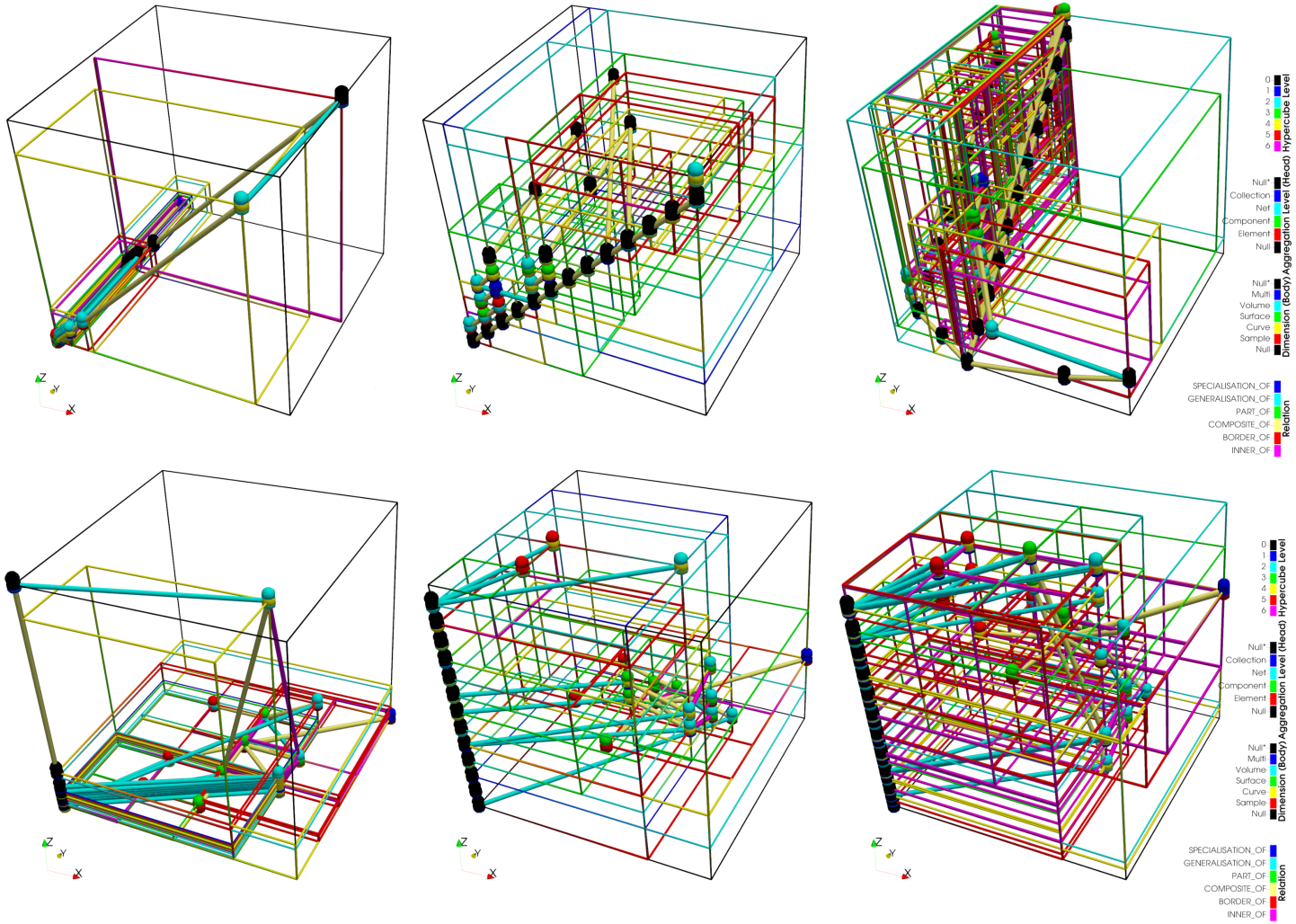

Figure 9. TOP LEFT X: *composite-of* count; TOP LEFT Y: *part-of* count; TOP MIDDLE X: *composite-of* distance to root; TOP MIDDLE Y: *part-of* distance to root; TOP RIGHT X: *composite-of* accumulated distances; TOP RIGHT Y: *part-of* accumulated distances; BOTTOM X: aggregation level; BOTTOM Y: dimension; LEFT Z: relation count; MIDDLE Z: distance to root; RIGHT Z: accumulated distances.

5. Discussion

The task of finding topological properties of geographical data is fairly complex. The *Property Graph* is able to manage any kind of topology. The structure of those graphs can be analyzed by using the encoded relationships between geo-objects. The previous work enabled us to generate such graphs from topologically inconsistent data (up to geometric uncertainty), and in this work we extracted some topological properties suitable for efficient access through the developed TOAM.

The time complexity to find, e.g., intersections within the geometric-based approach depends on the used spatial or spatiotemporal access method and the complexity of the involved corresponding objects. If the application is a simulation, then each spatial object will repeat the intersection operation to find topological neighbors. Furthermore, to find a connection of two objects which may be connected through other objects, e.g., a pen on a desk with the floor, path-finding algorithms, such as the Dijkstra algorithm, need to iteratively check a set of objects for intersections in order to find the shortest path to some object. Those intersections need to be taken into account when solving topological queries, such as finding a connection between two objects, and retrieve the connecting objects, e.g., the desk. Speaking of the sets of intersections which are connected to each object, each object may be connected to each other object. This is also the worst-case scenario, which yields $\mathcal{O}(n^2)$ as time complexity for each step of the path-finding algorithm, which would be only one step if each of the n objects were interconnected. In reality, the topological graph of geographic objects is not that simple and the sets of intersections of each object are relatively small (e.g., the pen on the desk is attached with a bounded number of other objects in the database), which leads to more steps of the path-finding algorithms to find the shortest path to any other object. Holding those sets in memory by mapping the topology turns the time complexity of finding intersections into memory complexity (they have been found once as a preprocessing step). The preprocessing steps to find each intersection and to adjust the graph can be done whenever a spatial or spatiotemporal object is changed or added to keep the graph consistent with the geometrical situation. This can turn into a bottleneck. However, the graph itself enables more opportunities than only being used for shortest path queries.

We consider here data which are already topologically consistent, and the topologies are presented by a *Property Graph* as described, cf. [8]. This will cost additional memory, but helps to establish an access method to query nodes by their topological properties. We extracted 23 topological properties prototypically and tested the outcome. Efficient queries are now possible. This result can be easily extended to an arbitrary number of topological properties. This amounts to having the ability of arbitrarily fine topological queries on geographic data. What remains for future work is to expand this abstract method to the generation of specific user-related topological queries.

The presented accessing method (TOAM) makes use of the structure of the graph. Figure 9 shows that in the example dataset, the diversity of some topological properties is relatively low, e.g., the relation counts, which leads to the following problem of degenerate access methods described below. This due to the fact, as mentioned before, the number of intersections of a given object with other objects in the database is bounded. On the other hand, some properties (e.g., distance to a root node, accumulated distances) show a more uniform distribution which helps the TOAM to avoid the degeneracy issues described below, and thus can perform more efficiently.

The distribution of the coordinates (of the 23-dimensional vector, in our case) is not necessarily well suited to some access methods, since equal hash codes created by, for example, a space filling curve, lead to overfilling of the corresponding hyper-cube [33], or if the coordinates concentrate on some mean with a relatively low variance using an access method such as the *R*-tree can lead to overlapping of its hyper-cubes, which leads to bad access behavior, since more branches of the *R*-tree need to be looked up to find all objects.

The main performance problems (relatively) when importing a node into the topological access method appeared when calculating the distance to some root node and the

accumulated distance. To calculate the distance to a root node, the shortest path to that root needs to be found. The complexity to do so is given by the used path searching algorithm. Calculating the accumulated distances is more complex, since every shortest path from a query node to any other node needs to be calculated and summed up. Both definitions are also not easy to maintain consistently if the graph changes.

6. Conclusions

In this more theoretical article, the focus is on the development of a new methodology for enabling topological queries on geo-data of any dimension, whether time dependent or not. The article contains the first experiments on city model data. In a later stage, this will certainly lead to more research on a practical implementation in order to find efficient solutions to real-world problems involving geo-data. The experiments here are conducted on a small model, rather than on a full-fledged city model at a large scale, as the point is more to illustrate the methods than to find optimal results.

In order to validate the graph derived from tetraheralized buildings and the interior intersections, the Euler characteristics helped with the comparison of the graph structure and for plausibility checks. Further topological invariants could help to refine these auxiliary analyses. Of course, these invariants in general cannot distinguish topological data up to isomorphism, and furthermore, invariants which can do this would lead to computationally hard problems (e.g., a graph isomorphism problem).

Another issue is the presence of geometries in different scales. Experiments in our previous work [32] have shown that finding reliable intersections becomes a problem when the scales of the geometries vary too much. However, the graph model can manage geometric data of different scales, and we can use our TOAM without problems even in this case. It is "only" the preprocessing step of consistent rendering which relies on a good solution of the numerical issues when the scales vary too much.

The method for topological property extraction yields versatile application. The presented prototype of a topological access method (TOAM) has the benefit of finding nodes which fulfil certain topological properties. We have examined only a few definitions of topological properties. In our case, we are able to find nodes with certain local topological properties, e.g., the relation count overall and of certain relation types, and the accumulated minimal distances to every other node as a measure for the position in the graph, using any relation type or some specified relation type. We are also able to find nodes with certain global topological properties, e.g., nodes within a certain distance interval to a root node, using all relation types or some specified relation type. However, the definition of those topological properties is mainly due to the application.

Future research would include an extension to nD topology, as well as the definition of useful topological properties which are calculable and accessible in a reasonable amount of time, and further research on hybrid access methods to solve geometrical and topological queries in one combined access method for geographic data. We did touch on this idea by including coordinates such as the dimension of the object or the aggregation level of the object. The combination of geometric properties of the spatial object together with the topological setting of the spatial object extends the possible queries in a useful way. Additionally, a comparison of our new topological access method with geometry-based methods to solve certain topological queries in an acceptable timeframe is of interest for future work.

Author Contributions: Conceptualization, Markus Wilhelm Jahn and Patrick Erik Bradley; methodology, Markus Wilhelm Jahn and Patrick Erik Bradley; software, Markus Wilhelm Jahn; validation, Markus Wilhelm Jahn; formal analysis, Markus Wilhelm Jahn and Patrick Erik Bradley; investigation, Markus Wilhelm Jahn and Patrick Erik Bradley; resources, Markus Wilhelm Jahn; data curation, Markus Wilhelm Jahn; writing—original draft preparation, Markus Wilhelm Jahn and Patrick Erik Bradley; writing—review and editing, Markus Wilhelm Jahn and Patrick Erik Bradley; visualization, Markus Wilhelm Jahn; supervision, Patrick Erik Bradley; project administration, Patrick Erik Bradley;

funding acquisition, Patrick Erik Bradley. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Deutsche Forschungsgemeinschaft under grant numbers BR 2128/18-1 and BR 3513/12-1.

Data Availability Statement: The publication of the implemented methods, and the datasets not having propriety restrictions, is in preparation.

Acknowledgments: We thank Arivaldo Leão de Amorim for providing data produced by the Laboratory of advanced studies on City Architecture and Digital technologies (LCAD) at the Federal University of Bahia (UFBA). We acknowledge support by the Deutsche Forschungsgemeinschaft and Open Access Publishing Fund of the Karlsruhe Institute of Technology. The anonymous reviewers are thanked for their helpful remarks, criticisms and suggestions that helped us to improve the paper, as well as their ideas for further research.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Giovanella, A.; Bradley, P.; Wursthorn, S. Detection and Evaluation of Topological Consistency in CityGML Datasets. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2018**, *IV-4*, 59–66. [CrossRef]
2. Alexandrov, P. Diskrete Räume. *Mat. Sb. (N. S.)* **1937**, *2*, 501–518.
3. Bradley, P.; Paul, N. Using the relational model to capture topological information of spaces. *Comput. J.* **2010**, *53*, 69–89. [CrossRef]
4. Jahn, M.; Bradley, P. Computing watertight volumetric models from boundary representations to ensure consistent topological operations. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2021**, *VIII-4/W2-2021*, 21–28. [CrossRef]
5. Meagher, D. *Octree Encoding: A New Technique for the Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer*; Technical Report IPL-TR-80-111; Rensselaer Polytechnic Institute, Troy, NY, USA, 1980.
6. Guttman, A. R-Trees: A Dynamic Index Structure for Spatial Searching. In Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data–SIGMOD '84, Boston, MA, USA, 18–21 June 1984.
7. Beckmann, N.; Kriegel, H.; Schneider, R.; Seeger, B. The *R**-tree: An efficient and robust access method for points and rectangles. In Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data-SIGMOD '90, Atlantic City, NJ, USA, 23–25 May 1990.
8. Jahn, M.; Kuper, P.; Breunig, M. Efficient Spatio-Temporal Modelling to Enable Topological Analysis. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* 2022, *accepted for publication*. [CrossRef]
9. Kolbe, T. Representing and Exchanging 3D City Models with CityGML. In *3D Geo-Information Sciences*; Lee, J., Zlatanova, S., Eds.; Lecture Notes in Geoinformation and Cartography; Springer: Berlin/Heidelberg, Germany, 2009.
10. Li, L.; Luo, F.; Zhua, H.; Ying, S.; Zhao, Z. A two-level topological model for 3D features in CityGML. *Comput. Environ. Urban Syst.* **2016**, *59*, 11–24. [CrossRef]
11. Salleh, S.; Ujang, U. Topological information extraction from buildings in CityGML. *IOP Conf. Ser. Earth Environ. Sci.* **2018**, *169*, 012088. [CrossRef]
12. Salleh, S.; Ujang, U.; Azri, S.; Choon, T. Spatial Adjacency Analysis of CityGML Buildings via 3D Topological Data Structure. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *XLII-4/W16*, 573–579. [CrossRef]
13. Sun, Q.; Zhou, X.; Hou, D. A Simplified CityGML-Based 3D Indoor Space Model for Indoor Applications. *Appl. Sci.* **2020**, *10*, 7218. [CrossRef]
14. Demian, P.; Ruikar, K.; Sahu, T.; Morris, A. Three-Dimensional Information Retrieval (3DIR): Exploiting 3D Geometry and Model Topology in Information Retrieval from BIM Environments. *Int. J.-Inf. Model. (IJ3DIM)* **2016**, *5*, 67–78. [CrossRef]
15. Jeong, W.; Son, J. An Algorithm to Translate Building Topology in Building Information Modeling into Object-Oriented Physical Modeling-Based Building Energy Modeling. *Energies* **2016**, *9*, 50. [CrossRef]
16. Wu, Y.; Shang, J.; Hu, X.; Zhou, Z. Extended Maptree: A Representation of Fine-Grained Topology and Spatial Hierarchy of Bim. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *42, 2/W7*, 409–415. [CrossRef]
17. Pasquale, L.; Ghezzi, C.; Pasi, E.; Tsigkanos, C.; Boubekeur, M.; Florentino-Liano, B.; Hadzic, T.; Nuseibeh, B. Topology-Aware Access Control of Smart Spaces. *Computer* **2017**, *50*, 54–63. [CrossRef]
18. Lienhardt, P. N-Dimensional generalized combinatorial maps and cellular quasi-manifolds. *Int. J. Comput. Geom. Appl.* **1994**, *4*, 275–324. [CrossRef]
19. Breunig, M.; Kuper, P.; Butwilowski, E.; Thomsen, A.; Jahn, M.; Dittrich, A.; Al-Doori, M.; Golovko, D.; Menninghaus, M. The Story of DB4GeO - A Service-Based Geo-Database Architecture to Support Multi-Dimensional Data Analysis and Visualization. *ISPRS J. Photogramm. Remote Sens.* **2016**, *117*, 187–205. [CrossRef]
20. Bradley, P.; Paul, N. Comparing *G*-maps with other topological data structures. *GeoInformatica* **2014**, *18*, 595–620. [CrossRef]
21. Worboys, M. A unified model for spatial and temporal information. *Comput. J.* **1994**, *37*, 26–34. [CrossRef]
22. van Oosterom, P. Variable-scale topological data structures suitable for progressive data transfer: The GAP-face Tree and GAP-edge Forest. *Cart.-Graphy Geogr. Inf. Sci.* **2005**, *32*, 331–346. [CrossRef]

23. Paul, N.; Bradley, P. Integrating Space, Time, Version, and Scale using Alexandrov Topologies. *Int. J.-Inf. Model. (IJ3DIM)* **2015**, *4*, 64–85. [CrossRef]
24. Krämer, T.; Huhnt, W. *Topological Information in Geometrical Models of Buildings*; International Workshop on Computing in Civil Engineering: Austin, TX, USA 2009.
25. Fellegara, R.; De Floriani, L.; Magillo, P.; Weiss, K. Tetrahedral Trees: A Family of Hierarchical Spatial Indexes for Tetrahedral Meshes. *ACM Trans. Spat. Algorithms Syst.* **2020**, *6*, 1–34. [CrossRef]
26. Jahn, M.; Bradley, P.; Al Doori, M.; Breunig, M. Topologically consistent models for efficient big geo-spatio-temporal data distribution. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *IV-4/W5*, 65–72. [CrossRef]
27. Dušan, J.; Branislav, B. Elements of spatial data quality as information technology support for sustainable development planning. *Spatium* **2004**, *11*, 88–83.
28. Li, S. On topological consistency and realization. *Constraints* **2006**, *11*, 3151. [CrossRef]
29. Kang, H.; Li, K. Assessing topological consistency for collapse operation in generalization of spatial databases. In *Perspectives in Conceptual Modeling*; Lecture Notes in Computer Science; Akoka, J., Ed.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3770.
30. Rodriguez, M.; Brisaboa, N.; Meza, J.; Luaces, M. Measuring consistency with respect to topological dependency constraints. In Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, San Jose, CA, USA, 2–5 November 2010; pp. 182–191.
31. Bradley, P. Supporting Data Analytics for Smart Cities: An Overview of Data Models and Topology. In *Statistical Learning and Data Sciences SLDS 2015*; Gammerman, A., Vovk, V., Papadopoulos, H., Eds.; Springer: Berlin/Heidelberg, Germany, 2015; LNCS 9047, pp. 406–413. [CrossRef]
32. Jahn, M.; Bradley, P. A Robustness Study for the Extraction of Watertight Volumetric Models from Boundary Representation Data. *ISPRS Int. J.-Geo-Inf.* **2022**, *11*, 224. [CrossRef]
33. Bradley, P.; Jahn, M. On the Behaviour of *p*-Adic Scaled Space Filling Curve Indices for High-Dimensional Data. *Comput. J.* **2020**. [CrossRef]
34. Jahn, M. Distributed & Parallel Data Management to Support Geo-Scientific Simulation Implementations. Ph.D. Thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, 2022.
35. Polthier, K.; Rumpf, M. A concept for time-dependent processes. In *Visualization in Scientific Computing*; Goebel, M., Mueller, H., Urban, B., Eds.; Springer: Vienna, Austria, 1994; pp. 137–153.
36. TinkerPop Gremlin. 2022. Available online: https://tinkerpop.apache.org/gremlin.html (accessed on 18 October 2022).