GI_Forum 2020, Issue 2
Page: 107 - 123
Full Paper
Corresponding Author:
julian.bruns@disy.net
DOI: 10.1553/giscience2020_02_s107

Spatial Operators for Complex Event Processing

Julian Bruns[1], Florian Micklich[1], Johannes Kutterer[1], Andreas Abecker[1] and Philipp Zehnder[2]

[1]Disy Informationssysteme GmbH, Germany
[2]FZI – Das Forschungszentrum Informatik, Germany

Abstract

The types of data available have changed in the last decade. While, historically, data were gathered in batches and distributed as such, e.g. as a database or shapefile, today we are dealing increasingly with real-time data. This data is produced and consumed continuously in real time. The phenomenon is most commonly known as streaming data. Traditionally, software for spatial analysis, such as a Geographical Information System (GIS) or spatial database, was created and optimized for the batch processing of data. However, the inherent characteristics of streaming data provide new challenges for data-stream processing systems, which have not yet been solved. In this paper, we propose enhancing systems for the handling and analysis of streaming data through the use of spatial operators. We identify Complex Event Processing (CEP) as a promising underlying concept for such a system and use the (open source) self-service IoT toolbox 'StreamPipes' as a representative for this. On the basis of a review of the literature, we selected 6 core types of spatial operator and implemented 33 basic spatial operators in 11 groups. These can be combined with the existing non-spatial operators for in-depth analysis of streaming data that involves spatial dimensions.

Keywords:

streaming data, spatial analysis, big data, complex event processing

1 Introduction

Data are at the core upon which each information system is built. In the last decade, the use of data streams has increased massively. In turn, this has led to new approaches to deal with this type of data and to new opportunities to gain insight into the physical world. This is especially true in the field of geography. While research in the field of streaming data has focused mainly on financial transactions or industrial production (Industry 4.0 or Smart Factory), spatially referenced data from remote sensing, in-situ sensing, social sensing and health sensing are becoming a new focus of academia and industry (see e.g. Graser and Widhalm (2018) or Yu et al. (2020)). This has led to the definition of the term GeoStreaming as: 'the ongoing effort in academia and industry to process, mine and analyze stream data with

geographic and spatial information' (Zhang et al. (2017, p. 1)). Another definition is proposed by Brandt and Grawunder (2018, p. 3): 'GeoStream is a data stream from spatio-temporal data. It is defined by three parts: (1) it is a data stream, (2) it has spatial information, and (3) it has temporal information.' They define three types of data which need to be handled by GeoStreams: point data at locations, data streams and evolving regions (e.g., weather phenomena).

These new opportunities come with additional challenges. Spatio-temporal data require special approaches to processing and analysis (see e.g. Yang et al. (2020)). Spatial databases as well as GIS were developed to deal with these challenges by using a batch processing approach. Streaming data, on the other hand are continuous but can be highly irregular, consisting of many different (small) events, which need to be gathered, processed, analysed and piped simultaneously as well as in real time. This is in contrast to the traditional IMAP (input, management, analysis and presentation) principle in the GIS world, where each of these steps is carried out sequentially. Their combination as a GeoStream requires solutions to be found for the existing challenges of spatio-temporal data within a continuous data stream, which can continue indefinitely.

In this paper, we address the research questions of how to handle GeoStreams efficiently and how to make them available to the wider GIS community. We identify and develop core operators for spatial analysis to be used in streaming systems. Complex event processing (CEP) is identified as the most promising concept from the field of streaming data processing. We base this decision on the research gaps identified by Brandt and Grawunder (2018, pp. 29–31): (1) Moving from static trajectories to continuous streams; (2) Connection of different types of GeoStreams; (3) Making sense of data, and (4) Extendable general-purpose systems.

2 State of the Art and Foundations

To achieve our goal in this paper, we need to investigate three different fields of research: Streaming data and CEP, GeoStreams and Spatial Operators. While some work exists which tries to combine these, this has been done mainly in relation to broader concepts as part of vision papers. Two examples for these broader concepts are PlanetSense, proposed by Thakur et al. (2015), which is a real-time streaming and spatio-temporal analytics platform for gathering geo-spatial intelligence from open source data, and BigGIS, presented by Wiener et al. (2016), which describes an overarching architecture in spatio-temporal big data. BigGIS envisions a system which integrates semantic information, streaming and batching in a continuous refinement process. Here, we focus on how to efficiently integrate and enhance state-of-the-art stream processing with spatial analysis.

2.1 Stream Processing and CEP

Stream processing describes the gathering, handling and use of continuous data flows (called data streams). Early works such as Terry et al. (1992) and Babu and Widom (2001) define data streams in contrast to traditional data, which they call persistent datasets. They demonstrate the key challenges with regard to queries on stream data: (1) The size of the data and therefore

its storage need are unbounded; (2) While performing queries, the query itself requires unbounded storage; (3) It is unclear how to deal with updates to the data regarding queries; (4) An exact answer to a query is impossible for reasons 1–3. Babu and Widom (2001) then refer to the concept of triggers, called 'event-condition-action rules', which enable a system to act upon defined or learned rules in data streams.

Since then, the field of stream processing has progressed, and it is now a thriving area of research. Many frameworks for handling and analysis of data streams such as Apache Storm[1], Apache Kafka[2] and Apache Flink[3] have been developed. However, to deal with the research gaps identified by Brandt and Grawunder (2018), we need an approach that not only handles the data but also analyses complex interactions and events in a continuous data stream. CEP provides the means to do this. Bruns and Dunkel (2015) describe CEP as a software technology for the dynamic analysis of big data in real time. It allows the analysis of interconnected events, e.g. in causal, temporal, spatial or various other relations, via the three core components of the architecture of a CEP, as described by Etzion and Niblett (2011): (1) an event producer, (2) an event consumer, and (3) an event processing agent (EPA). These are analogous to a data source such as an air-quality sensor, a data sink such as a database, and a processing component such as an analysis algorithm. An event can be an unmodified, raw event, or a derived event, e.g. created by processing through an EPA. A derived event can include modified parameters or new attributes. By combining several EPAs, an event processing Network (EPN) is created (see Bruns and Dunkel (2015)). This allows the handling of complex events and processing chains.

2.2 GeoStreams

In comparison to the general fields of stream processing and CEP, their combination as GeoStreams is a new research topic. It is, however, an important one (see Zhang et al. (2017)).

Early work in this area was presented by Huang and Zhang (2008). They based their approach on existing spatio-temporal databases and proposed extending these with a new data type. Huang and Zhang identified the same core challenges as those identified in early work on streaming data, e.g. by Babu and Widom (2001): '(1) it is difficult to decide how frequently the program should issue the queries to the spatial databases; (2) it is computationally expensive to find the latest data from all the historical data each time when the query is issued; (3) an integrated query optimization cannot be performed by the database system and kept transparent [for] the user' (Huang and Zhang (2008, p. 107)). They also identified a lack of support for spatial data in existing methods and approaches from the stream processing community, as 'previous work […] can only handle streaming point locations naively' (Huang and Zhang, 2008, p. 109). However, they define the necessary data types only broadly and do not elaborate on the spatial operators used or how they are chosen.

Galić (2016) and Galić et al. (2017) propose a framework for highly scalable spatio-temporal stream computing called MobyDick. This framework is based on existing approaches from the

[1] https://storm.apache.org/
[2] https://kafka.apache.org/
[3] https://flink.apache.org/

field of spatial databases; it uses relational snapshot queries, which are also called 'continuous queries' in the context of streaming data. Their own work is inspired by work in the field of mobility data and by the insight that state-of-the-art Data Stream Management Systems (DSMS) have inadequate spatial capabilities: 'However, spatio-temporal properties of both data streams and continuous queries have been disregarded, and most of the current DSMSs offer very rudimentary support for mobility data' (Galić et al., 2017, p. 3). Like Huang and Zhang (2008), they do not explain how they chose their spatial operators; they also focus on new data types for databases.

A recent definition of GeoStreams can be found in Brandt and Grawunder (2018, p. 5). In their extensive survey paper, they define a GeoStream as 'the intersection of two fields in computer and geography science: Data Stream Management and Geographic Information Science (GIS)'. Their work looks at 137 different publications, over 100 of which are from the field of computer science – either from the ACM or the IEEE. However, key studies in the literature, such as the dissertation by Whittier (2018), are in the field of spatial information science and geography. As well as the key challenges and research gaps in the field, Brandt and Grawunder also identify the evaluation of (spatial) predicates on GeoStreams as a key topic.

GeoStreams and their inherent potentials and challenges are also an ongoing topic in industry. Schmutz (2019), for example, combines geofences from databases with KSQL (Kafka SQL) for location analysis. He identifies problems with the asynchronous processing of long-running operations or the combination of different analyses. Other key players in industry also start with proprietary tools and products such as Esri Stream Event, IBM Streams or IBM Pairs.

2.3 Spatial Operators

To identify the spatial operators required, we first need to identify the available operators for spatial analysis. As one question is how to provide accessibility to GeoStreams to a broad userbase of the GIS community, we follow the approach of Brauner (2015) and investigate the spatial operators of widely used GIS. A quick search in the spatial analyst toolbox of Esri (2018 version), one of the best-known commercial vendors of GIS, results in more than 180 tools in 20 categories. Additional operators are available in their add-ons. While it would be possible to re-implement all these operators, we argue that not all are needed, or provide a benefit for stream processing, or are essential for most tasks. Instead we investigate the GIS literature of the last 25 years to provide the scientific basis for identifying the core spatial operators which are needed for stream processing and CEP.

Bailey and Gatrell (1995) provide a working and broadly accepted definition of 'spatial analysis' as the quantitative investigation of phenomena which are located in a geographical space. Two key functionalities which operators for spatial analysis have to perform are (1) The selection of existing data, and (2) the transformation of data. However, these two functionalities are still too broad.

Albrecht (1998) provides a 'universal framework' using a well-known classification of GIS operators (see Figure 1). While, as the author emphasizes, this is an approximation, it provides a strong basis for our selection. The six key classes and therefore the six functions required are: Search, Locational Analysis, Terrain Analysis, Distribution Neighbourhood, Spatial Analysis and Measurements. The first five can be subdivided into more specific subclasses. The concepts are broad, and similar classifications were made by Jones (1997), Longley et al. (2011) and Burrough et al. (2015). It can be argued that this 'universal framework' is too focused on the operators of commercial tools. But for our research question, these links to existing GIS make them more suitable, as they are familiar to users. Other approaches were also investigated, namely: the 10 core concepts of spatial information of Kuhn (2012); the algebraic approaches to specify operators provided by Scheider et al. (2016); the question-based approach of Scheider et al. (2019) in which they implement 8 tools which are well known to GIS users in a SPARQL extension called SPARQL_Constraint; the problem-oriented approach by Chrismann (2002). For further discussion about spatial operators, we refer the reader to chapter 4 of Brauner (2015).

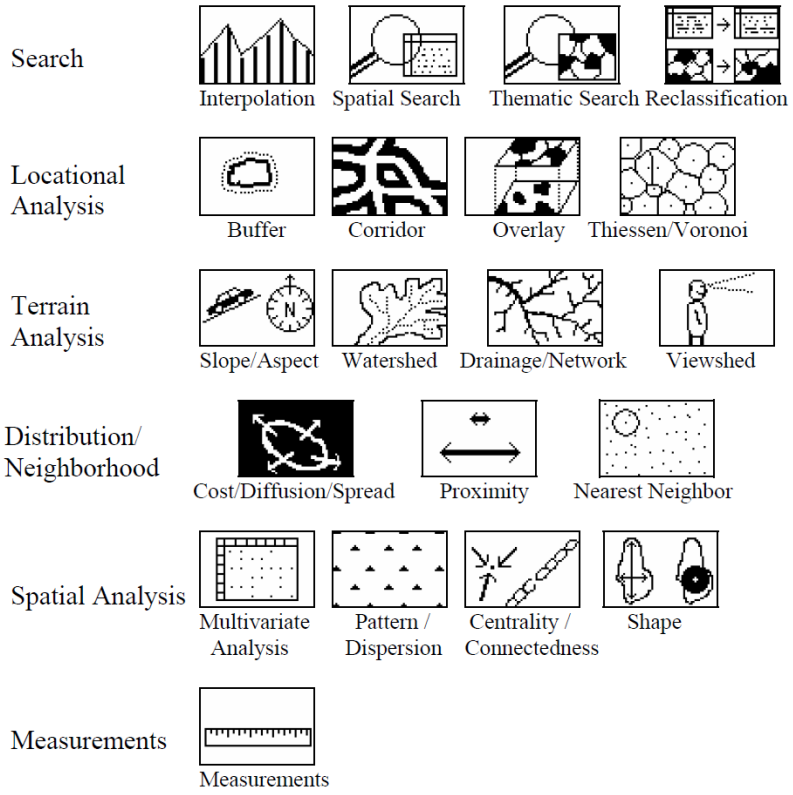

Figure 1: Classification of GIS-Operators by Albrecht (1998)

3 Spatial Operators for Streaming Data

In this section we will first describe the open source CEP tool StreamPipes (Riemer et al., 2015; Riemer, 2016; StreamPipes, 2020) which we used, second the operators selected and implemented, and finally two examples of analysis pipelines using the particular operators.

3.1 StreamPipes

StreamPipes, at its core, is a framework for the processing of streaming data in a big data environment – a CEP system in short (StreamPipes, 2020). This includes the import and handling of non-streaming data sources such as traditional databases, HDFS files, etc. It is designed to be a self-service analysis tool which allows complex analyses in a big data infrastructure without the need for technical expertise. StreamPipes can compute at least 54,000 events per second on a raspberry Pi (Zehnder et al., 2020), which was deemed acceptable for most use cases.

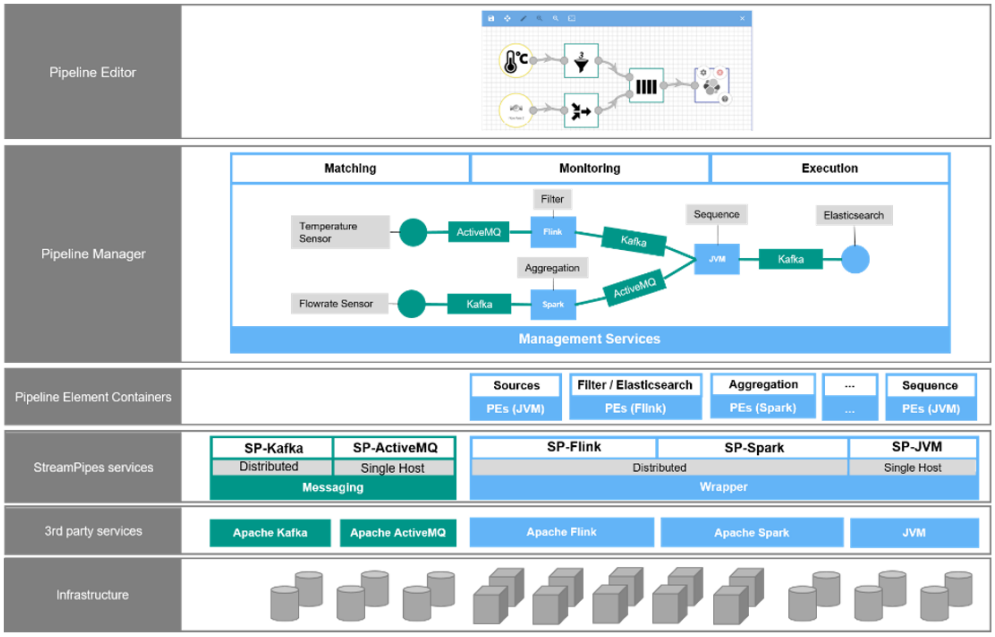

Figure 2: Overview of the StreamPipes architecture (StreamPipes, 2020)

StreamPipes uses a multi-layered technical architecture, shown in Figure 2. The user interacts only with the top layer, the pipeline editor. Here the different operators (called elements) are combined into an analysis pipeline. The user does not need to know how each component is connected to the underlying layers. However, semantic descriptors in the input and output of each element ensure that only valid data-processing pipelines can be created.

The pipeline manager manages the definition of the pipelines and their execution. It compares the semantic description of each element and ensures the validity of the pipeline. To execute a pipeline, the pipeline manager starts the Pipeline Element Containers, each of which contains a single element which has all relevant information. These elements in turn execute the code in an execution engine. Figure 3 shows an example of a meta-description element for a spatial event. It contains the core (spatial) attributes of the (spatial) position. The 'Schema' describes the different attributes of an event. The field 'Quality' defines different attributes, which are used to describe the quality inherent in the data. These attributes are dependent on the use event or analysis chain, so must be defined before the pipeline is executed. The 'Grounding' parameter describes how each event communicates with other events.

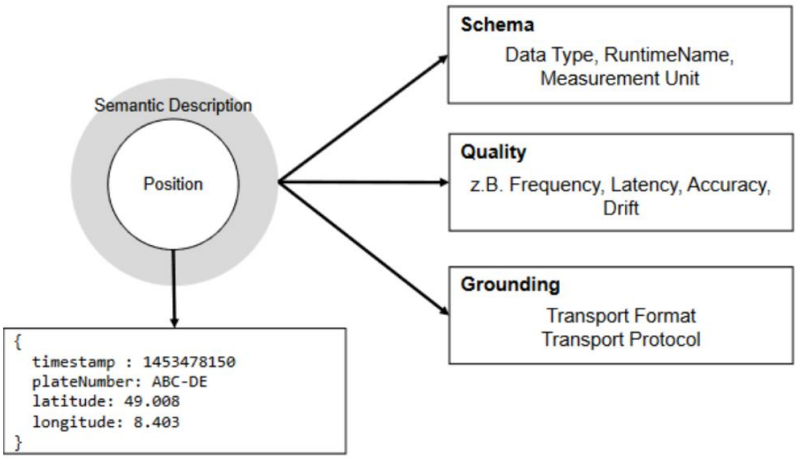

Figure 3: Description layer and data layer in StreamPipes for a geographical position (Riemer, 2016)

Finally, the last three layers in Figure 2 provide technical components such as virtual machines, third-party services, and the exact services which execute the element container. For a more in-depth view, we refer the reader to Riemer et al. (2015), Riemer (2016) and StreamPipes (2020).

We also need to discuss how implementation and integration of the spatial operators can be developed for StreamPipes. We use the StreamPipes SDK. This SDK allows the creation of new EPAs (see Section 2), which can be done using the Java programming language (Version 1.8.0_201 at the time of writing). We refer the interested reader to StreamPipes (2020) for more information and tutorials. A StreamPipes SDK consists of three different classes (see Figure 4). (1) The controller class defines the semantic model in the declairModel method, and the onInvocation method extracts the defined parameter for the algorithm class. (2) The parameter class therefore consists of the parameters from the controller class. (3) The algorithm class contains the algorithms for the data processing. Here, every parameter is initialized, e.g. through a database connection. The onEvent method then initializes every dynamic parameter to catch events and process these. Finally, in the onDetach method, every open connection is closed.

Figure 4: SDK class for a data processor

By using the StreamPipes SDK and its classes, we can freely define and implement any spatial operator needed for spatial analysis in a CEP system.

To summarize, we chose StreamPipes as it provides many advantages for our goal. While other tools would be feasible, the following key features needed for a spatial CEP system led us to our decision: (1) StreamPipes provides a native inclusion of different stream processing frameworks such as Apache Kafka, Apache Flink and Apache Spark; (2) It is built from the ground up with CEP in mind; (3) It is easy to modify as well as to create new pipeline elements with the StreamPipes SDK; (4) Each pipeline element can be created independently, but also combined with any other element, which allows for many different combinations in a pipeline; (5) It provides a simple drag-and-drop pipeline editor which reduces users' requisite technical know-how; (6) It is an open source tool under the Apache licence, which allows a wide range of fields and people to use it without restrictions, from academics and industry to the general public.

Additionally, a number of StreamPipes features are beneficial for a spatial CEP, and for future development of the system and the operators: (1) Since its inception, the system has been based on semantic technologies and concepts. This allows the easy combination of different operators. Each pipeline element describes its required input and the output it provides in a standardized semantic annotation. In addition, each dataset is transformed and annotated with the semantic information by the different EPAs. (2) The system uses JSON natively for the exchange of data. This allows an easy implementation of new data types, such as GeoJSON, and facilitates the integration in existing platforms. This means that: (3) It is interoperable at its core and can combine different data sources and types. (4) It already includes some simple, WGS84-based, spatial coordinates if they are point data. (5) It has advanced capabilities for time-series analyses as well as machine learning algorithms for stream processing. (6) It was granted Apache incubator status in 2019. This facilitates its high visibility and an active core team for the development of future features, as well as the integration of the operators developed.

3.2 Operators Implemented

For the selection of the spatial operators, we can now build on the foundations of StreamPipes, the challenges defined in the GeoStreams literature, and the classification of (spatial) operators for spatial analysis as set out in the existing GIS literature.

First, as discussed in the literature, we need to define and select a suitable spatial data model to use with the spatial operators in the CEP. As is well known, e.g. from Longley et al. (2011), there are two key models for spatial data: the entity model and the field model. Additionally, the data structure itself has to be modelled, either as vector or as raster data. For stream processing, we identified the entity and vector model as the most efficient, because the computation time is shorter, and storage needs of vector data are smaller compared to raster for stream processing. While raster is highly efficient in distributed computing and benefits from its similarity to image analysis, these benefits are mostly irrelevant here. To process and communicate data between the different CEP components, interoperability must be ensured (i.e. the autonomous interaction of software components by using standardized data formats; Andrae (2013)). Here, two further considerations come into play. First, for overall interoperability, we go to the OGC and use the standard ISO 9007 as well as the reduced simple feature model (OGC, 2019). Second, we need to define a semantic standard description of the data, an ontology. StreamPipes uses the basic geo vocabulary[4], but is missing an ontology domain for the geometry. Here, we propose a simple, provisional ontology which uses WKT. This allows us to minimize the computational workload. An example of code to define an event is shown in Figure 5. Alternatively, more extensive ontologies can be found in, for example, Bucher et al. (2017) or Scheider and Tomko (2016).

```
.requiredStream(StreamRequirementsBuilder
      .create()
      .requiredPropertyWithUnaryMapping(
          EpRequirements.domainPropertyReq("http://www.opengis.net/ont/geosparql#Geometry"),
          Labels.withId(GEOMETRY_KEY), PropertyScope.MEASUREMENT_PROPERTY
      )
      .requiredPropertyWithUnaryMapping(
          EpRequirements.domainPropertyReq("http://data.ign.fr/def/ignf#CartesianCS"),
          Labels.withId(EPSG_KEY), PropertyScope.MEASUREMENT_PROPERTY
      )
      .build()
)
```

Figure 5: Example of code for the provisional ontology

Not all spatial operators identified in the previous section are essential for our spatial CEP. From our point of view, the key operators are those that enable the analyses carried out most commonly by a GIS user. We therefore exclude statistical analysis functions and network analysis. Some of these common operations are already performed by existing EPAs, while others have specific requirements, such as the need for specific parametrizations for geostatistical models. Compared to other operators, the ones for such analyses are therefore of relatively little benefit for the initial system. Instead, additional operators such as the projection of coordinate systems, the integration of data sources and data sinks, as well as their use as static information like a geofence, provide a higher benefit and need to be implemented first. While these operators are not seen as special in traditional GIS, they need special care in a streaming environment. Additionally, adjustments to geometries, such as the refinement of lines into their subgroups, are needed. While we use vector data as the main data model, some

[4] https://www.w3.org/2003/01/geo/

special data usage scenarios and operations do exist which necessitate the implementation of raster operations. Galić (2016) and Galić et al. (2017) focus on this topic in their work.

The considerations just outlined lead us to 33 basic spatial operators, shown in Figure 5. Each operator is implemented as an EPA in StreamPipes. Therefore, complex operators can be realized either as a single EPA or as a combination of several EPAs. Following Brauner (2015, p. 38) that operators should be 'generally available, yet their discovery and usability is hampered', we divided the operators into eleven groups, for ease use: (1) Base Operators, (2) Change on Geometry Operators, (3) Geofence Operators, (4) Measure Operators, (5) Operators on Window Functions, (6) Thematic Operators, (7) Topology Operators, (8) Calculate from Raster Operators, (9) Routing Operators, (10) Data Operators and (11) Derived Geometry Operators. Some of the specific operations, particularly the raster operations and the routing operators, are implemented as a proof of concept and were required for the implementation of concrete realistic examples.

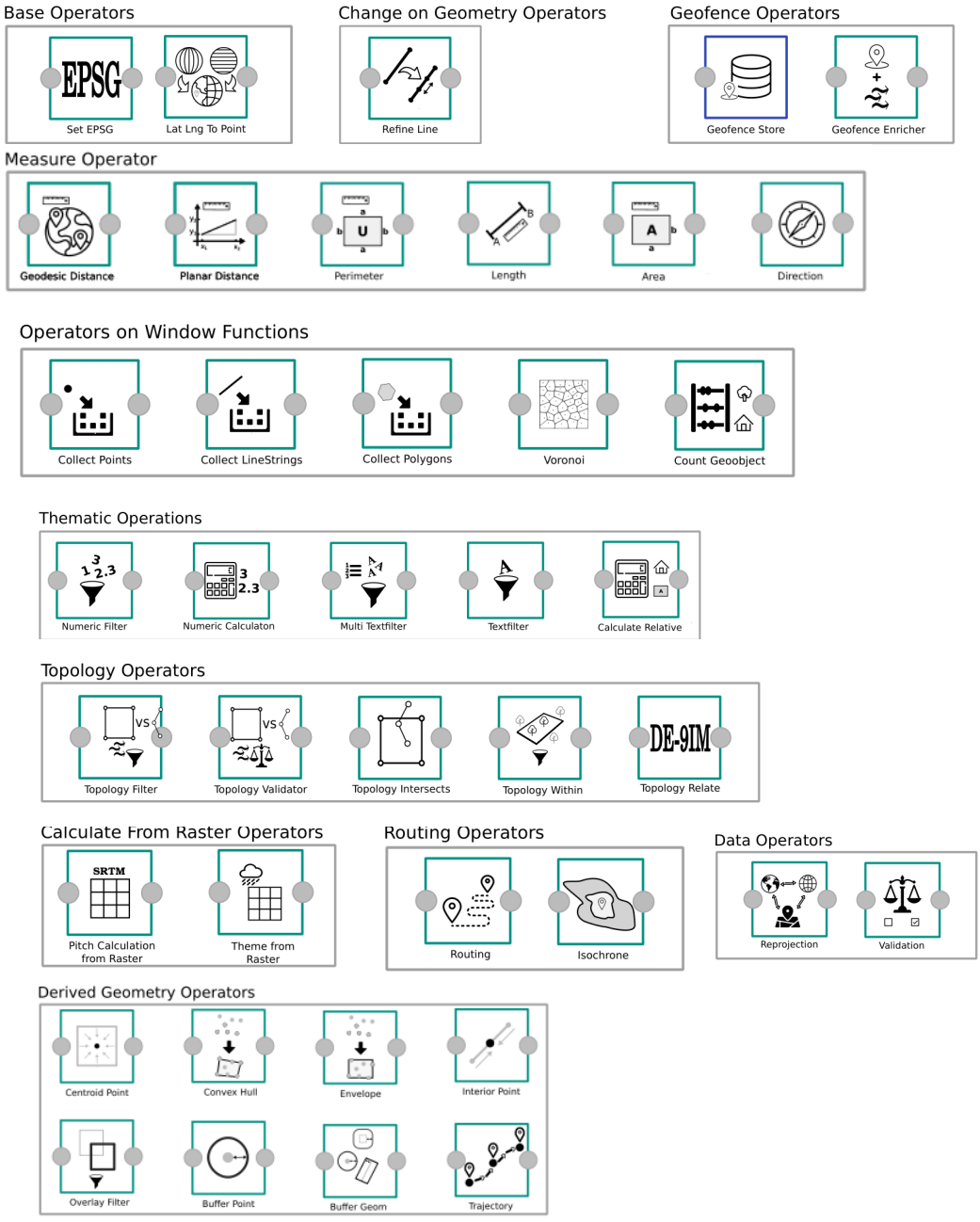

Figure 6: The basic spatial operators implemented

Which operator groups to include was by and large obvious, notably groups 1, 2, 4, 6, 7 and 10. We selected geofences as a key operator, as this operation is essential to detect the entering/leaving of moving objects in an area, which is often seen as a central functionality in GeoStreams. Window functions were included as the most efficient option to compute the number of objects in a time window, and for the temporary storage of events. This provides the basis for future extensions for statistical learning methods or machine learning algorithms. Raster operations are included for future extensions and to allow the addition of further data sources, such as remote sensing data or weather forecasts, as these are most often available directly only as raster data. Routing operators are included to enable flexible route planning or isochrones dependent on the changing circumstances. As any new event can, for example, change the road situation, it is important to be able to analyse the impact on other objects, such as an ambulance, and to react to the new situation. Finally, the derived geometry operators allow the most common operations between different data points as well as the plotting of the trajectory. The trajectory was chosen because the live movement of a point is a key benefit of a streaming approach compared to traditional GIS. We also used the existing operators of StreamPipes in addition to our selection.

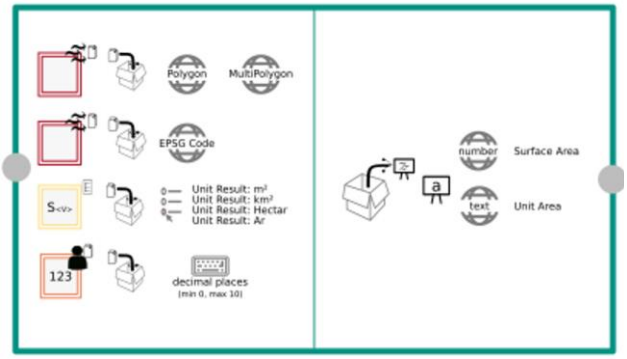

Figure 7: Example for (polygon) area calculator

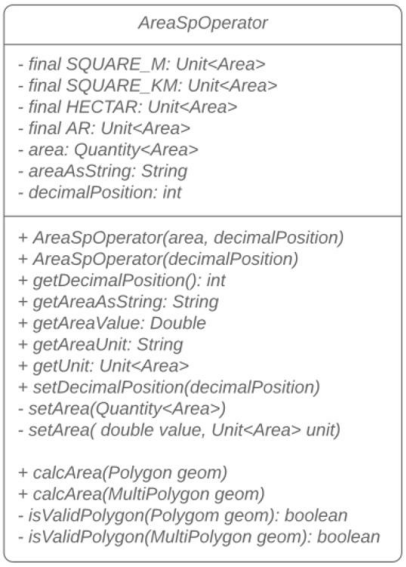

Figure 8: UML class diagramm

To show how each operator is implemented and what the required in- and outputs are, we use the (polygon) area calculator as an example. This operator allows the computation of the area of an object. Each operator can be shown as its SDK class with each operator or as the EPA in StreamPipes from a conceptual point of view. Figure 7 shows the different methods in the operator; Figure 8 shows the interaction of the classes within the StreamPipes SDK, as described in Section 3. The JTS geometry classes were used for the getLength and getPerimeter methods. As inputs, a polygon or multipolygon as well as an EPSG code are required. Additional required inputs are the accuracy, defined as the number of digits following the

decimal point, and the unit result. The operator then returns the size of the area as a number, and the chosen unit result as a text. If a MultiPolygon is used for the input, the sum of all geometries is returned.

For the implementation of the operators and the data models, existing software libraries can be used to reduce complexity and the time spent on details. We use JTS (Davis, 2018) and Apache SIS as our libraries.

3.3 Example Pipelines

We present two simple pipelines to show the possibilities of our approach and to provide a simple, qualitative evaluation. These examples allow a starting point for future analyses. Open data is used in both pipelines.

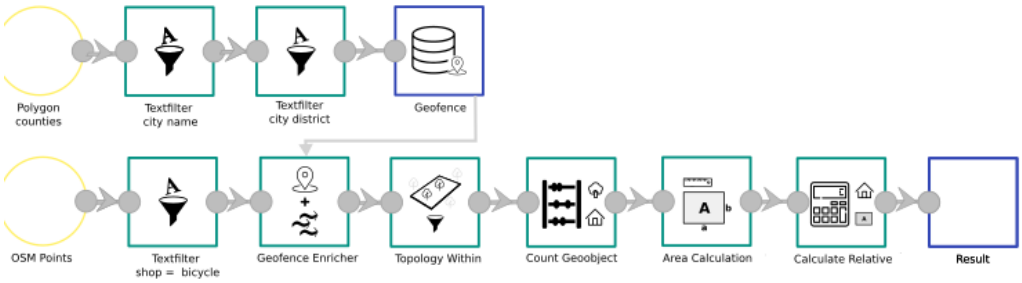

Figure 9: Computing the number of specific objects per area size in a polygon defined by a geofence

The first pipeline shown in Figure 9 is a simple computation of the number of bicycle shops per km^2 in a polygon defined by a geofence. This is a typical, simple computation, which can be done in a standard GIS. It illustrates how existing functions can be used, as well as future possibilities. To perform the analysis, two data streams are defined. The geofence is chosen by filtering existing polygons based on their name. In this example, the district of Karlsruhe is selected from a list of polygons of all districts in Germany. This geofence is stored in a database and combined with the second stream in the pipeline. For the second stream, first all OSM points are included and filtered to include only bicycle shops. These are combined and then filtered by the geofence using the 'topology within' operator. Finally, all geo objects are counted, the area of the geofence is calculated, and the number per km^2 is calculated. The result can then be further used, visualized and/or stored. To extend this example to a GeoStream, we simply need to change the input of the first and/or second stream to a real data stream, e.g. moving bicycles or cars. This then returns the density of vehicles in an area in real time, in order, for example, to monitor the risks of traffic jams. The usability of our chosen tools allows this operation without changing any operator in the pipeline itself.

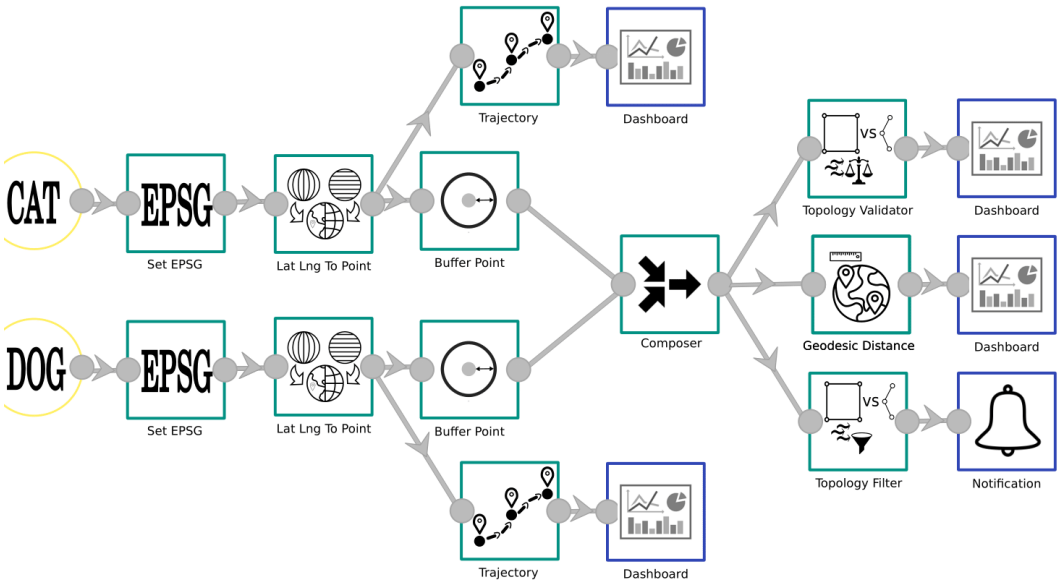

Figure 10: Computing distance and convergence of two moving points

A more interesting pipeline, which goes beyond the typical capabilities of a GIS, is shown in Figure 10. Here, we are interested in the live distance between two moving objects, the trajectory of each, and a warning if the objects risk converging. Two moving objects, a cat and a dog equipped with GPS sensors, were used as input streams. For each stream, the coordinate system was defined and transformed to a point geometry. The trajectories of the streams were first visualized in a dashboard as live data. Next, for each object, a buffer area was defined; both buffer areas were then merged into one stream for synchronization. Finally, their proximity was shown as 'TRUE' (close) or 'FALSE' (not close), the live (geodesic) distance between both points was visualized, and a notification was defined if the status was 'TRUE'. All of these operations are performed simultaneously. A simple, qualitative performance evaluation was done with 3,000 objects as input using an artificial data stream. No output lag could be detected when all streams were running.

The two examples show that combining the different spatial operators with existing StreamPipes operators allows for a broad range of analyses. Further examples of pipelines, together with the datasets and operators, can be found at https://StreamPipes.apache.org/ (StreamPipes, 2020).

4 Conclusion and Future Work

In this paper, we developed and presented spatial operators for a state-of-the-art, open source CEP tool, StreamPipes. These operators allow the execution of complex spatio-temporal analyses on streaming data for complex events. This addresses the research gaps identified by Brandt and Grawunder (2018), providing an easy-to-use tool which is modular and readily

extendable. By using the semantic properties of StreamPipes in the operators we implement, we can easily connect different types of GeoStreams. The existing machine learning tools for CEP can leverage spatio-temporal analyses in combination with the spatial operators which we developed. Our operators for spatial analysis are based on the existing literature in the fields of stream processing, GeoStreams and classical GIS literature. We illustrated the ease of use as well as feasibility of our approach in two examples of pipelines for spatial analysis. By using open data and open source software, the examples can be easily reproduced and extended.

Regarding our research question of how to handle GeoStreams to allow the wider GIS community to be able to work with them, we have shown that by using CEP together with simple, commonly used spatial operators, we can handle GeoStreams efficiently and make them simple and intuitive to use.

The purpose of this paper was to promote a new category of software system / software functionality, namely spatial CEP systems for processing GeoStreams as a novel and upcoming paradigm. We have shown an example implementation with a focus on efficiency, extensibility and user-friendliness. Our examples should illustrate possible applications and the usefulness of the approach; they were not used for a crisp and formal evaluation of the system. Extensive evaluation of the strengths and weaknesses of the approach should be the subject of future work. In general terms, the evaluation dimensions should include: (i) expressiveness of the approach in order to solve theoretically and practically relevant problems; (ii) efficiency and scalability of the runtime; (iii) formal tests of usability for end-users with little prior geo(-ICT) knowledge; (iv) efficiency of the modelling environment for end-users (with aspects such as modular reuse of operators, pipelines and sub-pipelines, or ease of implementing new operators with the StreamPipes SDK). Initial studies and experiments regarding (i) and (iii) have been undertaken within the WEKOVI research project, but much more systematic analysis is certainly required.

Another key limitation of the work presented here is its scope. We provide a foundation for more extensive GeoStream analysis, but many other interesting operations have yet to be implemented. For future work we therefore identify two focus topics: spatial data mining, and asynchronous analysis. Spatial data mining encompasses methods from statistical analysis (e.g. kriging and clustering), and network analysis. By carrying out these approaches in GeoStreams, additional challenges emerge, solving which could lead to new analyses and insights. The problem of analysing asynchronous data streams was discussed by Schmutz (2019), and a potential solution was proposed independently by Whittier (2018). This problem is inherent in all stream processing and is a feature that is missing in StreamPipes. Using the well-known technique of a petri net (Petri, 1966) could be an alternative solution to Whittier (2018). In addition, a more thorough and extensive evaluation setup is planned.

This work further contributes by providing non-technical researchers with a more approachable way to perform spatial analyses on streaming data. Using the drag-and-drop possibility of StreamPipes, complex analysis pipelines can be created easily. By extending it with spatial operators, this open source tool will provide additional benefits for various fields, but it will be a boon in particular for geographical researchers. We hope that this will provide a starting point for future researchers and practitioners using GeoStreaming and analyses on GeoStreams, opening up new (research) questions.

Acknowledgements

We would like to thank the reviewers for their feedback, comments and literature recommendations. They helped to improve, clarify and focus this work, especially in regard to the examples used. This work is based on the mFUND project WeKoVi and the NAIADES project, which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 820985.

References

Albrecht, J. (1998). Universal analytical GIS operations: a task-oriented systematization of data structure-independent GIS functionality. Dissertation, Hochschule Vechta.

Andrae, C. (2013). *Simple Features - Praxisnahe Standards für einfache Geoobjekte in Datenbanken und GIS*. Wichmann, Heidelberg.

Babu, S., & Widom, J. (2001). Continuous queries over data streams. *ACM Sigmod Record*, 30(3), 109-120.

Bailey, T. & Gatrell, A. (1995). *Interactive spatial data analysis*. Longman Scientific & Technical.

Brandt, T., & Grawunder, M. (2018). GeoStreams: A Survey. *ACM Computing Surveys (CSUR)*, 51(3), 1-37.

Brauner, J. (2015). Formalizations for geooperators-geoprocessing in spatial data infrastructures. Dissertation, Technische Universität Dresden.

Bruns, R. & Dunkel, J. (2015). *Complex Event Processing im Überblick*, 9–17. Springer Fachmedien Wiesbaden, Wiesbaden.

Bucher, D., Scheider, S., & Raubal, M. (2017, November). A model and framework for matching complementary spatio-temporal needs. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (pp. 1-4).

Burrough, P. A., McDonnell, R., McDonnell, R. A. & Lloyd, C. D. (2015*). Principles of geographical information systems*. 3rd edn. Oxford University Press.

Davis, M. (2018). JTS Topology Suite - Version 1.16. https://projects.eclipse.org/projects/locationtech.jts

Chrisman, N., (2002). *Exploring Geographic Information Systems* 2nd edn. New York: John Wiley & Sons. ISBN 0–471–31425–0. https://locationtech.org/projects/technology.jts, licence Eclipse Public License 1.0.

Etzion, O. & Niblett, P. (2011). *Event processing in action*. Software Engineering. Manning, Stamford.

Galić, Z. (2016). Spatio-temporal data streams and big data paradigm. In *Spatio-Temporal Data Streams* (pp. 47-69). Springer, New York.

Galić, Z., Mešković, E., & Osmanović, D. (2017). Distributed processing of big mobility data as spatio-temporal data streams. *Geoinformatica*, 21(2), 263-291.

Graser, A., & Widhalm, P. (2018). Modelling massive AIS streams with quad trees and Gaussian Mixtures. In *21st Int. Conf. Geogr. Inf. Sci.* (AGILE 2018) (pp. 1-5).

Huang, Y., & Zhang, C. (2008, September). New data types and operations to support geo-streams. In *International Conference on Geographic Information Science* (pp. 106-118). Springer, Berlin, Heidelberg.

Jones, C. (1997). *Geographical Information Systems and Computer Cartography*. Longman.

Kuhn, W. (2012). Core concepts of spatial information for transdisciplinary research. *International Journal of Geographical Information Science*, 26(12), 2267-2276.

Longley, P. A., Goodchild, M. F., Maguire, D. J. & W., R. D. (2011). *Geographic Information Systems and Science*. 3rd edn. John Wiley & Sons, New York.

OGC (2019). OGC® Standards and Supporting Documents. Online.

https://opengeospatial.org/standards.

Petri, C. A. (1966). *Communication with automata.*

Riemer, D. (2016). Methods and Tools for Management of Distributed Event Processing Applications. Dissertation, Karlsruher Institut für Technologie (KIT).

Riemer, D., Kaulfersch, F., Hutmacher, R. & Stojanovic, L. (2015). StreamPipes: Solving the Challenge with Semantic Stream Processing Pipelines. In: *Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems, DEBS '15*, 330–331. ACM, New York.

Scheider, S., Ballatore, A., & Lemmens, R. (2019*).* Finding and sharing GIS methods based on the questions they answer. *International Journal of Digital Earth*, 12(5), 594-613.

Scheider, S., Gräler, B., Pebesma, E., & Stasch, C. (2016). Modeling spatiotemporal information generation. *International Journal of Geographical Information Science*, 30(10), 1980-2008.

Scheider, S., & Tomko, M. (2016). Knowing whether spatio-temporal analysis procedures are applicable to datasets. *Formal Ontology in Information Systems* 283, pp. 67-80. IOS Press.

Schmutz, G. (2019). Location Analytics Real-Time Geofencing using Kafka. DOAG Conference 2019.

StreamPipes (2020). https://StreamPipes.apache.org/ .

Terry, D., Goldberg, D., Nichols, D., & Oki, B. (1992). Continuous queries over append-only databases. *ACM Sigmod Record*, 21(2), 321-330.

Thakur, G. S., Bhaduri, B. L., Piburn, J. O., Sims, K. M., Stewart, R. N., & Urban, M. L. (2015). PlanetSense: a real-time streaming and spatio-temporal analytics platform for gathering geo-spatial intelligence from open source data. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems* (pp. 1-4).

Yang, C., Clarke, K., Shekhar, S., & Tao, C. V. (2020). *Big Spatiotemporal Data Analytics: A research and innovation frontier.*

Yu, M., Bambacus, M., Cervone, G., Clarke, K., Duffy, D., Huang, Q., Li, J., Li, W., Li, Z., Liu, Q., Resch, B., Yang, J., Yang, C. (2020). Spatiotemporal event detection: a review. International Journal of Digital Earth, 1-27.

Whittier, J. (2018). Towards an Efficient, Scalable Stream Query Operator Framework for Representing and Analyzing Continuous Fields. University of Maine, PhD Thesis. https://digitalcommons.library.umaine.edu/etd/2927/

Wiener, P., Stein, M., Seebacher, D., Bruns, J., Frank, M., Simko, V., Zander, S., & Nimis, J. (2016). BigGIS: A continuous refinement approach to master heterogeneity and uncertainty in spatio-temporal big data (vision paper). In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (pp. 1-4).

Zehnder, P., Wiener, P., Straub, T., & Riemer, D. (2020). StreamPipes Connect: Semantics-Based Edge Adapters for the IIoT. In *European Semantic Web Conference* (pp. 665-680). Springer, Cham.

Zhang, C., Banaei-Kashani, F., & Hendawi, A. (2017). IWGS 2016 workshop report: The 7th ACM SIGSPATIAL International Workshop on GeoStreaming: San Francisco, CA, USA-October 31, 2016. *SIGSPATIAL Special*, 8(3), 12-13.