Towards Spatial Ontology-Mediated Query Answering over Mobility Streams

Thomas Eiter[1], Josiane Xavier Parreira[2], and Patrik Schneider[1,2]

[1] Vienna University of Technology, Vienna, Austria
[2] Siemens AG Österreich, Vienna, Austria

Abstract. The development of (semi)-autonomous vehicles, communication between vehicles and infrastructure (V2X) will aid to improve road safety by identifying dangerous traffic scenes. A key to this is the Local Dynamic Map (LDM), which acts as an integration platform for static, semi-static, and dynamic information about traffic in a geographical context. At present, the LDM approach is purely database-oriented with simple query capabilities, while an elaborate domain model as captured by an ontology and queries over data streams that allow for semantic concepts and spatial relationships are still missing. To fill this gap, we present an approach in the context of ontology-mediated query answering that features conjunctive queries over DL-Lite$_A$ ontologies allowing spatial relations and window operators over streams having a pulse. Query evaluation is possible by rewriting to ordinary DL-Lite$_A$ that involves transformation of spatial relations and epistemic aggregate queries.

1 Introduction

The development of (semi)-autonomous vehicles needs extensive communication between vehicles and the infrastructure, which is covered by Cooperative Intelligent Transport Systems (ITS). These systems collect temporal data (e.g., traffic light signal phases) and geospatial data (e.g., GPS positions), which is exchanged by vehicle-to-vehicle, vehicle-to-infrastructure, and combined communications (V2X). V2X allows to improve road safety by processing and understanding traffic scenes, e.g., red light violation, thad could lead to accidents. A key technology for this is the Local Dynamic Map (LDM) [2], which acts as an integration platform for static, semi-static, and dynamic information in a geographical context. Current approaches for an LDM are purely database-oriented with simple query capabilities. An elaborate domain model, captured by a mobility ontology, and extended queries over data streams allowing spatial relations are still missing.

Our aim is to enable spatial-stream conjunctive queries (CQ) over a semantically enriched LDM for safety applications, such as detection of red light violations on complex intersections. To realize spatial query answering (QA) over mobility streams, spatial and streaming data must be lifted to the setting of ontology-mediated QA with the frequently used ontology language DL-Lite$_A$. However, bridging the gap between stream processing and ontology-mediated QA is not straightforward, as the semantics of DL-Lite$_A$ must be extended with spatial relations and stream queries using window operators. For this, we build on the work on spatial QA in [13] and extend ontology-mediated QA with epistemic aggregate queries (EAQ) [11] to detemporalize the streams. The extension preserves first-order rewritability, which allows us to evaluate a CQ with spatial atoms over a stream RDBMS. Our contributions are summarized as follows:

- We outline the field of V2X integration using LDMs in the mobility context (Sec. 2);

- we introduce a data model and query language suited for mobility streams (Sec. 3,4);
- we present a spatial-stream QA approach for DL-Lite$_A$ defining its semantics and showing the preservation of FO-rewritability. The QA approach is based on CQ over DL-Lite$_A$ ontologies, which combines window operators over streams having a pulse and spatial relations over spatial objects (Sec. 5);
- we provide a technique for query rewriting taking the above into account. For query evaluation, we extend and apply the known techniques of (a) transformation of spatial objects resp. relations into ordinary DL-Lite$_A$; (b) epistemic aggregate queries, e.g., average, for a "detemporalization" of the streams; and (c) provide a technique for query rewriting with EAQs (Sec. 6).

In Sec. 7 and 8, we describe related work and conclude with future work.

2 V2X Integration using Local Dynamic Maps

The base communication technologies (i.e., the IEEE 802.11p standard) allow wireless access in vehicular environments, which enables messaging between vehicles themselves and the infrastructure, called V2X communication. Different types of messages are broadcast every 100ms by traffic participants and will inform others about their current state such as position, speed, vehicle type; the detailed topology of an intersection, including its lanes and their connections; the projected signal phases (e.g., green) for each lane; and information on specific events like road works in a designated area [2].

Local Dynamic Maps. As a comprehensive integration effort of V2X messages, the SAFESPOT project [2] introduced the concept of an LDM as an integration platform to combine static geographic information system (GIS) maps with dynamic environmental objects (e.g., vehicles, pedestrians). The integration was motivated by advanced safety applications as red light violation, which need an "overall" picture of the traffic environment. The LDM consists of the following four layers (see Fig. 1):

- *Permanent static*: the first layer contains static information obtained from GIS maps and includes roads, intersections, and points-of-interest;
- *Transient static*: the second layer extends the static map by detailed local traffic informations such as fixed ITS stations, landmarks, and intersection features like lanes;
- *Transient dynamic*: the third layer contains temporary regional information like weather, road or traffic conditions (e.g., traffic jams), and traffic light signal phases;
- *Highly dynamic*: The fourth layer contains dynamic information of road users detected by V2X messages, in-vehicle sensors like the GPS module.

Current research (e.g., [17], [20], [21]) on the LDM architecture identified that it can be built on top of a spatial RDBMS enhanced with streaming capabilities. As recognized by [17], an LDM should be represented by a world model, world objects, and data sinks on the streamed input. However, an elaborate domain model, captured by an LDM ontology, and extended queries over data streams allowing spatial relations are still missing in current approaches. The ontology represents an integration schema, which we modeled as a DL-Lite$_A$ LDM ontology capturing the different layers of an LDM (cf. [12]).

Safety Applications on Intersections. "Road intersection safety" is an important application for improving road safety [2]. Intersections are the most complex environments and need special attention, where hazardous situations like *obstructed view* or *red-light violation* might lead to accidents. We take them as a motivation and running example.

Example 1. The following query detects red-light violations on intersections by searching for vehicles y with speed above 30km/h on lanes x whose signals will turn red in 8s:

$$q_1(x,y) : LaneIn(x) \wedge hasLocation(x,u) \wedge intersects(u,v) \wedge pos_{(line,\ 4s)}(y,v) \\ \wedge\ Vehicle(y) \wedge speed_{(avg,\ 4s)}(y,r) \wedge (r > 30) \wedge\ isManaged(x,z) \\ \wedge\ SignalGroup(z) \wedge hasState_{(first,\ -8s)}(z, Stop)$$

Query q_1 exhibits the different dimensions which need to be combined: (a) $Vehicle(y)$ and $isManaged(x,z)$ are ontology atoms, which have to be unfolded in respect to the ITS domain ontology; (b) $intersects(u,v)$ and $hasLocation(x,u)$ are spatial atoms, where the first checks spatial intersection and the second the assignment of a geometry to an object; (c) $speed_{(Avg\ 2s)}(y,v)$ defines a window operator that aggregates the average speed of the vehicles over the stream and $hasState_{(first,\ -8s)}$ gives us the next upcoming traffic light state.

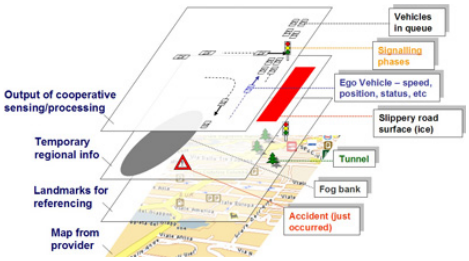

Fig. 1: The four layers of a LDM [2]

3 Streams, Pulses, and Spatial Databases

We now introduce the data model and sources which are used in spatial-stream QA.

Streams and Pulses. Our data model is *point-based* (vs. *interval-based*) and captures the *valid time* (vs. *transaction time*) saying that some data item is valid at that time point. We extend this validity of time, and say that a data item is valid *from its time point until the next data item is added* to the stream. To capture streaming data, we introduce the *timeline* $\mathbb{T}$, which is a *closed* interval of $(\mathbb{N}, \leq)$. A (data) *stream* is a triple $F = (\mathbb{T}, v, P)$, where $\mathbb{T}$ is a timeline, $v : \mathbb{T} \to \langle \mathcal{F}, \mathcal{S}_\mathcal{F} \rangle$ is a function that assigns to each element of $\mathbb{T}$, called *timestamp* (or time point), data items (called *membership assertions*) of $\langle \mathcal{F}, \mathcal{S}_\mathcal{F} \rangle$, where $\mathcal{F}$ (resp. $\mathcal{S}_\mathcal{F}$) is a *stream (resp. spatial with streams) database*, and P is an integer called *pulse* defining the general interval of consecutive data items on the timeline (cf. [6,18]). A pulse generates a stream of data items with the frequency derived from the interval length. We always have a *main pulse* $P_\mathbb{T}$ with a fixed interval length (usually 1) that defines the lowest granularity of the validity of data items. The pulse also aligns the data items, which arrive asynchronously in the database, to the timeline.

Extending [18], we allow additional *larger pulses* that generate streams with a lower frequency allowing larger intervals. Larger pulses also imply that their generated data items are valid longer than items from the main pulse, thus allowing us to resize the window size of a query and perform optimizations such as caching. Furthermore, pull-based queries are executed at any single time point i denoted as $\mathbb{T}_i$. Push-based queries are evaluated asynchronously where the lowest granularity is given by $P_\mathbb{T}$.

Example 2. For the timeline $\mathbb{T} = [0, 100]$, we have the stream $F_1 = (\mathbb{T}, v, 1)$ of vehicle positions and speed at the assigned time points $v(0) = \{speed(c_1, 30),\ pos(c_1, (5,5)),\ speed(b_1, 10),\ pos(b_1, (1,1))\}$, $v(1) = \{speed(c_1, 29),\ pos(c_1, (6,5))\ \ speed(b_1, 5),\ pos(b_1, (2,1))\}$, and $v(2) = \{speed(c_1, 34),\ pos(c_1, (7,5))\}$ for the individuals c_1 and b_1. A second "slower" stream $F_2 = (\mathbb{T}, v, 5)$ captures the next signal state of a traffic light:

$v(0) = \{hasState(t_1, Red)\}$ and $v(5) = \{hasState(t_1, Green)\}$. As F_2 has a pulse of $p = 5$, we know $v(4) = \emptyset$ but under an alternative semantics with an *inertia assumption*, we could conclude $v'(4) = \{hasState(t_1, Red)\}$. Further, the static ABox contains the assertions $Car(c_1)$, $Bike(b_1)$, and $SignalGroup(t_1)$.

Spatial Databases and Topological Relations. We recall the essential idea based on *Point-Set Topological Relations* (see [13]). Spatial relations are defined via pure set theoretic operations based on $P_E \subseteq \mathbb{R}^2$ of all points in the plane. An *admissible geometry* $g(s)$ is a sequence $p = (p_1, \ldots, p_n)$ of points over P_F, where $P_F \subseteq P_E$. We define a *spatial database* over Γ_S as a pair $\mathcal{S} = (P_F, g)$ of a point set P_F and a mapping $g : \Gamma_S \to \bigcup_{i \geq 1} {P_F}^i$, where Γ_S is a set of spatial objects. The *extent* of a geometry p (full point set) is given by the function $points(p)$ as a (possibly infinite) subset of P_E. For a spatial object s, we let $g(s)$ be its geometry and $points(s) := points(g(s))$. For our KB, we consider the following types of admissible geometries p over P_F, and let $P_E = \bigcup_{s \in \Gamma_S} points(s)$ (see [13] for more):

- *points* are the sequences $p = (p_1)$, where $points(p_1) = \{p_1\}$;
- *line segments* are sequences $p = (p_1, p_2)$, and $points(p) = \{\alpha p_1 + (1 - \alpha)p_2 \,|\, \alpha \in \mathbb{R}, 0 \leq \alpha \leq 1\}$;

We use $points$ to evaluate the spatial relations of two spatial objects via their respective geometries and define the relations in terms of *pure set operations* (see [13] for more):

- $Contains(x, y)$: $points(y) \subseteq points(x)$, $Intersect(x, y)$: $points(x) \cap points(y) \neq \emptyset$.

A spatial relation $S(s, s')$ with $s, s' \in \Gamma_S$ holds on a spatial database $\mathcal{S}$, written $\mathcal{S} \models S(s, s')$, if $S(g(s), g(s'))$ evaluates to true. Relative to $points$, this is easily captured by a first-order (FO) formula over $(\mathbb{R}^2, \leq)$, and with regard to geo-spatial RDBMS trivially FO expressible and rewritable into FO queries.

4 Syntax and Semantics of DL-Lite$_A$ (S,F)

We start from previous work in [13], which introduced spatial CQ answering for DL-Lite$_A$, and lift the semantics from the spatial DL-Lite$_A$ KB to the spatial-stream KB.

Syntax and Semantics of DL-Lite$_A$. We consider a vocabulary of individual names Γ_I, domain values Γ_V (e.g., $\mathbb{N}$), and spatial object Γ_S. Given atomic concepts A, atomic roles P, and atomic attributes E, we define (a) *basic* concepts B, and *basic roles* Q, and (b) *complex* concepts C and *complex* role expressions R, *complex attributes* V, *basic value-domains* E (range of attributes), and value-domain expressions D:

$$\begin{array}{llll} B & ::= & A \mid \exists Q \mid \delta(U_C) & C ::= \top_C \mid B \mid \neg B \mid \exists Q.C' \\ E & ::= & \rho(U_C) & D ::= \top_D \mid D_1 \mid \ldots \mid D_n \\ Q & ::= & P \mid P^- & R ::= Q \mid \neg Q \qquad V ::= U \mid \neg U \end{array}$$

where (c) P^- is the *inverse* of P, $\top_D$ is the *universal* value-domain and $\top_C$ is the *universal* concept; and U_C is a given attribute with $\delta(U_C)$ (resp. $\rho(U_C)$) as its domain (resp. range). A DL-Lite$_A$ *knowledge base* (KB) is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ where the TBox $\mathcal{T}$ (resp. ABox $\mathcal{A}$) consists of a finite set axioms such as (see [10] for a complete list):

- *inclusion assertions* of the form $B \sqsubseteq C$, $Q \sqsubseteq R$, $E \sqsubseteq D$, and $U \sqsubseteq V$;
- *membership assertions* of the form $A(a)$, $D(c)$, $P(a, b)$, and $U(a, c)$, where a resp. b are individual names in Γ_I and c is a value in Γ_V.

We denote by $pred(\mathcal{T})$ the set of all concept and role names. The semantics of DL-Lite$_A$ is given in terms of FO interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a nonempty domain,

the disjoint union of $\Delta_I^{\mathcal{I}}$ of $\Delta_V^{\mathcal{I}}$, and $\cdot^{\mathcal{I}}$ is an *interpretation function* as usual (see [10]). Satisfaction of axioms and logical implication are denoted by $\models$. We assume the *unique name assumption* (UNA) holds for different individuals resp. values and adopt the *constant domain assumption*, saying that all models share the same domain. Checking satisfiability of DL-Lite$_A$ ontologies is *FO rewritable* [10], i.e., for every $\mathcal{T}$, there is a Boolean FO query $q_{\mathcal{T}}$ (constructible from $\mathcal{T}$) s.t. for every $\mathcal{A}$, $\mathcal{T} \cup \mathcal{A}$ is satisfiable iff $DB(\mathcal{A}) \not\models q_{\mathcal{T}}$, where $DB(\mathcal{A})$ is the *least Herbrand model* of $\mathcal{A}$.

Syntax DL-Lite$_A$ (S,F). Let $\mathbb{T}$ be a timeline and let Γ_S, Γ_I, and Γ_V be pairwise disjoint sets as defined above. A *spatial-stream knowledge base* is defined as a tuple $\mathcal{K}_{\mathcal{SF}} = \langle \mathcal{T}, \mathcal{A}, \mathcal{S}_{\mathcal{A}}, \langle \mathcal{F}, \mathcal{S}_{\mathcal{F}} \rangle, \mathcal{B} \rangle$, where $\mathcal{T}$ (resp. $\mathcal{A}$) is a DL-Lite$_A$ TBox (resp. ABox), $\mathcal{S}_{\mathcal{A}}$ is a spatial database, and $\langle \mathcal{F}, \mathcal{S}_{\mathcal{F}} \rangle$ is a stream database with support for spatial data. Furthermore, $\mathcal{B} \subseteq \Gamma_I \times \Gamma_S$ is a partial function called the *spatial binding* from $\mathcal{A}$ to $\mathcal{S}_{\mathcal{A}}$ and $\mathcal{F}$ to $\mathcal{S}_{\mathcal{F}}$. In case we restrict to a spatial KB resp. stream KB, we have $\mathcal{K}_{\mathcal{S}} = \langle \mathcal{T}, \mathcal{A}, \mathcal{S}_{\mathcal{A}}, \mathcal{B} \rangle$ resp. as $\mathcal{K}_{\mathcal{F}} = \langle \mathcal{T}, \mathcal{A}, \mathcal{F} \rangle$.

We introduce for DL-Lite$_A$ the ability to specify the *localization* of $pred(\mathcal{T})$. For this, we extend the syntax similar to [13] and include concept and role localization:

$$C ::= \top_C \mid B \mid \neg B \mid \exists Q.C' \mid (loc\ A) \mid (loc_s\ A)$$
$$R ::= Q \mid \neg Q \mid (loc\ Q) \mid (loc_s\ Q),$$

where $s \in \Gamma_S$ and the concept and roles are as before. Intuitively, $(loc\ A)$ is the set of individuals in A that can have a spatial extension, and $(loc_s\ A)$ is the subset which have a single extension s. The extension with streaming is captured by the following axioms:

$$(stream_F\ C),\ (stream_F\ R),$$

where F is a particular stream over either complex concepts C or roles R in $\langle \mathcal{F}, \mathcal{S}_{\mathcal{F}} \rangle$.

Example 3. For Ex. 2 , a simple TBox contains $(stream_{F_1}\ speed)$, $(stream_{F_1}\ (loc\ pos))$, $(stream_{F_1}\ Vehicle)$, and $(stream_{F_2}\ hasState)$. Further, we have the axioms $Car \sqsubseteq Vehicle$, $Bike \sqsubseteq Vehicle$, and $Ambulance \sqsubseteq \exists hasRole.Emergency$.

Semantics DL-Lite$_A$ (S,F). We give a semantics to the localization $(loc\ Q)$ and $(loc_s\ Q)$ for individuals of Q with some spatial extension resp. located at s, such that a KB $\mathcal{K}_{\mathcal{S}} = \langle \mathcal{T}, \mathcal{A}, \mathcal{S}, \mathcal{B} \rangle$ can be readily transformed into an ordinary DL-Lite$_A$ KB $\mathcal{K}_{\mathcal{O}} = \langle \mathcal{T}', \mathcal{A}' \rangle$, using the fresh spatial top concept $C_{\mathcal{S}_{\mathcal{T}}}$ and spatial concepts C_s. An *interpretation of* $\mathcal{K}_{\mathcal{S}}$ is a structure $\mathcal{I}_S = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, b^{\mathcal{I}} \rangle$, where $\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ is an interpretation of $\langle \mathcal{T}, \mathcal{A} \rangle$ and $b^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Gamma_S$ is a partial function that assigns some individuals a location, such that for every $a \in \Gamma_I$, $(a, s) \in \mathcal{B}_{\mathcal{A}}$ implies $b^{\mathcal{I}}(a^{\mathcal{I}}) = s$. We extend the semantics with $(loc\ Q)$ and $(loc_s\ Q)$, where Q is an atomic role in $\mathcal{T}$ by ($(loc\ A)$ and $(loc_s\ A)$ are accordingly):

$$(loc\ Q)^{\mathcal{I}_S} \supseteq \{(a_1, a_2) \mid (a_1, a_2) \in Q^{\mathcal{I}} \wedge \exists s \in \Gamma_S : (a_2, s) \in b^{\mathcal{I}}\},$$
$$(loc_s\ Q)^{\mathcal{I}_S} = \{(a_1, a_2) \mid (a_1, a_2) \in Q^{\mathcal{I}} \wedge (a_2, s) \in b^{\mathcal{I}}\}.$$

The transformation of $\mathcal{K}_{\mathcal{S}}$ to an ordinary DL-Lite$_A$ KB $\mathcal{K}_{\mathcal{O}}$ adds the spatial concepts $C_{\mathcal{S}}$ and C_s for every $s \in \Gamma_S$ to $\mathcal{T}'$. Then, each $loc\ Q$ (resp. $(loc_s\ Q)$) is replaced in $\mathcal{T}'$ by the axioms $\exists Q.C_{\mathcal{S}}$ (resp. $\exists Q.C_s$) and C_s is connected to the top concepts by $C_s \sqsubseteq C_{\mathcal{S}}$. Finally, the binding is compiled into $\mathcal{T}'$ by adding $C_s(a)$ for every $(a, s) \in \mathcal{B}$. Following the approach in [13], it can be shown that the models of $\mathcal{K}_{\mathcal{S}}$ and $\mathcal{K}_{\mathcal{O}}$ correspond wrt. the same domain and $pred(\mathcal{T})$. As a consequence, satisfiability checking for DL-Lite$_A$ ontologies is in LOGSPACE (in fact FO-rewritability), the same holds for DL-Lite$_A(S)$.

The idea of an initial streaming semantics is by interpreting the stream over the full timeline, which can be captured by $\mathcal{F}_{\mathcal{A}}$ as a finite sequence of temporal ABoxes, such

that $\mathcal{F}_{\mathcal{A}} = (\mathcal{F}_i)_{\mathbb{T}_{\min} \leq i \leq \mathbb{T}_{\max}}$, which is obtained via the evaluation function v on $\mathcal{F}$ and $\mathbb{T}$ (cf. [7], [14]). Then, we define the interpretation of the point-based model over $\mathbb{T}$ as a sequence $\mathcal{I}_F{=}(\mathcal{I}_i)_{\mathbb{T}_{\min} \leq i \leq \mathbb{T}_{\max}}$ of interpretations $\mathcal{I}_i{=}\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}_i} \rangle$. $\mathcal{I}_F$ is a model of $\mathcal{K}_{\mathcal{F}}$, such that $\mathcal{I}_F \models \mathcal{K}_{\mathcal{F}}$ iff $\mathcal{I}_i \models \mathcal{F}_i$ and $\mathcal{I}_i \models \mathcal{T}$, for all $i \in \mathbb{T}$.

The semantics of the $(stream_F\ C)$ and $(stream_F\ R)$ axioms is along the same line. A stream axiom is satisfied, if a complex concept C (resp. role R) holds over all the time points of stream $F = (\mathbb{T}, v, P)$; thus we restrict our models such that:

$$(stream_F\ C)^{\mathcal{I}} = \bigcup_{i \in tp(\mathbb{T},P)} (\{e \in \Delta^{\mathcal{I}} \mid e \in C^{\mathcal{I}_i}\}),$$
$$(stream_F\ R)^{\mathcal{I}} = \bigcup_{i \in tp(\mathbb{T},P)} (\{(a_1, a_2) \mid (a_1, a_2) \in R^{\mathcal{I}_i}\}),$$

where $tp(\mathbb{T}, P)$ is a set of time points determined by the segmentation of $\mathbb{T}$ by P. This allows us to check for the *satisfiability* of a KB and gives us a global consistency, which is of theoretical nature, since we would need to know the full timeline.

5 Spatial-Stream Query Language over DL-Lite$_{\mathcal{A}}$ (S,F)

We next define spatial-stream conjunctive queries (STCQ) over $\mathcal{K}_{\mathcal{SF}}$. Such queries may contain ontology, spatial, and stream predicates. An STCQ $q(\mathbf{x})$ is a formula:

$$\bigwedge_{i=1}^{l} Q_{O_i}(\mathbf{x}, \mathbf{y}) \wedge \bigwedge_{j=1}^{n} Q_{S_j}(\mathbf{x}, \mathbf{y}) \wedge \bigwedge_{k=1}^{m} Q_{F_k}(\mathbf{x}, \mathbf{y}) \tag{1}$$

where $\mathbf{x}$ are the *distinguished* (*answer*) variables and $\mathbf{y}$ are either *non-distinguished* (*existentially quantified*) variables, individuals from Γ_I, or values from Γ_V. Each $Q_{O_i}(\mathbf{x}, \mathbf{y})$ has the form $A(z)$ or $P(z, z')$, with A resp. P from $pred(\mathcal{T})$ and z, z' from $\mathbf{x} \cup \mathbf{y}$. Each atom $Q_{S_j}(\mathbf{x}, \mathbf{y})$ is from the vocabulary of spatial relations (see Sec. 3) and of the form $S(z, z')$, with z, z' from $\mathbf{x} \cup \mathbf{y}$; $Q_{F_j}(\mathbf{x}, \mathbf{y})$ is similar to $Q_{O_i}(\mathbf{x}, \mathbf{y})$ but adds the vocabulary for stream operators, which are taken from [6] and relate to CQL operators [4]. Moreover, we have a window $\boxplus$ over a stream that is derived from L (in $\mathbb{Z}^+$ for past or in $\mathbb{Z}^-$ for future time units) resp. $\mathbb{T}_i$, and an *aggregate function* $agr \in \{count, min, max, sum, avg, first, last\}$ (see Sec. 6) that is applied to the data items in the window:[3]

- $Q_{F_j}\ \boxplus_T^L\ agr$ represents the aggregate of last/next L time units of $(stream\ F_j)$;
- $Q_{F_j}\ \boxplus_T$ represents the current tuples of F_j with $L = 0$;
- $Q_{F_j}\ \boxplus_T^O\ agr$: represents the aggregate of all previous L time units of F_j;

Example 4. We modify $q_1(x, y)$ of Ex. 1 and use the stream operators instead:

$$\begin{aligned} q_1(x, y) : {} & LaneIn(x) \wedge hasLocation(x, u) \wedge intersects(u, v) \wedge position_{\boxplus_T^4 line}(y, v) \\ & \wedge Vehicle(y) \wedge speed_{\boxplus_T^4 avg}(y, r) \wedge (r > 30) \wedge isManaged(x, z) \\ & \wedge SignalGroup(z) \wedge hasState_{\boxplus_T^{-8} first}(z, Stop) \end{aligned}$$

Certain Answer Semantics. In the streamless setting, due to the OWA, queries are evaluated over all (possibly infinitely many) models. *Certain answers* retain the tuples that are answers in all possible models. More formally, a *match* for $q(\mathbf{x})$ in an interpretation $\mathcal{I}{=}\langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ of $\mathcal{K}$ is a function $\pi : \mathbf{x} \cup \mathbf{y} \to \Delta^{\mathcal{I}}$ such that $\pi(c) = c^{\mathcal{I}}$, for each constant c in $\mathbf{x} \cup \mathbf{y}$, and for each $i = 1, \ldots n$:

- (i) $\pi(z) \in A^{\mathcal{I}}$, for $Q_{O_i}(\mathbf{x}, \mathbf{y}) = A(z)$; and
- (ii) $(\pi(z), \pi(z')) \in P^{\mathcal{I}}$, for $Q_{O_i}(\mathbf{x}, \mathbf{y}) = P(z, z')$.

[3] This would be represented in CQL as `R[Range L]`, `R[Now]`, `R[Range L Slide D]`, etc.

Then, a tuple $\mathbf{c} = c_1, \ldots, c_k$ over Γ_I is a *certain answer* for $q(\mathbf{x})$ in $\mathcal{I}$, $\mathbf{x} = x_1, \ldots, x_k$, if $q(\mathbf{x})$ has some match π in $\mathcal{I}$ such that $\pi(x_i) = c_i$, $i = 1, \ldots, k$; and $\mathbf{c}$ is a certain answer for $q(\mathbf{x})$ over $\mathcal{K}$, if it is a certain answer in every model $\mathcal{I}$ of $\mathcal{K}$. The *result* of $q(\mathbf{x})$ over $\mathcal{K}$, denoted $Cert(q(\mathbf{x}), \mathcal{K})$, is the set of all its (certain) answers. If we drop $\mathcal{T}$, we have a DB setting and denote with $Eval(q(\mathbf{x}), \mathcal{I})$ the set of matches of $q(\mathbf{x})$ over a single model (a database instance) $\mathcal{I}$ of $\mathcal{A}$ that coincides with $DB(\mathcal{A})$.

Spatial Queries. For spatial CQ we have additionally spatial atoms and extend a match for $q(\mathbf{x})$ and $j = 1, \ldots, m$ as follows:

- (iii) $\exists s, s' \in \Gamma_S : (\pi(z), s) \in b^{\mathcal{I}} \wedge (\pi(z'), s') \in b^{\mathcal{I}} \wedge \mathcal{S} \models S(s, s')$, for $Q_{S_j}(\mathbf{x}, \mathbf{y}) := S(z, z')$.

For spatial atoms individuals must have (unique) spatial extensions and the relationship between them must hold. As shown in [13], the semantic correspondence between $\mathcal{K}_{\mathcal{O}}$ and $\mathcal{K}_{\mathcal{S}}$ guarantees that we can transform $q(\mathbf{x})$ into an equivalent query over ${\mathcal{K}_{\mathcal{S}}}' = \langle \mathcal{T}', \mathcal{A}', \mathcal{S}_{\mathcal{A}} \rangle$ by replacing each spatial atom $S(z, z')$ in $q(\mathbf{x})$ with $q_{S_j}(z, z')$:

$$\bigvee_{s,s' \in \Gamma_S} (C_s(z) \wedge C_{s'}(z') \wedge S(s, s'));$$

it is easily transformed to a union of CQs (UCQ): $uq(\mathbf{x}) = q_{S_0}(\mathbf{x}) \wedge \cdots \wedge q_{S_n}(\mathbf{x}) \wedge q_{rest}(\mathbf{x})$, where $q_{rest}(\mathbf{x})$ is the rewriting of the non-spatial part of $q(\mathbf{x})$. A tuple $\mathbf{c}$ is then an answer in $uq(\mathbf{x})$ over ${\mathcal{K}_{\mathcal{S}}}'$ iff it is an answer in every model $\mathcal{I}'$ of ${\mathcal{K}_{\mathcal{S}}}'$. Using the above rewriting and the semantic correspondence of $\mathcal{K}_{\mathcal{O}}$ and $\mathcal{K}_{\mathcal{S}}$, spatial atoms can be rewritten into a "standard" DL-Lite$_A$ UCQ, and conclude that $Cert(q(\mathbf{x}), \mathcal{K}_{\mathcal{S}}) = Cert(uq(\mathbf{x}), {\mathcal{K}_{\mathcal{S}}}')$. Thus, answering spatial CQs is still FO-rewritable (details in [13]).

QA over Streams. The standard certain answers semantics for ontology-mediated QA can be extended to streams in different ways. A possible approach is described in [8], [7], and [14], where a boolean CQ is evaluated over $\mathcal{K}_{\mathcal{F}}$ with $\mathcal{A}$ by rewriting $\mathcal{T}$ such that for every time point $i \in \mathbb{T}$: $\mathcal{T}, \mathcal{A}, \mathbb{T} \models q$ iff $DB(\mathcal{A}_i) \models q_{\mathcal{T}}$, for all i $\in \mathbb{T}$,
where $q_{\mathcal{T}}$ is the FO rewriting of q in $\mathcal{T}$ and $DB(\mathcal{A}_i)$ is the least Herbrand model of A at time point $\mathbb{T}_i$. This approach is extended by either evaluation over the entire history of $\mathcal{A}$ using temporal operators (e.g., *always*) from Linear Temporal Logic (LTL) [7], or by using time intervals and a temporal query language with less/greater operators [14]. Our approach aims at answering queries at *a single* time point $\mathbb{T}_i$ with stream atoms that define *aggregate functions* on different windows sizes relative to $\mathbb{T}_i$; thus we ignore LTL operators and time intervals. We consider a semantics based on *epistemic aggregate queries* (EAQ) over ontologies [11] by dropping the order of time points for the membership assertions and handle the (streamed) assertions as *bags*, which is similar to "normal" stream processing approaches. As described in [11], aggregate queries are defined over bags of numeric and symbolic values, called *groups* and denoted as $\{| \cdot |\}$. Aggregates cannot be directly transferred to DL-Lite, thus [11] extended the database semantics for aggregates with an epistemic operator $\mathbf{K}$ and a two-layer evaluation using the completion w.r.t $\mathcal{T}$. More formally, an EAQ is defined as $q_a(\mathbf{x}, agr(y)) : \mathbf{K}\ \mathbf{x}, y, \mathbf{z}.\ \phi$, [4]
where $\mathbf{x}$ are the *grouping variables*, $agr(y)$ is the *aggregate function and variable*, and ϕ is a CQ called *main conditions*; $\mathbf{z}$ are the disjoint existential variables of ϕ. We call $\mathbf{w} := \mathbf{x} \cup y \cup \mathbf{z}$ the $\mathbf{K}$-variables of ϕ. The definition of a group was extended in [11] by a multiset $H_{\mathbf{d}}$ of groups $\mathbf{d}$, called $\mathbf{K}$-group, as:

$$H_{\mathbf{d}} := \{| \ \pi(y) \mid \pi \in KSat_{\mathcal{I},\mathcal{K}}(\mathbf{z}; \phi) \text{ and } \pi(\mathbf{x}) = \mathbf{d} \ |\},$$

[4] We simplified EAQs of [11] by omitting ψ and consider only aggregates with a single variable.

where $KSat$ are the *satisfying* $\mathbf{K}$-matches of ϕ *for* the model $\mathcal{I}$ of $\mathcal{K}$ and given by:

$$KSat_{\mathcal{I},\mathcal{K}}(\mathbf{w};\phi) := \{\pi \in Eval(\phi,\mathcal{I}) \mid \pi(\mathbf{w}) \in Cert(aux_{q_a},\mathcal{K})\},$$

where $aux_{q_a}(\mathbf{w}) \leftarrow \phi$ is the auxiliary atom used to map $\mathbf{w}$ only to *known solutions*. The set of $\mathbf{K}$-answers for an EAQ query q over $\mathcal{I}$ and $\mathcal{K}$ can now be derived as:

$$q_a^{\mathcal{I}} := \{(\mathbf{d}, agr(H_{\mathbf{d}})) \mid \mathbf{d} = \pi(\mathbf{x}),\ \text{for some } \pi \in KSat_{\mathcal{I},\mathcal{K}}(\mathbf{w};\phi)\}.$$

The *epistemic certain answers* $ECert(q_a,\mathcal{K})$ for a query q_a over $\mathcal{K}$ is the set of $\mathbf{K}$-answers that are answers in every model $\mathcal{I}$ of $\mathcal{K}$. To compute $ECert(q_a,\mathcal{K})$, [11] gave a "general algorithm" GA that (1) computes the certain answers, (2) projects on the $\mathbf{K}$-variables, and (3) aggregates the resulting tuples. Importantly, evaluating EAQs reduces to standard CQ evaluation over DL-Lite$_A$ with LOGSPACE data complexity.

6 Query Rewriting by Stream Aggregation

As discussed above, we drop the temporal order by evaluating the EAQ queries over the bag of temporal ABox assertions. Thus, we evaluate the EAQ over several *filtered and merged temporal* ABoxes that are created according to the window operator and $\mathbb{T}_i$. The filtering and merging, relative to the window size and $\mathbb{T}_i$, creates a *windowed* ABox $\mathcal{A}_{\boxplus}$ that is the union of a static ABox $\mathcal{A}$ and the filtered stream ABoxes from $\mathcal{F}$. The EAQ aggregates can be applied on normal objects by counting, on concrete values by aggregate functions like sum, and on spatial objects by applying geometrical functions that create geometries like $line$. More formally, a stream atom $\phi \boxplus_T^L agr$ is evaluated as an EAQ over ontologies defined as follows: $q_\phi(\mathbf{x}, agr(y)) : \mathbf{K}\ \mathbf{x}, y, \mathbf{z}.\ \phi \boxplus_T^L$,
where $\mathbf{x}$ is the grouping variables and y the aggregate variable, $\mathbf{z}$ are the disjoint existential variables, and ϕ is a subquery of q with atoms in the same *scope* of the window operator $\boxplus_T^L$ and aggregate function agr.

Example 5. For query $q_1(x,y)$ of Ex. 4, we have three EAQs represented as:

$$q_{pos}(y, line(v)) : \mathbf{K}\ y, v.\ Vehicle(y) \wedge position(y,v);$$
$$q_{speed}(y, avg(r)) : \mathbf{K}\ y, r.\ Vehicle(y) \wedge speed(y,r);$$
$$q_{state}(z, first(m)) : \mathbf{K}\ z, m.\ hasState(z,m)$$

We extend the evaluation of EAQs for the stream setting, such that an EAQ is evaluated over the window relative to $\mathbb{T}_i$, the window operator $\boxplus_T^L$, and the pulse P. $KSat$ is now the set of $\mathbf{K}$-matches of ϕ for a model $\mathcal{I}_{\boxplus}$ of $\mathcal{K}_{\boxplus}$, where the *windowed* ABox $\mathcal{A}_{\boxplus}$ is defined as $\mathcal{A}_{\boxplus} = \mathcal{A} \cup \bigcup\{\mathcal{A}_i \mid w_s \leq i \leq w_e\}$. We have four cases for the window size L and a pulse P, where P enlarges L according to its interval length:

- a current window with $L = 0$ that is $w_s = w_e = \mathbb{T}_i$;
- a past window with $L > 0$ leading to $w_s = (\mathbb{T}_i - L)$ and $w_e = \mathbb{T}_i$;
- a future window with $L < 0$ that is $w_s = \mathbb{T}_i$ and $w_e = (\mathbb{T}_i + L)$;
- the entire history with O resulting in $w_s = 0$ and $w_e = \mathbb{T}_i$.

We obtain KB $\mathcal{K}_{\boxplus} = \langle \mathcal{T}, \mathcal{A}_{\boxplus} \rangle$ as above; the *epistemic certain answers* $ECert_{\boxplus}(q_\phi, \mathcal{K}_{\boxplus})$ for q_ϕ over $\mathcal{K}_{\boxplus}$ are the $\mathbf{K}$-answers that are answers in every model $\mathcal{I}_{\boxplus}$ of $\mathcal{K}_{\boxplus}$ as:

$$q_\phi^{\mathcal{I}_{\boxplus}} := \{(\mathbf{d}, agr(H_{\mathbf{d}})) \mid \mathbf{d} = \pi(\mathbf{x}),\ \text{for some } \pi \in KSat_{\mathcal{I}_{\boxplus},\mathcal{K}_{\boxplus}}(\mathbf{w};\phi)\}.$$

In $ECert_{\boxplus}$, we have not addressed yet the *validity* of a membership assertion, say in $\mathcal{A}_{\boxplus_1}$, to the next assertion in $\mathcal{A}_{\boxplus_3}$. There are two suggestive semantics: in the first, we ignore intermediate time points, and thus $\mathcal{A}_{\boxplus_2}$ will be unknown. The second semantics fills the missing gaps with the previous assertion, i.e. copies the assertion from $\mathcal{A}_{\boxplus_1}$ to $\mathcal{A}_{\boxplus_2}$. For specific aggregation functions, e.g., max, min, or $last$, the two semantics coincide, but for sum, avg, and $count$, this leads to different results.

Example 6. We pose the query $q_1(x, y)$ at $\mathbb{T}_1$ and replace the stream atoms with auxiliary atoms related to the EAQ of Ex. 5:

$$q_1(x,y) : LaneIn(x) \wedge hasLocation(x,u) \wedge intersects(u,v) \wedge q_{pos}(y,v) \wedge q_{speed}(y,r) \\ \wedge (r > 30) \wedge isManaged(x,z) \wedge q_{state}(z, Stop)$$

The queries are computed using the ABoxes $\mathcal{A}_{\boxplus[0,1]} = \mathcal{A} \cup \mathcal{A}_0 \cup \mathcal{A}_1$ and $\mathcal{A}_{\boxplus[1,4]} = \mathcal{A} \bigcup_{1 \leq i \leq 4} \mathcal{A}_i$. This leads under $ECert_{\boxplus}$ for q_{speed} to the groups G_{c_1}=$\{|30, 29, 34|\}$ and G_{b_1}=$\{|10, 5|\}$, which results in q_{speed}=$\{(c_1, 31), (b_1, 7.5)\}$. The results for the other EAQ are q_{state}=$\{(t_1, Red)\}$ and q_{pos}=$\{(c_1, ((5,5),(6,5),(7,5))), (b_1, ((1,1),(2,1)))\}$.

$ECert_{\boxplus}$ gives the certain answers for a single EAQ including the streamless atoms in the same scope as the stream atoms. Answering the full CQ, denoted $ECertAll(q, \mathcal{K}_{\mathcal{F}}, \mathbb{T}_i)$, can be done by answering each EAQ q_{ϕ_j} by $ECert_{\boxplus}(q_{\phi_j}, \mathcal{K}_{\boxplus_{w(\phi_j, \mathbb{T}_i)}})$ separately and combining the answers as follows, where $w(\phi_j, \mathbb{T}_i)$ is the computed window size:

$$\mathcal{K}_{\mathcal{F}}, \mathbb{T}_i \models q_{\phi_1} \wedge q_{\phi_2} \text{ iff } \mathcal{K}_{\mathcal{F}}, \mathbb{T}_i \models q_{\phi_1} \text{ and } \mathcal{K}_{\mathcal{F}}, \mathbb{T}_i \models q_{\phi_2}$$

Details on deriving each q_{ϕ_j} from q via hypertree decomposition is left for future work.

We now introduce the algorithm `NSQ` (see Alg. 1), where $\mathbf{z}^{\phi}$ are the non-distinguished variables in ϕ and `PerfectRef` (resp. `Answer`) is the "standard" query rewriting (resp. evaluation) as in [10]. `NSQ` extends the `GA` of [11] to compute the answers for stream CQs as follow: (1) calculate the epistemic answer for each stream atom over the different windowed ABoxes and store the result in an *auxiliary ABox* using new atoms. Further, replace each stream atom with a new auxiliary atom; (2) calculate the certain answers over $\mathcal{A}$ and the auxiliary ABox, using the "standard" DL-Lite$_A$ query evaluation.

Theorem 1. *Correctness of* `NSQ`. *For every stream CQ q, KB $\mathcal{K}_{\mathcal{F}}$, and time point $\mathbb{T}_i$, it holds that $ECertAll(q, \mathcal{K}_{\mathcal{F}}, \mathbb{T}_i) =$ `NSQ`$(q, \mathcal{K}_{\mathcal{F}}, \mathbb{T}_i)$.*

Our proof sketch considers that q has to be restricted by $\mathcal{T}$ and standard aggregates have to fullfill aggregate conditions (see below). Correctness is based on the idea, that we exploit the computational property of decoupling answering each EAQ (`Step 1`) from answering the full CQ as follows (similar to [9]):

1. we let $q_{FOL}(\mathbf{x})$ be the FOL query obtained from $q(\mathbf{x})$ by replacing each EAQ q_{ϕ_i} by a new predicate R_{ϕ_i} with the same arity;
2. we define $\mathcal{I}_{q,\mathcal{K}_{\mathcal{F}}}$ as the FOL interpretation over $\Delta^{\mathcal{I}'}$ for R_{ϕ_i} such that (i) $\Delta^{\mathcal{I}} = \Delta^{\mathcal{I}'}$ (same domain) and (ii) the *extension* of R_{ϕ_i} is $R_{\phi_i}^{\mathcal{I}_{q,\mathcal{K}_{\mathcal{F}}}} = ECert_{\boxplus}(q_{\phi_i}, \mathcal{K}_{\boxplus})$.

For `Step 2`, we let $Eval(q_{FOL}, \mathcal{I}_{q,\mathcal{K}_{\mathcal{F}}})$ be the result of evaluating q_{FOL} over $\mathcal{I}_{q,\mathcal{K}_{\mathcal{F}}}$. The semantic correspondence between $Eval(q_{FOL}, \mathcal{I}_{q,\mathcal{K}_{\mathcal{F}}})$ and $ECertAll(q, \mathcal{K}_{\mathcal{F}}, \mathbb{T}_i)$ is guaranteed under the following conditions:

1. The evaluation of q_{FOL} should not be over a possibly infinite domain $\Delta^{\mathcal{I}}$, thus allowing FOL queries that are *domain independent*;
2. the extension of R_{ϕ_i} in $\mathcal{I}_{q,\mathcal{K}_{\mathcal{F}}}$ needs to be finite, or else the evaluation is infeasible.

Domain independence is ensured by using special interpretation based on the active domain, denoted as $\Delta^{\mathcal{I}}_{Adom}$. The active domain is a subset of $\Delta^{\mathcal{I}}$ built from all constants in $\mathcal{K}_{\mathcal{F}}$, $\mathbb{T}$, and q. Furthermore, FOL queries under active domain semantics are equivalent to range-restricted relational calculus (RRC) queries, with allows the translation from FOL into RRQ queries as shown in [1].

Standard Aggregates. Different aggregation functions for use in $ECert(q, \mathcal{K})$ were already provided in [11]. For this, $\mathcal{K}$, q and $H_{\mathbf{d}}$ have to fulfill *aggregate conditions* to avoid

Algorithm 1: `NSQ` - Answer Naive Stream Query

Input: A stream conjunctive query q, time point $\mathbb{T}_i$, and a KB $\mathcal{K}_\mathcal{F}$
Output: Set of tuples O
`/* Step 1: Detemporalize */`
foreach Q_{F_i} *of* q **do**
  $\mathcal{A}_{\boxplus_i} \leftarrow \mathcal{A} \bigcup_{w_s \leq j \leq w_e} \mathcal{A}_j$ according to $\boxplus_T^L$ and $\mathbb{T}_i$;
  $\mathcal{K}_{\boxplus_i} \leftarrow \langle \mathcal{T}, \mathcal{A}_{\boxplus_i} \rangle$;
  build $aux_i(\mathbf{x}, y, \mathbf{z})$ from $\phi \boxplus_T^L agr$ of Q_{F_i} ;
  build $q_{i,1}(\mathbf{x}, y, \mathbf{z}^\phi)$ from $\texttt{PerfectRef}(aux_i, \mathcal{T})(\mathbf{x}, y, \mathbf{z})$;
  build $q_{i,2}(x, agr(y))$ from $q_{i,1}(\mathbf{x}, y, \mathbf{z}^\phi)$ and $\phi \boxplus_T^L agr$;
  $R_{i,1} :=$ evaluate $\texttt{Answer}(aux_i, \mathcal{K}_{\boxplus_i})$ (Certain answers) ;
  $R_{i,2} :=$ evaluate $q_{i,1}$ over $R_{i,1}$ ($\mathbf{K}$ projection) ;
  $R_i \;\; :=$ evaluate $q_{i,2}$ over $R_{i,2}$ (Aggregation) ;
  Add R_i to $\mathcal{A}_{aux}$ and replace Q_{F_i} in q with $R_i(x, y)$;
`/* Step 2: Standard evaluation */`
$O :=$ evaluate $\texttt{Answer}(q, \langle \mathcal{T}, \mathcal{A} \cup \mathcal{A}_{aux} \rangle)$;

"wrong" aggregation and ensure correctness: (1) q has to be non-*trivial* and *coherent*, such that q is satisfiable regarding $\mathcal{T}$; (2) the variables in q must be *restricted*, which introduces functional dependency on epistemic variables; (3) q has *uniform grouping*; (4) dropping *zero groups*, i.e., groups having only zero values (see [11] for details). For *concrete values*, the following aggregation functions were introduced based on $ECert(q, \mathcal{K})$ for every $\mathcal{A}$: (i) $count, min, max$: (1) has to hold for q and $\mathcal{K}$; (ii) sum: (1), (2), and (4) have to hold for q and $\mathcal{K}$; (iii) avg: (1), (2), and (3) have to hold for q and $\mathcal{K}$.

For $last$ and $first$, we extend the definition of $H_\mathbf{d}$, as the sequence of time points is lost. However, by iteratively checking if we have a match in some ABox $\mathcal{A}_{\boxplus_i}$, $w_s \leq i \leq w_e$, at a single time point, we can determine where we have the first resp. last match. We extend $H_\mathbf{d}$ for $first$ as follows ($last$ is similar):

$$H'_\mathbf{d} := \{\!|\ \pi(y) \mid (\pi \in \pi(\mathbf{x}) = \mathbf{d} \text{ and } (\pi \in KSat_{\mathcal{I}_0,\mathcal{K}}(\mathbf{z}; \phi) \text{ or } (\pi \in KSat_{\mathcal{I}_1,\mathcal{K}}(\mathbf{z}; \phi) \text{ and } \pi \notin KSat_{\mathcal{I}_{[0,0]},\mathcal{K}}(\mathbf{z}; \phi)) \text{ or ... or } (\pi \in KSat_{\mathcal{I}_n,\mathcal{K}}(\mathbf{z}; \phi) \text{ and } \pi \notin KSat_{\mathcal{I}_{[0,n-1]},\mathcal{K}}(\mathbf{z}; \phi))\ )\ |\!\},$$

where $KSat_{\mathcal{I}_i,\mathcal{K}}$, resp. $KSat_{\mathcal{I}_{[i,j]},\mathcal{K}}$, is the set of *satisfying* $\mathbf{K}$-matches for a model $\mathcal{I}_i$ of $\langle \mathcal{T}, \mathcal{A} \cup \mathcal{A}_i \rangle$, resp. $\mathcal{I}_{[i,j]}$ of $\langle \mathcal{T}, \mathcal{A} \cup \mathcal{A}_i \cup \cdots \cup \mathcal{A}_j \rangle$.

Spatial Aggregates. For *spatial objects*, we define aggregation functions applied on the multiset of $H_\mathbf{d}$. As the order of assertions (i.e., points) is lost, we must rearrange them to an admissible geometry that is a point sequence $p = (p_1, \ldots, p_n)$ as defined in Sec. 3. We thus extend agr with the new functions agr_{point} and agr_{line} on $H_\mathbf{d}$ to create new admissible geometries $g(s_\mathbf{d})$:

- agr_{point}: we evaluate the function $last$ (as shown before) to get the last available position p_1 and assign it to g as $g(s_\mathbf{d}) = (p_1)$;
- agr_{line}: we create the sequence $(p_1, \ldots, p_n)$, where $p_1 \neq p_n$ and determine a total order on the bag of points in each $\mathbf{K}$-group. The total order is defined such that we have a starting point using $last$ and search backwards using $last$ subsequently.

Further functions such as computing a polygon could be defined. We must consider the binding $\mathcal{B}$ for spatial aggregates, hence we add $\exists s \in \Gamma_S : (\pi(y), s) \in b^\mathcal{I}$ to $KSat$. In contrast to numerical aggregates, spatial aggregates introduce for each $\mathbf{K}$-group $(\mathbf{d}, agr(H_\mathbf{d}))$ a new spatial object $s_\mathbf{d}$ of Γ_S and an admissible geometry $g(s_\mathbf{d})$ with

$agr(H_{\mathbf{d}}) = (p_1, \ldots, p_n)$. This is achieved by adding a binding $(\mathbf{d}, s_{\mathbf{d}})$ to $\mathcal{B}$ and creating a new mapping $g : s_{\mathbf{d}} \rightarrow (p_1, \ldots, p_n)$ in $\mathcal{S}_{aux}$.

Combining $\mathcal{K}_{\mathcal{F}}$ and $\mathcal{K}_{\mathcal{S}}$. We combine the spatial and temporal elements of a query q and KB $\mathcal{K}$ as follows: (1) detemporalize the stream atoms using EAQs; (2) transform the spatial query and the KB to an ordinary UCQ and KB as shown in Sec. 4. We keep LOGSPACE data complexity, as evaluating an EAQ is in LOGSPACE and the number of EAQ computations is limited by the number of aggregate atoms. Spatial binding and relations do not increase the data complexity.

7 Related Work

Data stream management systems (DSMSs) such as STREAM [4], were built supporting streaming applications by extending RDMBS. More recently, RDF stream processing engines, such as C-SPARQL [5], SPARQLstream [8], and CQELS [16], have been proposed to enable the processing of RDF streams integrated with other Linked Data sources. Besides C-SPARQL, most of them follow the DSMSs paradigm and do not support stream reasoning. EP-SPARQL [3] resp. LARS [6] proposes a language that extends SPARQL resp. CQ with stream reasoning, but translated their KB into expressive (less efficient) logic programs. For spatio-temporal RDF stream processing, a few SPARQL extensions have been proposed, such as st-SPARQL [15]. Besides [7] and [14] (see Sec. 5), closest to our work are: (i) [19] is a framework for efficient spatio-temporal queries that support spatial operators and aggregation over temporal features; (ii) [8] allows evaluating ontology-mediated QA queries over a range of data streaming systems, and (iii) [18] extends SPARQL using state sequences and aggregates, which are evaluated over streamed ABoxes. Our work differs regarding: (a) the evaluation approach based on EAQ with aggregations and (b) the main focus on querying streams of spatial data. We use results on EAQ in [11], but introduce streams over spatial data and give an initial implementation.

8 Conclusion and Future Work

Our work is sparked by the LDM for V2X communications, which serves as an integration effort for streaming data (e.g., vehicle movements) in a spatial context (e.g., intersections) over a complex domain (e.g., a mobility ontology). We introduced a suitable approach using ontology-mediated QA for realizing the LDM. For spatial-streaming queries, bridging the gap between stream processing and ontology-mediated QA is not straightforward. Thus, we provide an extension using epistemic aggregate queries to detemporalize the stream sources and extend previous work in [13]. The detemporalization preserves FO-rewritability, which allows us to evaluate conjunctive queries with spatial atoms over existing stream RDBMS. Currently, we are working on a technique for suitable query execution plans using hypergraph decomposition. We have been implementing an experimental prototype to check the feasibility of our approach on initial experiments with mobility data (details on www.kr.tuwien.ac.at/research/projects/loctrafflog/sr2016/).

Future research aims at the theoretical and practical side of our approach. On the former, complexity results for our algorithm under different aggregate functions are of interest, as well as dealing with consistent QA. Our bag semantics could introduce inconsistencies in larger windows, we thus may consider different repairs as in belief revision. On the later, the implementation could be extended to pull-based QA and constraints such as guaranteed latency. This requires optimizations, such as caching, window size adaptation,

and efficient query rewriting techniques. Also more complex spatial aggregation, i.e., polygons or trajectories, are of interest. Finally, evaluation and testing using e.g., the SUMO traffic simulation (http://sumo.dlr.de/) under real conditions is desired.

References

1. Abiteboul, S., Hull, R., Vianu, V. (eds.): Foundations of Databases: The Logical Level. Addison-Wesley Longman Publishing Co., Inc. (1995)
2. Andreone, L., Brignolo, R., Damiani, S., Sommariva, F., Vivo, G., Marco, S.: Safespot final report. Tech. Rep. D8.1.1 (2010), available online.
3. Anicic, D., Fodor, P., Rudolph, S., Stojanovic, N.: Ep-sparql: a unified language for event processing and stream reasoning. In: Proc. of WWW 2011. pp. 635–644 (2011)
4. Arasu, A., Babu, S., Widom, J.: The CQL continuous query language: semantic foundations and query execution. VLDB J. 15(2), 121–142 (2006)
5. Barbieri, D.F., Braga, D., Ceri, S., Valle, E.D., Grossniklaus, M.: C-sparql: a continuous query language for rdf data streams. Int. J. Semantic Computing 4(1), 3–25 (2010)
6. Beck, H., Dao-Tran, M., Eiter, T., Fink, M.: LARS: A logic-based framework for analyzing reasoning over streams. In: Proc. of AAAI 2015. pp. 1431–1438 (2015)
7. Borgwardt, S., Lippmann, M., Thost, V.: Temporalizing rewritable query languages over knowledge bases. J. Web Sem. 33, 50–70 (2015)
8. Calbimonte, J., Mora, J., Corcho, Ó.: Query rewriting in RDF stream processing. In: Proc. of ESWC 2016. pp. 486–502 (2016)
9. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Eql-lite: Effective first-order query processing in description logics. In: Proc. of IJCAI 2007. pp. 274–279 (2007)
10. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The *dl-lite* family. J. Autom. Reasoning 39(3), 385–429 (2007)
11. Calvanese, D., Kharlamov, E., Nutt, W., Thorne, C.: Aggregate queries over ontologies. In: Proc. of ONISW 2008. pp. 97–104 (2008)
12. Eiter, T., Füreder, H., Kasslatter, F., Parreira, J.X., Schneider, P.: Towards a semantically enriched local dynamic map. In: Proc. of ITS World Congress 2016. To appear.
13. Eiter, T., Krennwallner, T., Schneider, P.: Lightweight spatial conjunctive query answering using keywords. In: Proc. of ESWC 2013. pp. 243–258 (2013)
14. Klarman, S., Meyer, T.: Querying temporal databases via OWL 2 QL. In: Proc. of RR 2014. pp. 92–107 (2014)
15. Koubarakis, M., Kyzirakos, K.: Modeling and querying metadata in the semantic sensor web: The model strdf and the query language stsparql. In: Proc. of ESWC 2010. pp. 425–439 (2010)
16. Le-Phuoc, D., Dao-Tran, M., Parreira, J.X., Hauswirth, M.: A native and adaptive approach for unified processing of linked streams and linked data. In: ISWC 2011. pp. 370–388 (2011)
17. Netten, B., Kester, L., Wedemeijer, H., Passchier, I., Driessen, B.: Dynamap: A dynamic map for road side its stations. In: Proc. of ITS World Congress 2013 (2013)
18. Özçep, Ö.L., Möller, R., Neuenstadt, C.: Stream-query compilation with ontologies. In: Proc. of AI 2015. pp. 457–463 (2015)
19. Quoc, H.N.M., Le Phuoc, D.: An elastic and scalable spatiotemporal query processing for linked sensor data. In: Proc. of SEMANTICS 2015. pp. 17–24. ACM (2015)
20. Shimada, H., Yamaguchi, A., Takada, H., Sato, K.: Implementation and evaluation of local dynamic map in safety driving systems. J. Transportation Technologies 5(2), 102–112 (2015)
21. Ulbrich, S., Nothdurft, T., Maurer, M., Hecker, P.: Graph-based context representation, environment modeling and information aggregation for automated driving. In: Proc. IEEE Intelligent Vehicles Symposium 2014. pp. 541–547 (2014)