

# More structure Linked data

- Route planning
- Substring autocomplete
- Interface align with the domain of application?

TREE | Autocomplete over TREE structured fragmentations

Choose TREE fragmentation to query over:

Streets of Flemish Address Registry

Give a search term

© the Linked Data Fragments collaborators. [Contact us.](#)

Tell us what you want to see while walking, and we'll make it happen. Don't forget to hit **Plan** when you're done.

**Shops**

They're ok

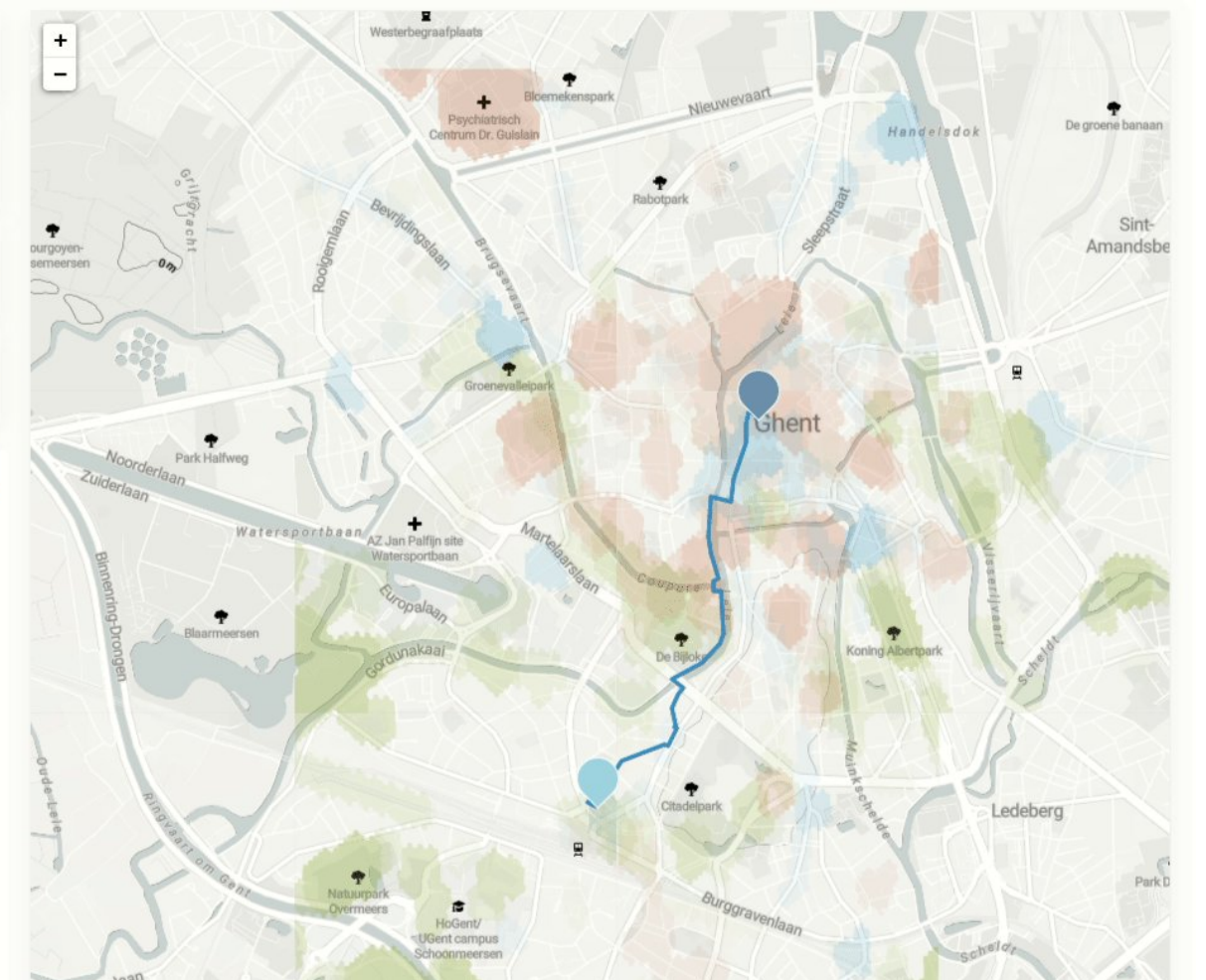
**Trees**

They're ok

**Historic Buildings**

They're ok

**PLAN**



<http://hdelva.be/sem4tra2020/demo.html>

Combination of:  
Clustering of points of interest in OSM  
Routable Tiles (Linked OSM PoC)  
Weighted client-side route planning

[Live demo at tree.linkeddatafragments.org](http://tree.linkeddatafragments.org)

# HOW TREE HYPERMEDIA CAN SPEED UP LINK



# TRAVERSAL-BASED QUERY PROCESSING FOR

# SPARQL QUERIES WITH FILTERS

Bryan-Elliott Tam

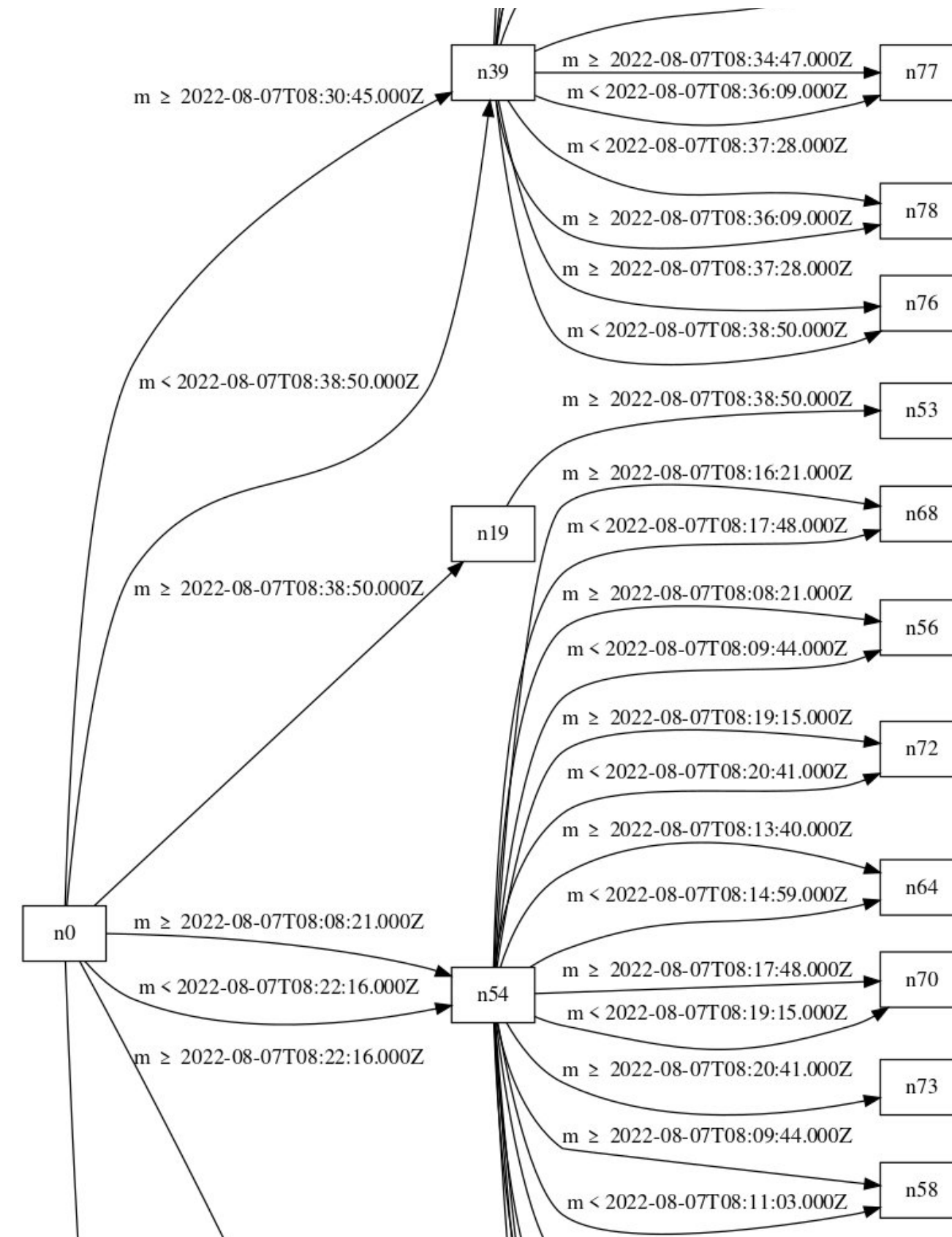
# AGENDA

- Context or the TREE specification
- Formalisation
- Benchmark
- Future work



# TREE specification

- **Fragmented database**
- Graph
- Vertex/Node ([tree:Node](#)) are the data container
- Edge ([tree:Relation](#)) declare a constraint on the data inside a vertex
- The document can grow over time
- We don't maintain a central index
  - Link Traversal Query Processing to fetch the data





# EXAMPLE USE CASE

## Querying observations after the first of January



```
PREFIX sosa: <http://www.w3.org/ns/sosa/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?s ?temp
WHERE {
  ?s sosa:resultTime ?t;
     sosa:hasSimpleResult ?temp.

  FILTER(?t <= "2023-01-01T00:00:00Z"^^xsd:dateTime)
}
```

**Listing 1:** SPARQL query to get the temperature and the associated un

**A solver will be needed to match a SPARQL query pattern to tree:Relations**

```
@prefix tree: <https://w3id.org/tree#> .
@prefix sosa: <http://www.w3.org/ns/sosa/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix ex: <https://example.be/> .

ex:node tree:relation _:b1 .
_b1 a tree:EqualToRelation ;
    tree:node ex:nextNode ;
    tree:value "2023-01-07T00:00:00Z"^^xsd:dateTime ;
    tree:path sosa:resultTime ;
    tree:remainingItems "21"^^xsd:integer .
```

is showing a set of triples  
tion. The relation can be  
ving boolean equation  $x =$   
 $unitTime(2023-01-07T00:00:00Z)$  where  $x$  is any variable  
inside the client SPARQL query that as the predicate  
sosa:resultTime.





# How do we prune or prioritize relations based on a certain client's input?



Client's input: in our case this will be a SPARQL query

# GUIDED LINK TRAVERSAL QUERY PROCESSING


- . We take advantage of a priori knowledge or meta knowledge (understanding) for the choice of links
- . Structure of a document 
  - . Fragmented database
- . Discovery of knowledge 
  - . Relation

# FORMALISATION



# PROBLEM

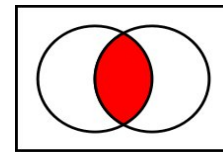
We have a boolean expression from the TREE  
Document and a boolean expression from the  
SPARQL filter

 We want to know if the combination of the two  
expression can be satisfy  
If so we follow the link

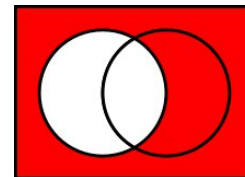
# STRATEGY OF RESOLUTION

- We calculate the domain of each sub expression and represent it as a set {lower bound, upper bound}
  - Eg:  $x > 5$
  - $\{5+dx, \text{infinity}\}$
- We concatenate the expression based on the boolean operator

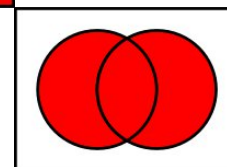
- AND



- OR

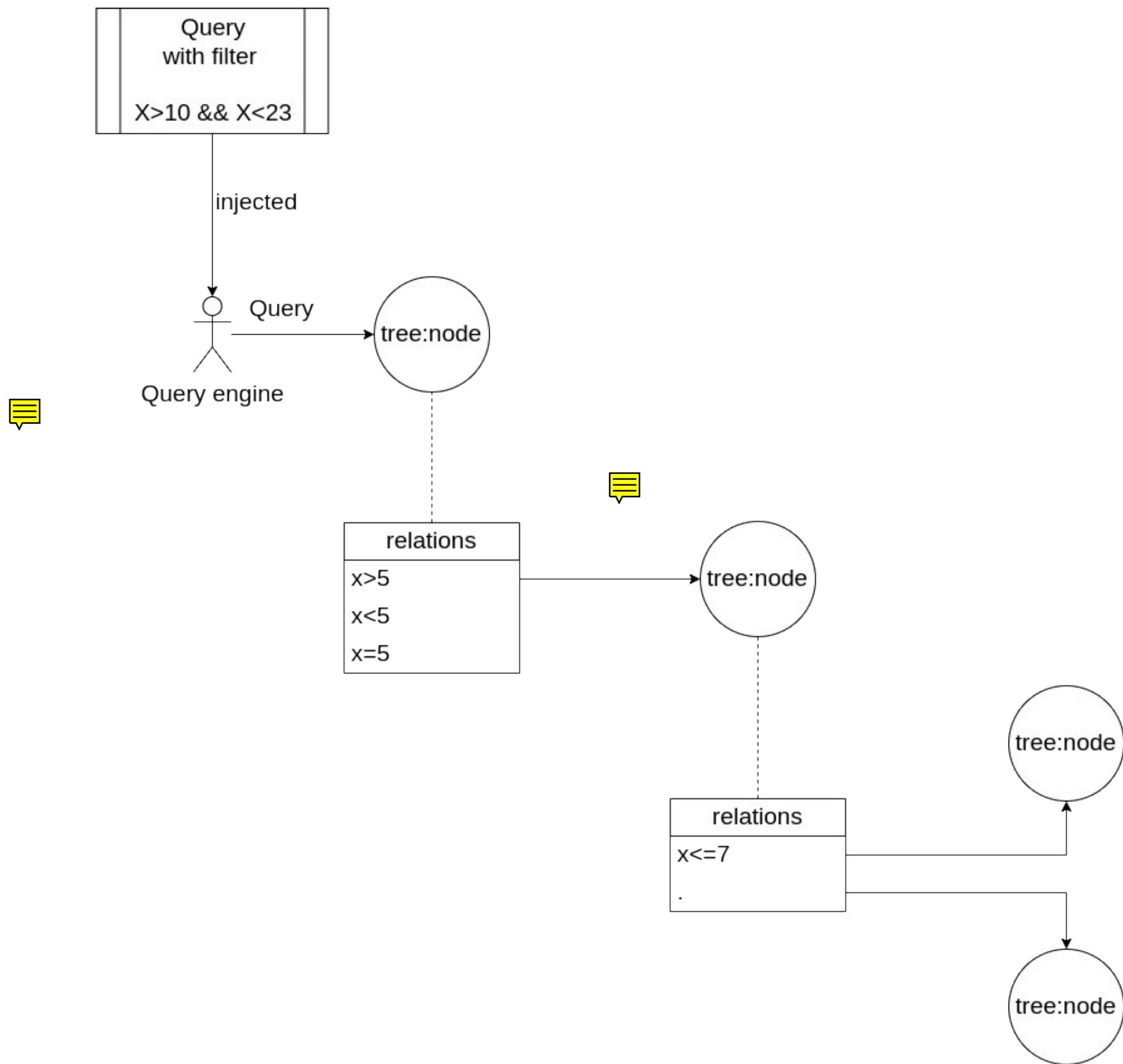


- NOT



The domain is a set of set containing the domain (subdomain) of each expression

$$\begin{aligned}
 & \circ_j \in \{U, \cap\} \\
 & p_z \in \{^c, ^1\} \\
 & (\dots (SD_1^{p_1} \circ_1 SD_2^{p_2})^{p_3} \circ_2 SD_3^{p_4} \dots) \circ_{n-1} SD_n^{p_{n+1}}
 \end{aligned}$$



# Testing query performance

- .Data: Time series, public transportation
- .Different topologies: B-tree, linked list, ...
- .Parameters
  - .Number of request
  - .Query execution time
  - .Number of results

```
1 {  
2  
3   './benchmark/config_comunica_follow_tree.json': {  
4  
5     exec_time: 2908.7810100317,  
6  
7     number_request: 267,  
8  
9     number_result: 29  
10  },  
11  
12  './benchmark/config_comunica_follow_tree_solver.json': {  
13  
14     exec_time: 252.93893498182297,  
15  
16     number_request: 20,  
17  
18     number_result: 29  
19  },  
20  
21  './benchmark/config_comunica_follow_all.json': {  
22  
23     exec_time: 2938.53779900074,  
24  
25     number_request: 374,  
26  
27     number_result: 29  
28  },  
29  
30  },  
31  
32  },  
33
```

# Future work

Benchmarking of query clients

Better visualization tool

Seeing which node we traversed

- Support for string
- View selection based on the query

