DESIGN OF DATABASE SYSTEMS FOR OPTIMIZED SPATIO-TEMPORAL QUERYING TO FACILITATE MONITORING, ANALYSING AND FORECASTING IN THE "INTERNET OF WATER"

Erik Bollen[1], Brianna R. Pagán[2], Bart Kuijpers[1], Stijn Van Hoey[3], Nele Desmet[2], Rik Hendrix[4], Jef Dams[2], and Piet Seuntjens[2]

[1]*UHasselt - Hasselt University, Databases and Theoretical Computer Science, DSI (erik.bollen@student.uhasselt.be)*

[2]*Environmental Modelling Unit, Flemish Institute for Technological Research (VITO)*

[3]*Fluves, Gent, Belgium*

[4]*Data Science Hub, Flemish Institute for Technological Research (VITO)*

Abstract

Monitoring, analysing and forecasting water systems, such as rivers, lakes and seas, is an essential part of the tasks for an environmental agency or government. In the region of Flanders, in Belgium, different organisations have united to create the ”Internet of Water” (IoW). During this project, 2500 wireless water quality sensors will be deployed in rivers, canals and lakes all over Flanders. This network of sensors will support a more accurate management of water systems by feeding real-time data. Applications include monitoring real-time water-flows, automated warnings and notifications to appropriate organisations, tracing pollution and the prediction of salinisation. In this research, we focus on the performance of spatio-temporal queries taking into account the spatial configuration of a strongly human-influenced water system and the real-time acquisition and processing of sensor data.

Objectives

The Vlaamse Hydrografische Atlas (VHA) (Fig. 1) is a set of line geometries with flow direction within segments, but no overall flow direction information.

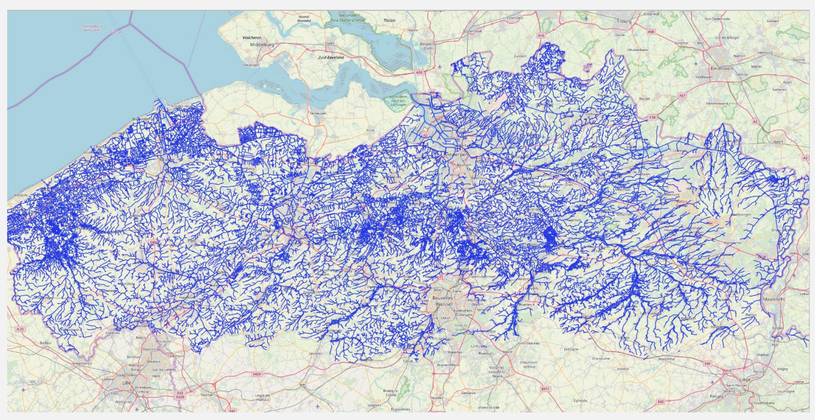

Figure 1: Overview of all major rivers in Flanders within the VHA.

Here our objectives are to:

- estimate and integrate the flow information into the dataset, resulting in a river topology;
- find flow-paths downstream and upstream (Fig. 2) given a location on the topology;
- compare and determine which database (PostgreSQL or Neo4j) is most suited for the application in respect to data handling, querying and transitive closure calculations.

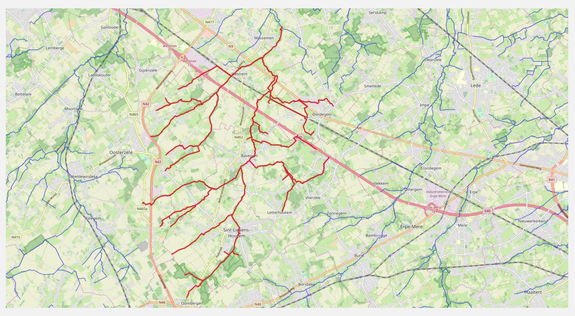

Figure 2: Example output of an upstream path query for a small basin.

Methods

Comparisons between the databases include:

- data handling ACID vs BASE,
- importing and transforming VHA to topology,
- querying the flow-path, ease (SQL vs Cypher) and speed (Fig. 3),
- visualisations of result and
- possibilities for later extensions in GIS.

For the retrieval speed test various testcases where selected out of the VHA upstream, downstream and various sizes. The time to retrieve all segments was measured in Neo4j and PostgreSQL.

```
MATCH (N:segment)-[:flowsTo*]->(M:segment)
WHERE N.vhas = id_startsegment
RETURN DISTINCT M.vhas;
```

```
WITH RECURSIVE outcome(vhas, source, target) AS (
                (SELECT vhas, source, target
                 FROM wlas
                 WHERE vhas = id_startsegment)
        UNION
                SELECT wlas.vhas, wlas.source, wlas.target
                FROM outcome, wlas
                WHERE wlas.source = outcome.target)
        SELECT vhas FROM outcome);
```

Figure 3: Downstream query expressed in Cypher (top) or SQL (bottom). The SQL statement has to make use of the more complicated “WITH RECURISVE”.

Retrieval Time Test Setup

The retrievals were tested on a local server, ProLiant MicroServer Gen8, Intel(R) Celeron(R) CPU G1610T, 4GB HP DIMM DDR3 1600 MHz memory, Intel(R) SSDSC2KW12 120GB storage device with a clean Ubuntu server 18.04 LTR. Installed versions of the software were PostgreSQL 12.2 (Ubuntu 12.2-2.pgdg18.04+1) and Neo4J Kernel 4.0.1 community edition. The testing script was written in python and executed with Python 3.6.9 (default OS install) using psycopg2 (2.8.4) and neo4j (1.7.6) packages.

Results

Action	PostgreSQL	Neo4j
GIS Data Support	Great support due to PostGIS and pgRouting	Basic support with Neo4j Spatial
Topology data model	Provided by extensions	Standard support
Importing VHA-shapefile	Many possibilities	Basic shapefile import
Conversion to Topology	Well supported	Impossible
Flow-path query	Not easy and not clear	Easy and clear
Retrieval speed	Up to 7s	Up to 3s or several minutes*
Output (as graph)	No, relations and rows	No, records of Key-value pairs

Retrieval times:

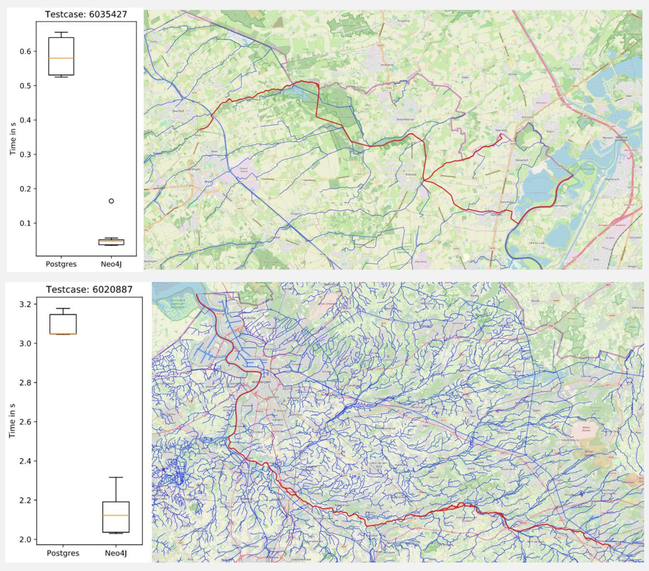

Figure 4: A small (top) and large (bottom) downstream test case used in the retrieval speed test. Left of each image are the measured retrieval times shown for each database system. Retrieval times increase with increasing size. However, Neo4J growth rates are less than PostgreSQL. The same can be seen for upstream test cases, with the exception of one shown in Figure 5.

*Here is an exception on the overall result that Neo4j is quicker than PostgreSQL. The reason for this is not clear cut, lack of sufficient main memory and limited recursion depth seem factors.

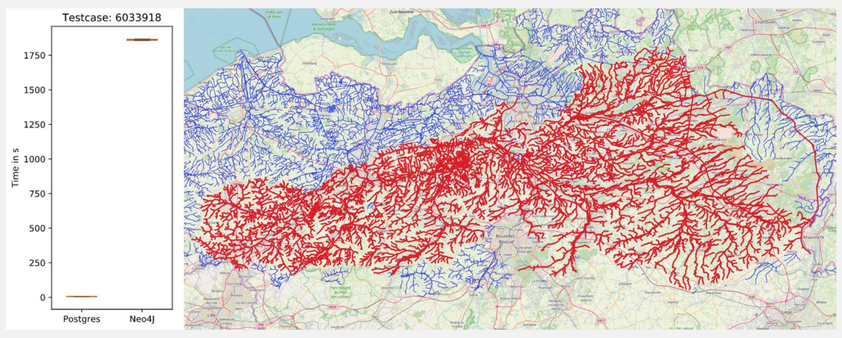

Figure 5: Showing a testcase where Neo4j is unable to cope with the task where as PostgreSQL can solve the query in 7 s.

Applications

Using this research as foundation it can facilitate:

Salinity extrapolation

The topology and overall flow information enabled data mining approaches to extrapolating gauge salinity measurements across river segments (Pagán et al., 2020; Fig. 6).

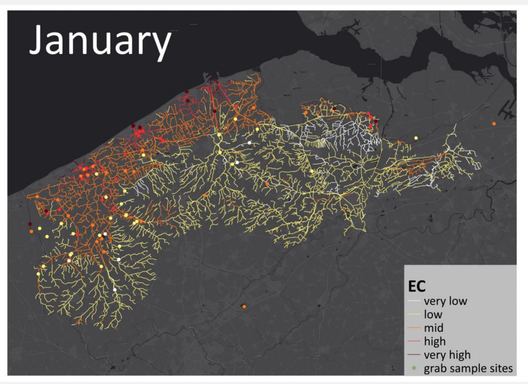

Figure 6: Showing the extrapolation of the grab-sampled salinity created by a neural network [Pagán et al., 2020] and the distance to sea made available by this work.

Discharge prediction

Future work will focus on the prediction of river discharge given a location, enabling inter-sensor validation. Results are expected to be published summer 2020.

Conclusions

Results demonstrate that PostgreSQL is better suited as it provides the most of the required functionality and data conversion tools required. While Neo4j has better performance and ease of querying compared to PostgreSQL, Neo4j can exhibit severe performance issues in large cases. While Neo4j is useful in controlled smaller projects, the uncertainty for larger applications in production environments makes it less suitable for the IoW.

References:

[Pagán et al. 2020] Brianna Pagán, Nele Desmet, Piet Seuntjens, Erik Bollen, & Bart Kuijpers (2020). Data driven methods for real time flood, drought and water quality monitoring: applications for Internet of Water, EGU 2020

CC BY