A rule-based geospatial reasoning system for trip price calculations

Stefan Schenk

Supervisor: Willem Brouwer

Advisor: Mewis Koeman

Department of Software Engineering
Amsterdam University of Applied Sciences

This dissertation is submitted for the degree of
Bachelor Software Engineering

May 2018

Todo list

Table of contents

Chapter 2

Encoding Locations

2.1 Introduction

Encoding of locations has historically been of great importance, and is always being modernized. This chapter explains the general definition of locations, which types of locations are important for this project, and how to represent these locations so that they are universally interpretable and do not rely on a postal code system.

2.2 A Brief History Of Geographic Locations

A location is roughly described as a place or position. Throughout history, various navigational techniques and tools like the sextant, nautical chart and marinner's compass were used, measuring the altitude of the North Star to determine the latitude ϕ, in conjunction with a chronometer to determine the longitude λ of a location on the Earth's surface. The combination of coordinates is a distinct encoding of a location. This particular system is still commonly used today.

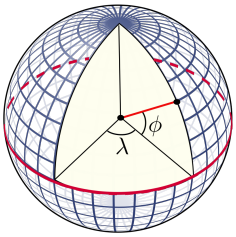

Fig. 2.1 A perspective view of the Earth showing how latitude and longitude are defined on a spherical model.

The history of this encoding goes way back to when it was first proposed in the 3rd century BC by Eratosthenes. He invented the discipline of geography, and was known for also being the first person to calculate the circumference of the Earth with remarkable accuracy. Today, navigation relies on satellites that are capable of providing information to determine a location with a precision of 9 meters. Hybrid methods using cell towers, Wi-Fi Location Services, and the new Galileo global navigation sattelite system, provide tracking with a precision down to the meter range. These locations are ordinarily communicated using the same established latitude and longitude encoding. For a human being, it is not practical to exchange day-to-day locations as geographical coordinates. For that, addresses much more suitable, but can be ambiguous, imprecise, and inconsistent in format. Addresses commonly make use of Postal Code systems, which have reliably been assigned to geographical areas with the purpose of sorting mail. Although even today, there are countries that do not have a Postal Code system. This forces the legacy system to support addresses for the fixed pricing functionality as well. In contrast to the geographic coordinate system, postal codes describe streets and areas of varying shapes and sizes. A location being roughly described as a place or position, can be decomposed as an abstract term to describe physical or imaginary areas with varying radiusses and shapes. You could prepend 'the location of' to the following terms as an example: America, the birthplace of Sokrates, Wall Street, the center of the universe, the Laryngeal Nerve of the Giraffe, churches in the Netherlands. The final example presents the main challenge of this project, how to communicate the location of a collection with points or areas of differing shapes and sizes that may overlap?

2.3 Requisite Location Types

While setting up a backlog for a project, a shared knowledge about the terminology used in the issues must be achieved in order to collaborate effectively. Words or symbols do not have an absolute meaning, and ambiguity of abstract linguistic terms should be elucidated. In section 3.2.1 of Appendix A, an agreement was made on what the terms "area" and "point" meant. The MySQL documentation notes that "The term most commonly used is geometry, defined as a point or an aggregate of points representing anything in the world that has a location." in [2]. During the process of implementing TPS, the definitions of a location have been refined to represent a common and useful understanding.

2.3.1 The Point

A point is a unique place expressed as a distinct coordinate pair. An address in the legacy system could be translated to a point. For example, the address that is tied to Schiphol arrival is: Aankomstpassage, 1118 AX Schiphol Centrum. The point that encodes this location is (52.308891, 4.760900). This location is contained in the set of all possible points on Earth, which could be expressed using set builder notation:

$$P = \{(\phi, \lambda) \in \mathbb{R}^2 | -90 < \phi < 90, -180 < \lambda < 180\}$$

$$(52.308891, 4.760900) \in P$$

A point itself can not be used to match whether another point is contained within it, because the probability of a match is infitesimal. Only when decimals were disregarded to decrease the precision of a point it would be possible, in which case it would still not be usefull in this application, because the imprecise point would be a square.

2.3.2 The Area

An area is a set of points points with an infinite granularity. This definition allows for an area to have holes inside them, consist of other locations and contain other locations, and be infinitely precise. The most useful property of this area is to check whether a point is contained within the area, or which areas contain a given point. For this to be the case, the points must be packed together as it would form a shape. This definition, however conceptually valuable, will not be of much practical use. For example, P is an infinitely long set of coordinates, an area that represents the earths surface. If ϕ ranged between 0 and 90, the set sould describe all points located in the northern hemisphere, but would still be infinitely long. Checking whether a given point is contained by checking an infinite amount of real number pairs will take an infinite amount of time in the worst case scenario. An area can be described as a subset of all points:

$$a_1 \subseteq P$$

The set of all possible areas can be defined by the power set of P:

$$A = \mathscr{P}(P)$$

such that a subset of points, called an area, is in all possible areas A:

$$a_1 \in A$$

At the equator, 1 degree is 111320m, so 0.000001 degrees is around 11cm. Six decimal places will be sufficient for location matching for this application. But even when reducing coordinates to having six decimal places, it would be impractical. For this reason, it is more realistic to only describe the rough edges of an area using a polygon shape. Instead of checking for a single point in a non-terminating iteration over all points in an area, a mathematical calculations could be used to check whether a unique point is contained within the polygon.

2.3.3 Postal Codes, Addresses, and Polygons

All postal codes that start with a ten describe the city of Amsterdam, the entire area of Amsterdam can be drawn as a big polygon containing all the postal codes that start with a ten.

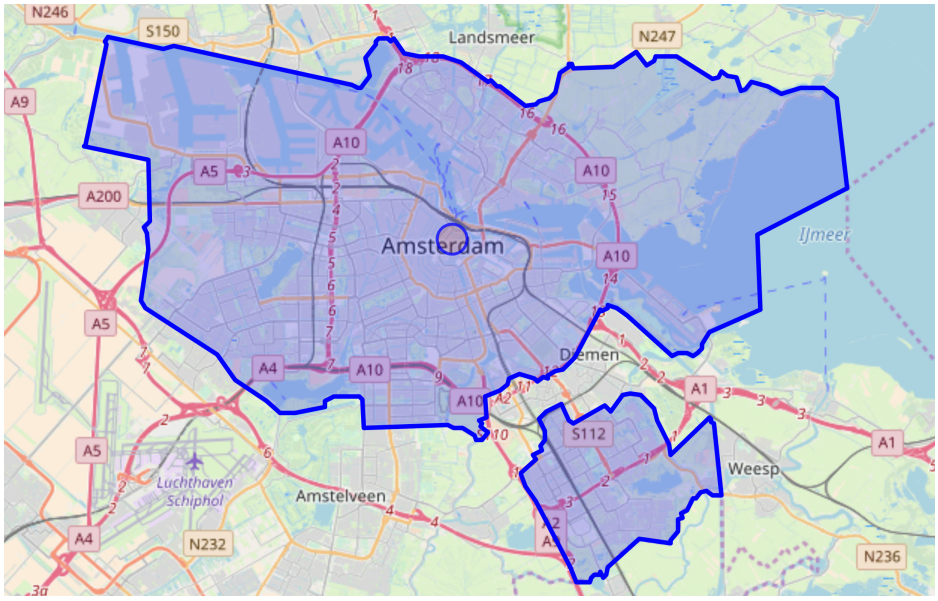

Fig. 2.2 Amsterdam - A single location contrived of multiple locations.

In reverse, this procedure would not work. If a polygon was drawn cutting Amsterdam in half diagonally, a single postal code pattern would never be flexible or precise enough to be able to describe the boundaries of the polygon. One big taxi company making use of taxiID's legacy system is located in the United Arab Emirates. This company would not be able to

convert anything at all, because the United Arab Emirates does not have a postal code system to begin with.

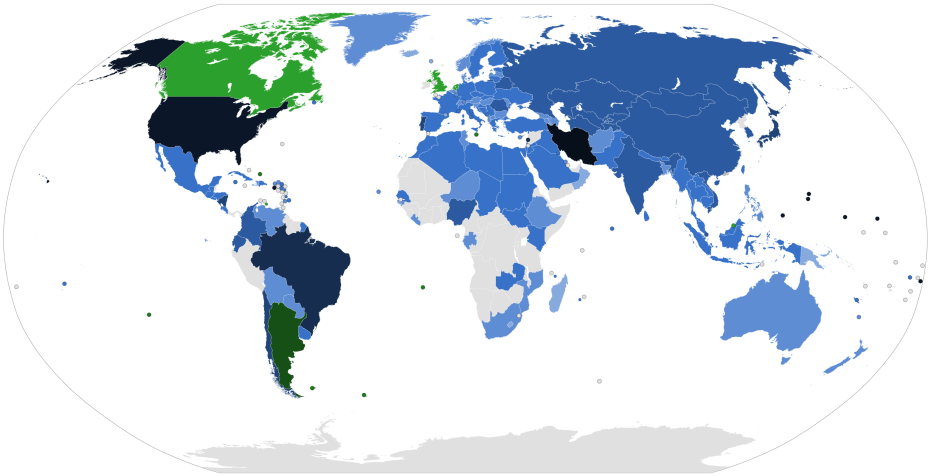

Fig. 2.3 Postal codes by country with amount of digits ranging from three digits (lightblue) to eight digits (darkgreen) and no postal code system (gray).

When a trip is booked in the United Arab Emirates, the only information provided as a destination or departure location are geospatial coordinates and addresses. These addresses are compared with long lists of matching addresses in the legacy database, as previously described with postal codes. Addresses and postal code systems do not provide universally interpretable and precise encodings of locations, especially for the locations that matter for this project: points and areas. Addresses can be ambiguous, addresses and postal codes can be imprecise, postal codes are not uniform, and some countries do not have a postal code system. In contrast, polygons would provide unique and precise location definition that is uniform and universal. But postal codes and addresses have been useful in the legacy system. When moving to other encoding techniques, this usefulness must be preserved.

2.3.4 Requirements for Location Matching

If the following statements are true for a given location encoding using the definitions of the Point and Area, the location encoding is useful and able to operate independently from the postal code and address systems.

Nr	Description
1.	Every location is stored in a database as a single entity
2.	Locations can consist of multiple locations (see figure 2.2)
3.	A deterministic predicate of whether a location is fully contained within a location is achievable
4.	A deterministic method of finding all locations containing a single location can be used
5.	A method of determining precedence of location in case of overlap must always yield one result, and discard all others
6.	Locations must be importable from external sources

Table 2.1 Location Matching Requirements

2.4 Literature Review

Many spatial database systems support a basic Geometry hierarchy of Points, Polygons, MultiPoint and MultiPolygon Classes, as described in the OGC [3] and ISO 19125 [4] standard. These spatial datatypes would could be stored as single entities. The MultiPolygon class is able to support multiple polygons to be stored as a single entity. The standard provides containment predicate, and methods to distinguish larger locations from smaller ones, which could be used in precedence checks. Databases like MySQL and PostgreSQL provide similar functionalities that adhere to the OGC standard, while MongoDB has its own implementation. These functionalities can be seen as a straight forward approach in solving geometry based queries. What 3 words, a multi-award winning global addressing system, bases 3m x 3m squares, covering the planet, on a combination of three words. In their whitepaper: "Efficient and future-proof", CEO Chris Cheldrick explains how locations can be communicated more effectively by describing a three by three meter areas using three words [5]. The what3words API offers functionalities that can find what3word geocodings near a specified latitude and longitude location. It is also able to find results within a clamped area, as documented in [6]. This system could be used to retrieve area's as three words in a database with given clamp parameters, which could then be stored in the database as single entities. This system would solve the problem of having ambiguity in address or postal code systems, and it is also very accurate. But in order to provide complex polygon matching, a spatial database system is still required. If a radius was assigned to the centroid of the shape that is formed by the boundaries of the street, neighbourhood, province or country, the encoding could technically be defined as a what3words query. A combination of clustering

of points around what3word encodings could provide a system that is far less accurate, but is usable.

kks32: Keep trying to find if other people found solutions for determining precedence of polygon matches

http://geoawesomeness.com/discrete-global-grid-system-dggs-new-reference-system/

2.5 Database Prerequisites

The database that is used must be able to determine whether a polygon contains a given point, and must be able to aggregate all points that are contained within a given polygon. The scenario presented in image 2.4 should be replicable.

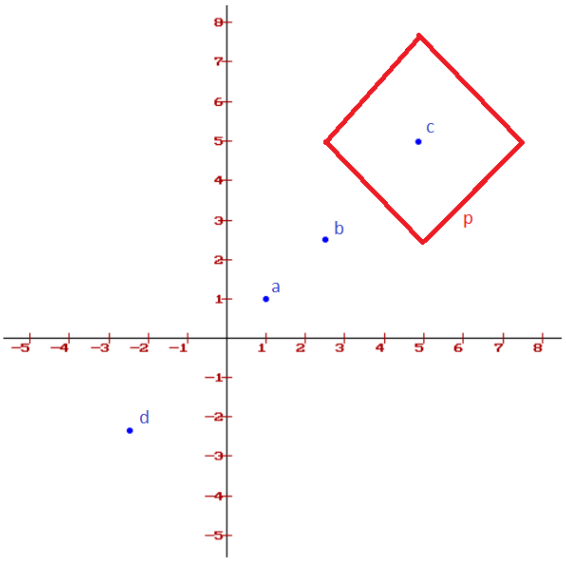

Fig. 2.4 Four Points, one Polygon p containing Point c.

As contained in Appendix A, this example provides a proof that a minimal requirement is satisfied, so that a list of candidate Database Management Systems could be constructed.

2.5.1 OpenGIS Compatible databases

MYSQL's innate integrity is a good reason to opt for a full MYSQL database setup. MariaDB is a fork of MYSQL that performs better according to benchmarks, however they don't always translate to real life situations. It's easy to migrate from MYSQL to MariaDB, so choosing MYSQL at first could be preferable as an instance of MYSQL is already used at

TaxiID. PostgreSQL offers a spatial database extender for that is OpenGIS compliant called PostGIS that adds support for geographic objects and location queries.

All spatial data types inherit properties such as type and spatial reference identifier (SRID). For rigorous documentation, both PostGIS documentation [7] and MYSQL documentation [8] could be consulted. When a generic geometry column, or point column is created, points can be inserted as shown in snippet 2.1.

```
START TRANSACTION;
SET @a = ST_GeomFromText('POINT(1 1)');
INSERT INTO point (point) VALUES (@a);
SET @b = ST_GeomFromText('POINT(2.5 2.5)');
INSERT INTO point (point) VALUES (@b);
SET @c = ST_GeomFromText('POINT(5 5)');
INSERT INTO point (point) VALUES (@c);
SET @d = ST_GeomFromText('POINT(-2.5 -2.5)');
INSERT INTO point (point) VALUES (@a);
# also insert @b, @c, and @d
COMMIT;

START TRANSACTION;
# First and last point should be the same
SET @a = PolygonFromText('POLYGON((2.5 5,5 7.5,7.5 5,5 2.5,2.5 5))');
INSERT INTO polygon (polygon) VALUES (@a);
COMMIT;
```

Listing 2.1 Insert four points and one polygon in MySQL.

It is evident that c is contained in p. To determine which points are contained in p, the function as seen in Snippet 2.2 can be used, which returns the point with coordinates $[5,5]$ as expected.

```
// All points contained in polygon
SELECT ST_ASTEXT(POINT)
FROM POINT
WHERE
ST_CONTAINS(
  (
    SELECT POLYGON
    FROM POLYGON
    WHERE id = 1
  ),
  POINT
)

// All polygons containing point
SELECT ST_ASTEXT(POLYGON)
FROM POLYGON, POINT
WHERE
  POINT.id = 3 AND ST_CONTAINS(
    POLYGON.polygon,
    POINT.point
  )
```

Listing 2.2 Select points contained in polygon, and all polygons containing a point in MySQL.

2.5.2 OpenGIS Incompatible databases

MongoDB doesn't offer OpenGIS implementations but has geospatial query operators that may provide enough functionalities for current requirements [9]. The argument for choosing one over the other depends on the vast differences between SQL and NoSQL, next to performance and extensiveness of geospatial features. The setup displayed in image 2.4 is recreated in MongoDB using queries shown in snippets 2.3 and 2.4.

```
db.point.insertMany([
  { shape: { type: "Point", coordinates: [1, 1] } },
  { shape: { type: "Point", coordinates: [2.5, 2.5] } },
  { shape: { type: "Point", coordinates: [5, 5] } },
  { shape: { type: "Point", coordinates: [-2.5, -2.5] } },
])

db.polygon.insert({
  shape: {
    type: "Polygon",
    coordinates: [ [ [2.5, 5], [5, 7.5], [7.5, 5], [5, 2.5], [2.5, 5] ] ]
  }
})

db.point.createIndex({ 'shape': '2dsphere' })
db.polygon.createIndex({ 'shape': '2dsphere' })
```

Listing 2.3 Insert four points and one polygon in MongoDB.

```
// All points contained in polygon
var p = db.polygon.find({})

db.point.find({
  shape: {
    $geoWithin: {
      $polygon: [
        [2.5, 5],
        [5, 7.5],
        [7.5, 5],
        [5, 2.5],
        [2.5, 5]
      ]
    }
  }
})

// All polygons containing point
var p = db.point.findOne({ coordinates: [5, 5] })

```

```
db.polygon.find({
  shape: {
    $geoIntersects: {
      $geometry: {
        type: "Point",
        coordinates: [5, 5]
      }
    }
  }
})
```

Listing 2.4 Select points contained in polygon, and all polygons containing a point in MongoDB.

Next to database solutions for this requirement, services exist that are capable of geofencing. Although these services may not be free, and the added dependencies restrict extensibility.

2.6 Performance and Clustering Trade-offs

Agarwal and Rajan state that NoSQL take advantage of cheap memory and processing power, thereby handling the four V's of big data more effectively, but lack the robustness over SQL databases in [10]. The report dives deeper into spatial queries and concludes that their tests suggest that MongoDB performs better by an average factor of 10, which increases exponentially as the data size increases, but lack many spatial functions that OpenGIS supports. Although improvements have been made, as per [11]. After the paper Schmid et al. 2015 [12] was published. The team argues that clustering is much easier in MongoDB, which may be important in the future when the company grows. As the required functionalities exist in both SQL and NoSQL, it is beneficial to opt for MongoDB for its performance and alignment with the teams experience. Although if robustness is desired, or extra GIS functionalities required, SQL should be taken into consideration.

References

[1] U. T. Inc. (2011) The uber story. [Online]. Available: https://www.uber.com/en-NL/our-story/

[2] (2018) Mysql 5.7 reference manual - spatial data types. [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/spatial-types.html

[3] J. R. Herring, "Implementation standard for geographic information - simple feature access - part 1: Common architecture." May 2011. [Online]. Available: http://portal.opengeospatial.org/files/?artifact_id=25355

[4] (2004) Geographic information – simple feature access – part 1: Common architecture. [Online]. Available: https://www.iso.org/standard/40114.html

[5] C. Sheldrick, "Efficient and future-proof." April 2018. [Online]. Available: https://what3words.com/2018/04/read-our-white-paper-efficient-and-future-proof/

[6] (2016) what3words - restful api. [Online]. Available: https://docs.what3words.com/api/v2/#autosuggest-params

[7] (2018) Postgis 2.4.5dev manual. [Online]. Available: https://postgis.net/docs/manual-2.4/using_postgis_dbmanagement.html#PostGIS_GeographyVSGeometry

[8] (2018) Mysql 5.7 reference manual - geometry class. [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/gis-class-geometry.html

[9] (2018) "geospatial query operators — mongodb manual 3.6. [Online]. Available: https://docs.mongodb.com/manual/reference/operator/query-geospatial/

[10] K. R. Sarthak Agarwal, "Analyzing the performance of nosql vs. sql databases for spatial and aggregate queries." September 2017. [Online]. Available: https://scholarworks.umass.edu/cgi/viewcontent.cgi?article=1028&context=foss4g

[11] (2015) Geospatial performance improvements in mongodb 3.2. [Online]. Available: https://www.mongodb.com/blog/post/geospatial-performance-improvements-in-mongodb-3-2

[12] W. R. Stephan Schmid, Eszter Galicz, "Performance investigation of selected sql and nosql databases." June 2015. [Online]. Available: https://agile-online.org/conference_paper/cds/agile_2015/shortpapers/68/68_Paper_in_PDF.pdf

[13] (2011) Iso/iec 25010:2011. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en

[14] I. K. Center. (2018) Three-tier architectures. [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSAW57_8.5.5/com.ibm.websphere.nd.doc/ae/covr_3-tier.html

[15] R. Stephens, *Beginning Software Engineering*. John Wiley & Sons, 2015.

[16] P. Hauer. (2015) Microservices in a nutshell. pros and cons. [Online]. Available: https://blog.philipphauer.de/microservices-nutshell-pros-cons/

[17] J. Stenberg. (2014) Experiences from failing with microservices. [Online]. Available: https://www.infoq.com/news/2014/08/failing-microservices

[18] (2018) Mocha - the fun, simple, flexible javascript test framework. [Online]. Available: https://mochajs.org

[19] (2018) Chai assertion library. [Online]. Available: http://chaijs.com

List of figures

List of tables