Instantly joining 1B points with hundreds of polygons

Perlen der Informatik

Andreas Kipf, 2019-07-12

Technical University of Munich
Geospatial Join Problem

Points
• E.g., GPS positions

Polygons
• Typically disjoint political boundaries such as neighborhoods

Point/polygon join
• Which polygon does a given point lie in?
• Summary statistics for all points that lie in a certain polygon
Traditional Approach

1. Construct an R-tree index on the polygons’ MBRs
2. Perform an index nested loop join
Our Approach

Skip the expensive refinement phase

- Referred to as true hit filtering
- Invented in the 90s
- Only a few systems have used this idea in the last two decades
Google S2

Maps every point on Earth onto a cube

Recursively subdivides the cube

Identifies each cm² on Earth with 64 bits
Polygon Approximations

**Covering**
- A collection of non-uniform cells *covering* a polygon

**Interior covering**
- A collection of non-uniform cells *lying fully within* a polygon
Polygon Approximations

Super covering

• A combination of multiple coverings and interior coverings with each cell mapping to one or many polygons

Cell types

• Blue cells are covering cells of single polygons
• Red cells are covering cells of multiple polygons
• Green cells are interior cells of single polygons
Data Structure

Radix tree
- A trie data structure
- Fanout of 256
- Inlined payloads

Lookup table
- A single 32-bit vector storing large payloads
Evaluation

**Evaluation system**
- 2x Intel(R) Xeon(R) CPU E5-2680 v4 CPU (2.40 GHz, 3.30 GHz turbo)
- 256 GB DDR3 RAM
- Ubuntu 16.04

**Points**
- NYC taxi rides (1B)

**Polygons**
- NYC boroughs (5)
- NYC neighborhoods (290)
- NYC census blocks (40k)
Evaluation

**Throughput in M points/s**

<table>
<thead>
<tr>
<th></th>
<th>boroughs</th>
<th>neighborhoods</th>
<th>census blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostGIS</td>
<td>0.39</td>
<td>1.09</td>
<td>0.69</td>
</tr>
<tr>
<td>Spark Magellan</td>
<td>0.88</td>
<td>4.57</td>
<td>2.24</td>
</tr>
<tr>
<td>R-tree</td>
<td>3.88</td>
<td>61.2</td>
<td>28.9</td>
</tr>
<tr>
<td>exact</td>
<td>3735</td>
<td>1459</td>
<td>431</td>
</tr>
<tr>
<td>approx.</td>
<td>4532</td>
<td>2280</td>
<td>874</td>
</tr>
</tbody>
</table>

**This is more than 1B points against 290 polygons in < 1 sec**
Why does this all matter?

It allows you to forecast the stock market!

Satellite image processing companies provide a virtual representation of the real world

- They extract features (e.g., cars) from satellite images and repeatedly join these features with existing datasets (e.g., US parking lots)
- Show that they can forecast the stock price of US retail chains

"Orbital Insight uses deep learning algorithms to accurately identify cars from satellite images at 55,000+ parking lots of major retail chains across the U.S."