Conversion of OSM data to RDF Data and the use of Simplified Geometries when computing Spatial Relations

Iradj Solouk
Part I
Problems
Problems

● Turning OSM Data to RDF
  ○ Memory efficiency
Problems

- Turning OSM Data to RDF
  - Memory efficiency
  - Structure of OSM Data
Structure of OSM Data

- Nodes, Ways, Relations
- IDs
- May have OSM Tags
  - (key, value) pairs like “Amenity = restaurant”
Structure of OSM Data

- Nodes, Ways, Relations
- IDs
- May have OSM Tags
  - (key, value) pairs like “Amenity = restaurant”
- Nodes are points in space (longitude, latitude)
- Ways are polylines/polygons
  - No lon/lat, but IDs to nodes
- Relations are groups of possibly Nodes, Ways and/or Relations
  - ID and role pairs are given as references
Problems

- Turning OSM Data to RDF
  - Memory efficiency
  - Structure of OSM Data
    - Computing explicit geometry by resolving references
Example of converted element

<relation id="13038953" ...>

<member type="node" ref="8964237082" role="guidepost"/>
<member type="way" ref="220021661" role=""/>
<member type="way" ref="34611919" role=""/>
<member type="node" ref="8964237098" role="guidepost"/>
...
</relation>
Example of converted element

<relation id="13038953" ...>

 <member type="node" ref="8964237082" role="guidepost"/>
 <member type="way" ref="220021661" role=""/>
 <member type="way" ref="34611919" role=""/>
 <member type="node" ref="8964237098" role="guidepost"/>
 ...
</relation>

<osmrel:13038953 geo:hasGeometry "GEOMETRYCOLLECTION(POINT(7.986672 48.506398),
 POINT(7.989988 48.502834),  LINESTRING(7.986689 48.506284,7.986729
48.506251,7.987331 48.505752,7.987984 48.505333,7.988140 48.505199,7.988266
48.504216,7.988844 48.504160,7.989198 48.503798,7.989576 48.503397,7.990016
48.502844))"^^geo:wktLiteral .
Problems

- Turning OSM Data to RDF
  - Memory efficiency
  - Structure of OSM Data
    - Computing explicit geometry by resolving references
    - Computing spatial relations: contains & within
Problems

- Turning OSM Data to RDF
  - Memory efficiency
  - Structure of OSM Data
    - Computing explicit geometry by resolving references
    - Computing spatial relations: contains & within
    - Computing way-clusters
Way-cluster

- Streets are fragmented
Possible Application - Overpass Turbo

area[name="Altstadt"];
node
    [amenity=fountain]
    (area);
out;
rel(1960176);
way(r);
out geom;
Possible Application - Overpass Turbo

area[name="Altstadt"]; node [amenity=fountain] (area); out;
rel(1960176); way(r); out geom;

SELECT ?osm_id ?hasgeometry WHERE {
  ?osm_id osmt:amenity "fountain".
}
Part II

Solutions
Solutions

- Turning OSM Data to RDF
  - Memory efficiency
  - Structure of OSM Data
    - Computing explicit geometry by resolving references
    - Computing spatial relations: contains & within
    - Computing way-clusters
Memory efficiency

- Serialize the data with STXXL
  - Tries to mimic STL containers and algorithms
  - Not in-memory but on-disk
  - Sorting is possible, but only POD structures can be used
Solutions

● Turning OSM Data to RDF
   ○ Memory efficiency
   ○ Structure of OSM Data
     ■ Computing explicit geometry by resolving references
     ■ Computing spatial relations: contains & within
     ■ Computing way-clusters
Resolving geometry

- E.g. resolving geometry of ways
  - Sort node vector (containing lon/lat) by ID
  - Sort way vector (containing reference ID) by reference ID
Solutions

- Turning OSM Data to RDF
  - Memory efficiency
  - Structure of OSM Data
    - Computing explicit geometry by resolving references
    - Computing spatial relations: contains & within
    - Computing way-clusters
Spatial relations

- Spatial relations only wrt. elements tagged as *administrative boundaries*
- Using Boosts Rtree as a spatial index
Spatial relations

- Spatial relations only wrt. elements tagged as *administrative boundaries*
- Using Boosts Rtree as a spatial index
- Using overestimated and underestimated polygon to infer spatial relations: contains & within
  - Modified Ramer-Douglas-Peucker Algorithm
  - Simplified Polygons have less points
- Use directed acyclic graph for transitive comparison
Solutions

- Turning OSM Data to RDF
  - Memory efficiency
  - Structure of OSM Data
    - Computing explicit geometry by resolving references
    - Computing spatial relations: contains & within
    - Computing way-clusters
Way-clusters

- Sort wayrefs by name ID and reference ID
- Find the connected components
- Intersect tags for connected components
Part III
Evaluation
### Point reduction from using simplified polygons

<table>
<thead>
<tr>
<th></th>
<th>Freiburg</th>
<th>Baden-Württemberg</th>
<th>Germany</th>
<th>Europe</th>
</tr>
</thead>
<tbody>
<tr>
<td>averaged point reduction for overestimations</td>
<td>82.4%</td>
<td>81.6%</td>
<td>76%</td>
<td>74.1%</td>
</tr>
<tr>
<td>averaged point reduction for underestimation</td>
<td>76%</td>
<td>74.79%</td>
<td>69.45%</td>
<td>71.8%</td>
</tr>
</tbody>
</table>
### Coverage by simplified polygons

<table>
<thead>
<tr>
<th></th>
<th>Freiburg</th>
<th>Baden-Württemberg</th>
<th>Germany</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepts via underestimation</td>
<td>1035909</td>
<td>2976069</td>
<td>19637209</td>
</tr>
<tr>
<td>Rejections via overestimation</td>
<td>1964445</td>
<td>9108820</td>
<td>59858222</td>
</tr>
<tr>
<td>Regular accepts/rejections</td>
<td>251566</td>
<td>1219006</td>
<td>13687931</td>
</tr>
<tr>
<td>Relative coverage by simplifications</td>
<td>92.26%</td>
<td>90.84%</td>
<td>85.31%</td>
</tr>
</tbody>
</table>
## Runtimes

<table>
<thead>
<tr>
<th></th>
<th>Freiburg</th>
<th>Baden-Württemberg</th>
<th>Germany</th>
<th>Europe*</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Printing tags</em></td>
<td>3min</td>
<td>12min</td>
<td>88min</td>
<td>678min</td>
</tr>
<tr>
<td><em>Compute boundaries</em></td>
<td>5s</td>
<td>34s</td>
<td>9min</td>
<td>86min</td>
</tr>
<tr>
<td><em>Create RTree and DAG</em></td>
<td>6s</td>
<td>1min43s</td>
<td>58min</td>
<td>373min</td>
</tr>
<tr>
<td><em>Store and process OSM nodes</em></td>
<td>4min</td>
<td>24min</td>
<td>828min</td>
<td>22min</td>
</tr>
<tr>
<td></td>
<td>Freiburg</td>
<td>Baden-Württemberg</td>
<td>Germany</td>
<td>Europe*</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------</td>
<td>-------------------</td>
<td>------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Compute way-clusters</td>
<td>13s</td>
<td>1min23s</td>
<td>14min30s</td>
<td>264min</td>
</tr>
<tr>
<td>Process and print OSM ways</td>
<td>7min41s</td>
<td>64min</td>
<td>2458min</td>
<td>325min</td>
</tr>
<tr>
<td>Process and print OSM relations</td>
<td>10min</td>
<td>79min</td>
<td>1741min</td>
<td>276min</td>
</tr>
<tr>
<td>Net Runtime</td>
<td>23min</td>
<td>184min</td>
<td>5237min</td>
<td>2021min</td>
</tr>
</tbody>
</table>
Thank you for your attention!