A Data Integration Approach To Dynamically Fusing Geospatial Sources

Snehal Thakkar
Ph. D. Dissertation
August 2007
Outline

- Introduction & motivation
- Quality-driven Geospatial Mediator (QGM)
  - Representing content and quality
  - Automatic source description generation
    - Content
    - Quality
  - Quality-driven query answering
  - Plan execution
- Related work
- Conclusions & future work
Introduction & Motivation

- Many disaster response and urban planning require integrated view of geospatial data
Introduction & Motivation

- Many disaster response and urban planning require integrated view of geospatial data
- Manually integrating geospatial data from a large number of sources is very hard
Introduction & Motivation

- Many disaster response and urban planning require an integrated view of geospatial data.
- Manually integrating geospatial data from a large number of sources is very hard.
- There is a need for a geospatial data integration framework that:
  - Automatically generates representations of sources
  - Dynamically provides high quality data
Thesis Statement

- This thesis demonstrates that by discovering geospatial sources available on the web, automatically learning the representations of both the content and the quality of data provided by the discovered sources, and exploiting the representations of the sources during query answering we can provide high quality geospatial data in response to user queries.
Outline

- Introduction & motivation
- Quality-driven Geospatial Mediator (QGM)
  - Representing content and quality
    - Automatic source description generation
      - Content
      - Quality
    - Quality-driven query answering
    - Plan execution
- Related work
- Conclusions & future work
Representation: Domain Concepts

- Content
  - Set of domain concepts by merging FGDC, NGA, and NationalMap concepts

- Quality
  - Similar hierarchy for quality of data for each domain concept
    - E.g. Road → RoadQuality
Representation: Domain Concept Attributes

- **Vector**
  - source, type, format, cs, bbox, vectorobj
- **Raster**
  - source, type, format, cs, bbox, size, resolution, rasterobj
- **VectorQuality**
  - source, type, date, completeness, resolution, horizontalaccuracy, verticalaccuracy, vectorswithinaccuracybounds, attributecompleteness
- **RasterQuality**
  - source, type, date, completeness, originalresolution, multispectral
Representation: Source Descriptions

- Source represented using two relations
  - Content
  - Quality
- Datalog descriptions
  - Content
    - Type of data: domain relation in the body
    - Coverage specified using constraints with spatial operations
  - Quality
    - Facts specifying the quality
    - Rule defining the relationship with corresponding quality relation

```
NavteqRoads(bbbox, vectorobj):-
  Roads(type, format, cs, bbox, source, vectorobj)^bbox coveredby
\ `[[33,-117],[34,-118]]`
  source = `Navteq`
  type = `Roads`
  format = `Shapefile`
  cs = `EPSG:4326`
```
Source represented using two relations
- Content
- Quality

Datalog descriptions
- Content
  - Type of data: domain relation in the body
  - Coverage specified using constraints with spatial operations
- Quality
  - Facts specifying the quality
  - Rule defining the relationship with corresponding quality relation

```
NavteqRoadsQuality(res, date, horiz-acc, vert-acc, vectorsin-acc-bounds, attr-comp, completeness):-
  RoadQuality(source, type, res, date, horiz-acc, vert-acc, vectorsin-acc-bounds, attr-comp, completeness)^
  source = `Navteq'
  type = `Roads'

NavteqRoadsQuality(5,1/1/2005,3.6,3.6,85%,90%,96%)
```
Representation: Queries

- Expressed by Datalog rules
- Three parts: data, quality, combination
  - Predicates allowed
    - Domain relations
    - Operations
      - Spatial selection
        - intersects, coveredby, disjoint
      - Aggregate
        - pack, unpack, sum, average,
        - min, max, skylinemin, skylinemax
      - Mathematical
        - add, subtract, multiply, divide
  - Order Constraints
    - e.g. completeness > 50
Find **road** vector data covering the bounding box `[[33,-115],[34,-116]]′ with **completeness** over 50%.

Q1(vectorobj, completeness):-
  Q1Data(type, source, vectorobj)^
  Q1Quality(type, source, completeness)

Q1Data(type, source, vectorobj):-
  Roads(type, format, cs, bbox, source, vectorobj)^
  bbox coveredby `[[33,-115],[34,-116]]′

Q1Quality(type, source, completeness):-
  RoadQuality(source, type, res, date, horiz-acc, vert-acc, vectorsin-acc-bounds, attr-comp, completeness)^
  completeness > 50
Find road vector data covering the bounding box \`[[33,-115],[34,-116]]\’ with the highest completeness

Q2(vectorobj, completeness):-
  Q2Data(type, source, vectorobj)^
  Q2Quality(type, source, completeness)

Q2Data(type, source, vectorobj):-
  Roads(type, format, cs, bbox, source, vectorobj)^
  bbox coveredby \`[[33,-115],[34,-116]]\’

Q2Quality(type, source, completeness):-
  RoadQuality(source, type, res, date, horiz-acc, vert-acc, vectorsin-acc-bounds, attr-comp, completeness)^
  pack(completeness, packedcompleteness)^
  max(packedcompleteness, maxcompleteness)^
  maxcompleteness = completeness
Find satellite image and road vector data covering bounding box `[[33,-116],[34,-117]]` such that both the resolution and date differences are minimized

Q3(imageobj, vectorobj, resdiff, datediff):-
  Q3Data(itype, isource, vtype, vsource, imageobj, vectorobj)^
  Q3Quality(itype, isource, vtype, vsource, resdiff, datediff)

Q3Data(itype, isource, vtype, vsource, imageobj, vectorobj):-
  Roads(vtype, vformat, cs, bbox, vsource, vectorobj)^
  SatelliteImage(itype, iformat, size, resolution, cs, bbox, isource, rasterobj)^
  bbox coveredby `[[33,-115],[34,-116]]`^
  size = `[[400,400]]`^
  cs = `EPSG:4326`^

Q3Quality(itype, isource, vtype, vsource, resdiff, datediff):-
  RoadQuality(vsource, vtype, vres, vdate, horiz-acc, vert-acc, vectorsin-acc-bounds, attr-comp, completeness)^
  SatelliteImageQuality(isource, itype, idate, ires, multispectral, completeness)^
  Subtract(idate, vdate, datediff)^
  Subtract(ires, vres, resdiff)^
  Pack(datediff, resdiff, date-res-diff)^
  SkylineMin(date-res-diff, skylineresultrel)^
  Unpack(skylineresultrel, smindatediff, sminresdiff)^
  smindatediff = datediff^
  sminresdiff = resdiff
Outline

- Introduction & motivation
- Quality-driven Geospatial Mediator (QGM)
  - Representing content and quality
  - Automatic source description generation
    - Content
    - Quality
      - Quality-driven query answering
      - Plan execution
- Related work
- Conclusions & future work
Automatic Source Description Generation

- Idea: Utilize well-known formats, existing standards, and information from existing sources to automatically generate description of new source.
Content Description Generator

- Match domain concepts with source layers
  - Create tokens from names of layers and titles/descriptions
  - Use Dice similarity [Rijsbergen79]
- Coverage
  - Use the coverage information from the capabilities file
  - Address different coordinate systems by using coordinate conversion operations
Experimental Results: Content Description Generator

- Tested on 1248 real-world sources
- Used QGM to find matching domain concepts
- Ground truth by manually matching domain concepts with sources
  - Using name, title, and actual data returned by sources

<table>
<thead>
<tr>
<th>Layer</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roads</td>
<td>98.02</td>
<td>95.84</td>
<td>96.92</td>
</tr>
<tr>
<td>Orthophoto</td>
<td>82.44</td>
<td>64.67</td>
<td>72.48</td>
</tr>
<tr>
<td>Raster</td>
<td>70.63</td>
<td>84.87</td>
<td>77.10</td>
</tr>
<tr>
<td>Vector</td>
<td>71.62</td>
<td>92.98</td>
<td>80.92</td>
</tr>
<tr>
<td>Rivers</td>
<td>94.55</td>
<td>98.11</td>
<td>96.30</td>
</tr>
<tr>
<td>Cart Tracks</td>
<td>100.00</td>
<td>97.83</td>
<td>98.90</td>
</tr>
<tr>
<td>Bridge/Overpasses</td>
<td>100.00</td>
<td>91.49</td>
<td>95.56</td>
</tr>
<tr>
<td>Ramp Lines</td>
<td>95.45</td>
<td>91.30</td>
<td>93.33</td>
</tr>
<tr>
<td>Topographic Maps</td>
<td>95.35</td>
<td>95.35</td>
<td>95.35</td>
</tr>
<tr>
<td>Airports</td>
<td>97.30</td>
<td>100.00</td>
<td>98.63</td>
</tr>
<tr>
<td>Schools</td>
<td>70.73</td>
<td>93.55</td>
<td>80.56</td>
</tr>
<tr>
<td>Administrative Areas</td>
<td>71.05</td>
<td>100.00</td>
<td>83.08</td>
</tr>
<tr>
<td>Counties</td>
<td>87.50</td>
<td>95.45</td>
<td>91.30</td>
</tr>
<tr>
<td>Flood Zones</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>Census Blocks</td>
<td>84.21</td>
<td>88.89</td>
<td>86.49</td>
</tr>
<tr>
<td><strong>Totals</strong></td>
<td><strong>88.21</strong></td>
<td><strong>89.66</strong></td>
<td><strong>88.93</strong></td>
</tr>
</tbody>
</table>
Outline

- Introduction & motivation
- Quality-driven Geospatial Mediator (QGM)
  - Representing content and quality
  - Automatic source description generation
    - Content
    - Quality
      - Quality-driven query answering
      - Plan execution
- Related work
- Conclusions & future work
Estimating Vector Quality

- Sample data from known and new source
- Compute value for completeness and positional accuracy attributes
  - Completeness
    - $\frac{\# \text{features}_{\text{new}} \times \text{completeness}_{\text{known}}}{\# \text{features}_{\text{known}}}$
  - Accuracy bounds
    - Use accuracy bounds of the known sources
  - Features within accuracy bounds
    - $\frac{\# \text{of features that fall within accuracy bounds}}{\# \text{features}}$
## Experimental Results: Vector Quality Completeness Estimation

<table>
<thead>
<tr>
<th>Type</th>
<th>Sample Size %</th>
<th># of Layers</th>
<th>Avg. Completeness With 100% Sampling</th>
<th>% Error with Sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Diag.</td>
</tr>
<tr>
<td>Points</td>
<td>5</td>
<td>93</td>
<td>91.76</td>
<td>17.22</td>
</tr>
<tr>
<td>Points</td>
<td>10</td>
<td>93</td>
<td>91.76</td>
<td>17.54</td>
</tr>
<tr>
<td>Points</td>
<td>20</td>
<td>93</td>
<td>91.76</td>
<td>14.27</td>
</tr>
<tr>
<td>Points</td>
<td>25</td>
<td>93</td>
<td>91.76</td>
<td>13.68</td>
</tr>
<tr>
<td>Polylines</td>
<td>5</td>
<td>297</td>
<td>38.09</td>
<td>30.18</td>
</tr>
<tr>
<td>Polylines</td>
<td>10</td>
<td>297</td>
<td>38.09</td>
<td>24.69</td>
</tr>
<tr>
<td>Polylines</td>
<td>20</td>
<td>297</td>
<td>38.09</td>
<td>20.74</td>
</tr>
<tr>
<td>Polylines</td>
<td>25</td>
<td>297</td>
<td>38.09</td>
<td>19.68</td>
</tr>
<tr>
<td>Polygons</td>
<td>5</td>
<td>12</td>
<td>68.12</td>
<td>19.54</td>
</tr>
<tr>
<td>Polygons</td>
<td>10</td>
<td>12</td>
<td>68.12</td>
<td>25.61</td>
</tr>
<tr>
<td>Polygons</td>
<td>20</td>
<td>12</td>
<td>68.12</td>
<td>24.97</td>
</tr>
<tr>
<td>Polygons</td>
<td>25</td>
<td>12</td>
<td>68.12</td>
<td>23.68</td>
</tr>
</tbody>
</table>
## Experimental Results: Vector Accuracy Estimation

<table>
<thead>
<tr>
<th>Type</th>
<th>Sample Size%</th>
<th># of Layers</th>
<th>Avg. Vec. in Bounds With 100% Sampling</th>
<th>% Error with Sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Diag.</td>
</tr>
<tr>
<td>Points</td>
<td>5</td>
<td>93</td>
<td>95.6</td>
<td>12.27</td>
</tr>
<tr>
<td>Points</td>
<td>10</td>
<td>93</td>
<td>95.6</td>
<td>9.83</td>
</tr>
<tr>
<td>Points</td>
<td>20</td>
<td>93</td>
<td>95.6</td>
<td>7.95</td>
</tr>
<tr>
<td>Points</td>
<td>25</td>
<td>93</td>
<td>95.6</td>
<td>7.71</td>
</tr>
<tr>
<td>Polylines</td>
<td>5</td>
<td>297</td>
<td>80.28</td>
<td>9.8</td>
</tr>
<tr>
<td>Polylines</td>
<td>10</td>
<td>297</td>
<td>80.28</td>
<td>8.68</td>
</tr>
<tr>
<td>Polylines</td>
<td>20</td>
<td>297</td>
<td>80.28</td>
<td>8.95</td>
</tr>
<tr>
<td>Polylines</td>
<td>25</td>
<td>297</td>
<td>80.28</td>
<td>8.67</td>
</tr>
<tr>
<td>Polygons</td>
<td>5</td>
<td>12</td>
<td>82.19</td>
<td>10.63</td>
</tr>
<tr>
<td>Polygons</td>
<td>10</td>
<td>12</td>
<td>82.19</td>
<td>9.81</td>
</tr>
<tr>
<td>Polygons</td>
<td>20</td>
<td>12</td>
<td>82.19</td>
<td>10.12</td>
</tr>
<tr>
<td>Polygons</td>
<td>25</td>
<td>12</td>
<td>82.19</td>
<td>9.97</td>
</tr>
</tbody>
</table>
Raster Quality Estimation: Overstated Coverage

- Water no coverage
- Land no coverage
- B/W Satellite Image Only
- Topo maps & B/W Satellite Image
- Multi-spectral Satellite Image
Estimating Raster Coverage & Completeness

- Address the problem of sources overstating coverage
- Sample data from a source
- Use the sampling results and Voronoi diagram
- Estimate accurate coverage and completeness
Experimental Results: Raster Coverage Estimation

- Automatic estimation of Raster Quality
  - 60 queries with resolutions 1, 5, 10, 50 m/p
  - Compare reported coverage with estimated coverage by sampling
  - Estimated coverage
    - loses some images (lower recall)
    - returns fewer empty images (higher precision)

<table>
<thead>
<tr>
<th>Resolution 1 meter/pixel</th>
<th>Reported Coverage</th>
<th>Estimated Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
</tr>
<tr>
<td>1</td>
<td>72.15</td>
<td>100.00</td>
</tr>
<tr>
<td>5</td>
<td>82.43</td>
<td>100.00</td>
</tr>
<tr>
<td>10</td>
<td>81.82</td>
<td>100.00</td>
</tr>
<tr>
<td>15</td>
<td>81.58</td>
<td>100.00</td>
</tr>
<tr>
<td>Total</td>
<td>79.23</td>
<td>100.00</td>
</tr>
</tbody>
</table>
Outline

- Introduction & motivation
- Quality-driven Geospatial Mediator (QGM)
  - Representing content and quality
  - Automatic source description generation
    - Content
    - Quality
  - Quality-driven query answering
  - Plan execution
- Related work
- Conclusions & future work
QGM’s Query Answering

Source & Operator Descriptions
Domain Knowledge

User Query

Inverse Rules
[Duschka1997]
Generic Optimizations
[Thakkar2003, Kambhampati2003]

Initial Datalog Program
Plan graph

Evaluate Quality Criteria

Plan graph, Quality Results

Pruned Graph

Theseus Plan

Execute Plan

Datalog to Theseus

Answer
Sample Query 3

Find satellite image and road vector data covering `[[33,-116],[34,-117]]` such that both the resolution and date differences are minimized.

- $Q3(imageobj, vectorobj, resdiff, datediff)$:
  - $Q3Data(itype, isource, vtype, vsource, imageobj, vectorobj)\wedge Q3Quality(itype, isource, vtype, vsource, resdiff, datediff)$

  $Q3Data(itype, isource, vtype, vsource, imageobj, vectorobj)$:
  - $Roads(vtype, vformat, cs, bbox, vsource, vectorobj)\wedge SatelliteImage(itype, iformat, size, resolution, cs, bbox, isource, rasterobj)\wedge bbox\text{ covered by } `[[33,-115],[34,-116]]`\wedge size = `[400,400]`\wedge cs = `EPSG:4326`$

  $Q3Quality(itype, isource, vtype, vsource, resdiff, datediff)$:
  - $RoadQuality(vsource, vtype, vres, vdate, horiz-acc, vert-acc, vectorsin-acc-bounds, attr-comp, completeness)\wedge SatelliteImageQuality(isource, itype, idate, ires, multispectral, completeness)\wedge Subtract(idate, vdate, datediff)\wedge Subtract(ires, vres, resdiff)\wedge Pack(datediff, resdiff, date-res-diff)\wedge SkylineMin(date-res-diff, skylineresultrel)\wedge Unpack(skylineresultrel, smindatediff, sminresdiff)\wedge smindatediff = datediff\wedge sminresdiff = resdiff$
Inverse Rules [Duschka 1997]

- Determine how to query domain relations
- Invert the source descriptions
- In the example query
  - Definition of Roads & SatelliteImage as views over sources
  - Definition of RoadQuality and SatelliteImageQuality as views over source quality

NavteqRoads(bbox, vectorobj):-
  Roads(type, format, cs, bbox, source, vectorobj)^
  bbox coveredby `[[33,-116],[34,-117]]`
  source = `Navteq`
  type = `Roads`
  format = `Shapefile`
  cs = `EPSG:4326`

Roads(`Roads`, `Shapefile`, `EPSG:4326`, bbox, `Navteq`, vectorobj):-
  NavteqRoads(bbox, vectorobj)^
  bbox coveredby `[[33,-116],[34,-117]]`
Datalog Program Generation

- Identify Relevant Rules
  - Extension: Check geospatial constraints
  - Find sources that
    - Appear in definition of relevant domain concepts
    - Do not have conflicting coverage constraints
  - In the example query
    - Find sources that appear in definition of Roads or SatelliteImage
    - Have coverage intersecting with Query’s bounding box

```
Roads(`Roads', `Shapefile', `EPSG:4326', bbox, `Navteq', vectorobj):-
  NavteqRoads(bbox, vectorobj)^bbox coveredby
  `[[33,-116],[34,-117]]'

Roads(`Roads', `Shapefile', `EPSG:4326', bbox, `Navteq', vectorobj):-
  TigerRoads(bbox, vectorobj)^bbox coveredby
  `[[33,-116],[34,-117]]'

Parks(`Roads', `Shapefile', `EPSG:4326', bbox, `Navteq', vectorobj):-
  NGAParks(bbox, vectorobj)^bbox coveredby
  `[[33,-116],[34,-117]]'
```
Generated Plan

- Two branches
  - Content
    - Has requests to data sources
    - Select operations to apply constraints
  - Quality
    - Has requests to obtain facts about quality of data for sources that appear in the content plan
    - May have requests to mathematical, aggregate, or skyline operations
  - In our example query
    - Assume two relevant vector sources
    - Assume one image source
Generated Plan

- Two branches
  - Content
    - Has requests to data sources
    - Select operations to apply constraints
  - Quality
    - Has requests to obtain facts about quality of data for sources that appear in the content plan
    - May have requests to mathematical, aggregate, or skyline operations
  - In our example query
    - Assume two relevant vector sources
    - Assume one image source
Generated Plan

- Two branches
  - Content
    - Has requests to data sources
    - Select operations to apply constraints
  - Quality
    - Has requests to obtain facts about quality of data for sources that appear in the content plan
    - May have requests to mathematical, aggregate, or skyline operations
  - In our example query
    - Assume two relevant vector sources
    - Assume one image source
QGM’s Query Answering

- **Source & Operator Descriptions**
- **Domain Knowledge**

1. **User Query**
2. **Inverse Rules** [Duschka1997]
3. **Initial Datalog Program**
4. **Generic Optimizations** [Thakkar2003, Kambhampati2003]
5. **Plan graph**
6. **Datalog to Theseus**
7. **Theseus Plan**
8. **Pruned Graph**
9. **Execute Plan**
10. **Execute Quality Criteria**
11. **Prune-based on Quality Results**

Answer
Executing Quality Criteria

- Obtain Quality facts
- Apply necessary relational, mathematical, or aggregate operations
- Apply constraints and/or skyline operations
- Resulting tuples include source name and type for each type of data and any other attributes requested in quality query
Prune based on Quality Results

- Remove all sources that did not satisfy quality criteria
  - If a source (S1) has completeness 20% and quality criteria is completeness > 50%
  - Remove source (S1) from the content subtree
- Check join constraints in the graph connected to quality subtree
  - Remove branches that do not produce tuples

<table>
<thead>
<tr>
<th>Combination</th>
<th>Resdiff (m/p)</th>
<th>Datediff (days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navteq &amp; TS</td>
<td>5</td>
<td>365</td>
</tr>
<tr>
<td>Tiger &amp; TS</td>
<td>12</td>
<td>1825</td>
</tr>
</tbody>
</table>

Quality Statistics

- TerraServer
- Union
- X
- Tiger
- Navteq
Execute Final Plan

- QGM converts the plan to Theseus
  - Streaming, dataflow-style execution
- QGM also generates plans to access
  - Shapefiles
  - ArcIMS services
  - Web Map Servers
  - Web services
  - Databases
Experimental Evaluation

- Setup
  - Dual Xeon processor, 3 GB memory
    - Actual use: half processor, 1GB memory
  - Data sources
    - Real-world shapefiles, ArcIMS services, and Web Map Services

- Method
  - Compare with Prometheus
    - Data integration system that supports geospatial data without any quality information
  - Compare
    - Quality
    - Response time
Query Answering: Quality of Answers

- Query answering
  - Quality
    - One standard deviation better in completeness for most queries
    - Half standard deviation better for accuracy

<table>
<thead>
<tr>
<th>Type</th>
<th>QGM</th>
<th>Average</th>
<th>Std. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>% Comp.</td>
<td>% Acc.</td>
<td>% Comp.</td>
</tr>
<tr>
<td>Constraint</td>
<td>59.81</td>
<td>87.61</td>
<td>47.71</td>
</tr>
<tr>
<td>Aggregate</td>
<td>68.19</td>
<td>89.97</td>
<td>47.71</td>
</tr>
<tr>
<td>Skyline</td>
<td>64.03</td>
<td>87.90</td>
<td>47.71</td>
</tr>
</tbody>
</table>
## Query Answering: Response Time

<table>
<thead>
<tr>
<th>Query</th>
<th># of Sources</th>
<th>Prometheus</th>
<th></th>
<th>QGM</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Time in Seconds</td>
<td># results</td>
<td></td>
<td>Time in Seconds</td>
<td># results</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Constraint</td>
<td>0-5</td>
<td>32</td>
<td>98</td>
<td>126</td>
<td>256</td>
<td>3.7</td>
<td>32</td>
<td>109</td>
</tr>
<tr>
<td>Constraint</td>
<td>5-10</td>
<td>33</td>
<td>119</td>
<td>279</td>
<td>431</td>
<td>7.9</td>
<td>33</td>
<td>116</td>
</tr>
<tr>
<td>Constraint</td>
<td>20-30</td>
<td>34</td>
<td>168</td>
<td>1985</td>
<td>2187</td>
<td>24.3</td>
<td>34</td>
<td>159</td>
</tr>
<tr>
<td>Aggregate</td>
<td>0-5</td>
<td>32</td>
<td>98</td>
<td>126</td>
<td>256</td>
<td>3.7</td>
<td>32</td>
<td>113</td>
</tr>
<tr>
<td>Aggregate</td>
<td>5-10</td>
<td>33</td>
<td>119</td>
<td>279</td>
<td>431</td>
<td>7.9</td>
<td>33</td>
<td>116</td>
</tr>
<tr>
<td>Aggregate</td>
<td>10-20</td>
<td>32</td>
<td>131</td>
<td>872</td>
<td>1035</td>
<td>16.1</td>
<td>32</td>
<td>137</td>
</tr>
<tr>
<td>Aggregate</td>
<td>20-30</td>
<td>34</td>
<td>168</td>
<td>1985</td>
<td>2187</td>
<td>24.3</td>
<td>34</td>
<td>162</td>
</tr>
<tr>
<td>Skyline</td>
<td>0-5</td>
<td>32</td>
<td>98</td>
<td>126</td>
<td>256</td>
<td>3.7</td>
<td>32</td>
<td>140</td>
</tr>
<tr>
<td>Skyline</td>
<td>5-10</td>
<td>33</td>
<td>119</td>
<td>279</td>
<td>431</td>
<td>7.9</td>
<td>33</td>
<td>192</td>
</tr>
<tr>
<td>Skyline</td>
<td>10-20</td>
<td>32</td>
<td>131</td>
<td>872</td>
<td>1035</td>
<td>16.1</td>
<td>32</td>
<td>297</td>
</tr>
<tr>
<td>Skyline</td>
<td>20-30</td>
<td>34</td>
<td>168</td>
<td>1985</td>
<td>2187</td>
<td>24.3</td>
<td>34</td>
<td>421</td>
</tr>
</tbody>
</table>
Outline

- Introduction & motivation
- Quality-driven Geospatial Mediator (QGM)
  - Representing content and quality
  - Automatic source description generation
    - Content
    - Quality
  - Quality-driven query answering
  - Plan execution

- Related work

- Conclusions & future work
Related Work (1/2)

- Geospatial Data Integration
  - Hermes [Adali95], MIX [Gupta99], GeonGrid [Manipura03], VirGIS [Boucelma04]
    - Focus access methods and formats
    - GeonGrid also has some quality and ontology components
  - ODGIS [Fonesca 02], GSA [Arpinar 06], SWING [Klien06]
    - Creation ontology for geospatial data and matching data layers

- Quality-driven data integration
  - Biological data [Eckman06, Mihila05]
    - Focus is on completeness
  - General-purpose [Neumann01, Bleiholder2006, Scannapieco05]
    - Assign one quality score based on user-supplied weights

- QGM
  - A Geospatial mediation framework that supports quality
  - Automatic generation of descriptions
  - More expressive quality criteria specification
Related Work (2/2)

- OpenGIS & mapping systems
  - Standards, Web-based mapping, Desktop GIS
- Source modeling [Carman06]
  - Learn source descriptions by sampling data from a source
- Geospatial data quality
  - Conflation Operation [Saalfield 1993, Chen 2003]
  - Representation [Goodchild02-03]
  - Visualization [Worboys01]
- QGM
  - QGM utilizes the existing standards and well-known formats
  - QGM provides output using OpenGIS standards
  - Quality specification in QGM is flexible and can utilize existing specifications
Conclusion: Contributions

- A declarative specification of both the content and the quality of geospatial sources
- Algorithms to automatically generate source descriptions and estimate the quality of data provided by geospatial data sources
- A quality-driven query answering algorithm
- An approach to map the generated integration plans and source requests to a program that is efficiently executed by a streaming, dataflow-style execution engine.
Conclusion: Broader Implications

- Geospatial data integration framework that supports quality-driven integration
- QGM’s query answering technique can be easily applied in other domains
- Quality estimation techniques can be utilized for automated quality estimation in many domains
Conclusion: Future Work

- **Source discovery**
  - Use terms from gazetteer to create keywords for searching geospatial data
  - Utilize the Web catalog service (OpenGIS standards)

- **Automatic source description generation**
  - Utilize token weights and transformation weights

- **Quality estimation**
  - Density-based sampling
  - Adaptive raster sampling
    - Sample at the edges of the cells

- **Query answering**
  - Utilize spatial database to prune based on coverage
  - Improve response time by parallel processing on multiple machines
Questions