Quality-Driven Geospatial Data Integration

Snehal Thakkar, Craig A. Knoblock, Jose Luis Ambite
University of Southern California
Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
thakkar,knoblock,ambite@isi.edu

ABSTRACT

Accurate and efficient integration of geospatial data is an important problem with applications in areas such as emergency response and urban planning. Some of the key challenges in supporting large-scale geospatial data integration are automatically computing the quality of the data provided by a large number of geospatial sources and dynamically providing high quality answers to the user queries based on a quality criteria supplied by the user. We describe a framework called the Quality-driven Geospatial Mediator (QGM) that supports efficient and accurate integration of geospatial data from a large number of sources. The key contributions of our framework are: (1) the ability to automatically estimate the quality of data provided by a source by using the information from another source of known quality, (2) representing the quality of data provided by the sources in a declarative data integration framework, and (3) a query answering technique that exploits the quality information to provide high quality geospatial data in response to user queries. Our experimental evaluation using over 1200 real-world sources shows that QGM can accurately estimate the quality of geospatial sources. Moreover, QGM provides better quality data in response to the user queries compared to the traditional data integration systems and does so with lower response time.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Algorithms, Management, Design, Performance

Keywords

Geospatial Data Quality, Quality-driven Query Answering

1. INTRODUCTION

The proliferation of geospatial data on the Internet has resulted in the availability of a large number of geospatial data sources with different types of data of varying quality. While it is hard to estimate the total number of available geospatial data sources in the web, a quick check on collections of available data reveals the following statistics: (1) the Geospatial Information Database (GiDB) [1] project provides a list of over 1400 web map servers that provide over 200,000 map data layers and (2) for vector data, a Google search for keywords 'download shapefile' produces over 344,000 result pages. These simple statistics clearly show that there are many available geospatial data sources on the web. However, each source is different from another in respect to the types of data that it provides, coverage of the data, and the quality of the data. Moreover, a lot of geospatial data sources do not provide information about the quality of their data or bury the quality information in a text document.

While for most users it is easy to find geospatial sources and possibly even identify the type of data they provide, it is difficult for them to evaluate the quality of data provided by the available sources. For example, there are over 317 road network data sources covering the area of the Los Angeles county in the GiDB portal and Mapdex[2] repositories. A user searching for high quality road network data would have to go through all the sources and assess the quality of data from each source to find high quality road network data. The task of manually evaluating the quality of sources is very tedious. Moreover, the geospatial sources on the Internet have huge differences in the quality of data. At one end of the spectrum is the high quality data provided by various universities or commercial companies that usually have limited coverage. At the other end of the spectrum, public data sources, like the U.S. Census Bureau's Tigerlines, cover much larger areas, but have lower quality.

We have developed a flexible geospatial data integration framework called Quality-driven Geospatial Mediator (QGM) that can automatically assess the quality of a large number of data sources, represent the quality information declaratively, and exploit the quality information to provide high quality data in response to a user query. QGM allows data providers to specify available data sources and their coverages. QGM has the ability to automatically assess the quality of data. Given the information from the source provider, the assessed quality information, a user query, and a quality metric, QGM can then provide the best quality results for the given query.

The rest of the paper is organized as follows. Section 2 discusses a motivating example that we use throughout the

ACMGIS'07 November 7-9, 2007, Seattle, WA
Copyright 2007 ACM 978-1-59593-914-2/07/11 ...$5.00.

[1] http://dmap.nrlssc.navy.mil/
[2] http://www.mapdex.org

Source Name	Type	Coverage	QualityInfo
NavTeqRoads	Road Vector	[[33.5,-117],[34,-118]]	Yes
TigerRoads	Road Vector	[[33.5,-117],[34,-117.5]]	No
CasilRoads	Road Vector	[[33,-117],[34,-118]]	No
NavteqHospitals	Hospital Points	[[33.5,-117],[34,-118]]	Yes
TerraServerImages	Satellite Image	[[33,-116],[34,-118]]	Yes
GoogleMapsImages	Satellite Image	[[33,-116],[34,-118]]	Yes
TerraServerTopoMaps	Topo Maps	[[33,-116],[34,-118]]	Yes

Table 1: Available Sources

paper to explain our approach. Section 3 describes QGM's method to estimate the quality of geospatial sources. Section 4 describes QGM's approach to representing geospatial sources and the quality of data they provide. Section 4 describes the representation of geospatial data sources. Section 5 describes QGM's query answering technique. Section 6 describes the results of our experimental evaluation using real-world geospatial data sources. Section 7 discusses closely related work. Section 8 concludes the paper by discussing our contributions and plans for future work.

2. MOTIVATING EXAMPLE

In this section we describe an example scenario using real-world geospatial data sources to illustrate our approach. In the example scenario QGM has access to the data sources shown in Table 1. Different sources provide different types of data and have different coverage. While QGM can handle sources with different coordinate systems, in the motivating example we assume that all sources provide information in one coordinate system. We also assume QGM has information about the quality of data provided by some of the data sources, but not all sources.

In general, QGM allows a domain expert to specify different quality attributes for different sources and domain concepts. For example, we can utilize the quality attributes from FGDC geospatial metadata guidelines. In our motivating example, we use the most common measures representing positional accuracy of data provided by sources and the completeness of data. For vector data sources that provide information about a feature,[3] QGM has information about the completeness of a source and the positional accuracy of features provided by a source. The completeness of a vector data source refers to the percentage of real-world features that the source provides. For example, if there are 100 hospitals in an area and a source provides 25 hospital points, then the source is 25% complete.

We use two variables to represent the positional accuracy of a data source: *accuracy bound* and *features within the accuracy bounds*. The *accuracy bound* attribute represents an area around the actual location of the feature, while the *features within the accuracy bounds* attribute provides the percentage of features provided by a source that fall within the area around the actual location of the feature. The two attributes relate to the following quality information that is often present in the metadata provided by sources: The location of the provided features is accurate within 'n units' for 'k %' features. The 'n units' refer to the accuracy bounds for the dataset, while the 'k%' refers to the *features within accuracy bounds* attribute.

Source	% Completeness	Positional Accuracy	
		Accuracy Bounds (meters)	% Features Within Bounds
NavteqRoads	85	3.6	91
NavteqHospitals	89	3.6	93

Table 2: Quality of Vector Data Sources

Source	Date Collected	Orig. Resolution
TerraServerImages	1/1/2001	0.3 m/p
GoogleMaps	1/1/2004	0.5 m/p
TerraServerTopoMaps	1/5/1999	2 m/p

Table 3: Quality of Raster Data Sources

In our motivating example, we assume that we have quality information for the *NavteqRoads* and *NavteqHospitals* data sources. Table 2 shows the quality information. For image sources, we use two measures: (1) the date the image was taken and (2) the resolution at which the image was collected. Table 3 shows the quality data for the image sources.

Given this information, QGM's task is to answer the following user query: *Find the most accurate and complete road vector data set and a satellite image collected at resolution better than 1 meter/pixel for the area covered by the bounding box '[[33,-116][34,-118]]'.*

3. ASSESSING QUALITY OF SOURCES

In this section, we describe QGM's approach to assessing the quality of data provided by sources. QGM only needs to assess the quality of data for sources that do not provide the quality information. In our running example QGM assesses the quality of data from the *TigerRoads* and the *CasilRoads* data sources. The estimation of quality may lead to better quality answers to the user query or improved coverage.

The intuition behind the automatic quality assessment is to utilize the information from sources with known quality of data to estimate the quality of data provided by sources with unknown quality of data. When QGM encounters a source that does not provide information about the quality of data, QGM estimates the quality of data for each type of data provided by a source separately. First, for each source with unknown quality, QGM identifies a source with known quality that provides the same type of data and has some overlapping coverage with the source with unknown quality. If no such sources exist, QGM adds the source with unknown quality to list of available sources with unknown quality of data. After QGM successfully assesses the quality of data provided by any new source, it checks to see if the expanded coverage due to the new source allows QGM to evaluate any new sources. For example, consider that QGM needs to assess the quality of road network data covering the city San Diego provided by some data source. However, the only source with known quality road network data only covers the city of Los Angeles. As there is no overlap, QGM adds the road network source covering the city of San Diego to list of sources with unknown quality of data. Next, QGM needs to assess the quality of road network data covering the entire Southern California region provided by some data source. As the Southern California region overlaps with the city of Los Angeles, QGM can assess the quality of road network data for the source covering the Southern California region. Once QGM finishes assessing the quality of data provided by the source, it checks the list of sources with unknown quality and discovers that it can now assess the quality of

[3] We use the term feature to refer to a geographic entity that can be represented using either a point, polyline, or polygon.

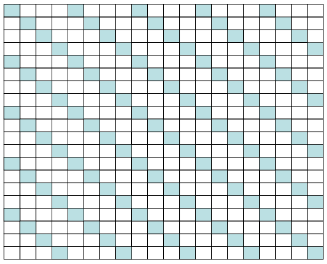

Figure 1: Sampling pattern Utilized by QGM

the source that provides the road network data for the city of San Diego.

If QGM finds a source with known quality and overlapping coverage, it continues the quality assessment process using the source with the known quality as a reference source. The second step of the assessment process is to sample data from the source with unknown quality data and the reference source. As geospatial data sources may contain a lot of data and may not allow querying of all of its data, it is important to sample a small amount of data instead of retrieving all the data provided by a source. In order to obtain a representative sample, QGM divides the overlapping area between the source with unknown quality and the reference source into a grid with equal size cells.

From the generated grid, QGM samples data from several cells located along the diagonals in the grid similar to the pattern shown in Figure 1. The dark cells in the figure indicate the sampled area. As geospatial data may not be distributed uniformly, it is important to select cells that represent the distribution of the data provided by a source. The rationale behind selecting this pattern was that by selecting cells distributed throughout the coverage area, we would get a good representative set of features. The assumption in the quality estimation process is that the quality of data provided by a source is uniform across its coverage.

The third step in the quality assessment process is to analyze the sampled features from both sources to compute values for the quality attributes for the source with unknown quality. QGM has to assess values for one attribute corresponding to the completeness of the data and two attributes corresponding to the positional accuracy of the data (accuracy bound and features within accuracy bounds).

3.1 Estimating Completeness

QGM estimates the completeness of a new source given the sampled data from the reference source and the completeness of the reference source. The completeness of a geospatial source refers to the percentage of total existing features that the source provides. As QGM does not know the total number of existing features in the area, it estimates the existing features using the reference source. For example, if the reference source is 50% complete and it provides two features in an area, QGM estimates that there are four features in the area. QGM estimates the completeness for the new source by comparing the number of features provided by the new source with the estimated number of features in the area using the following formula:

$$C_{new} = \frac{\#\text{ of features}_{new}}{\#\text{ of features}_{reference}} * C_{reference}$$

The formula takes into account the fact that the reference source may not be complete. First, the formula computes the completeness of the new source compared to the reference source and multiplies that number with the completeness of the reference source to obtain the completeness of the new source.

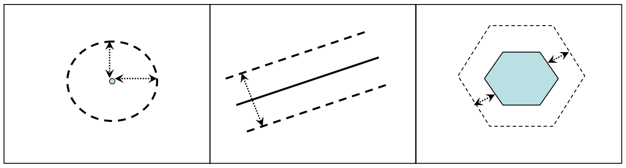

Figure 2: Examples of buffers for points, lines, and polygons

In the case of polylines, the number of features is not a good indicator as different sources may use different granularity to define features. For example, one source may consider a freeway as a set of separate features each representing one segment, while the other may consider entire freeway as one feature. Therefore, if a source provides polylines, QGM utilizes a different formula that takes into account the length of the polylines, instead of the number of polylines.

$$C_{new} = \frac{\sum(\text{length of all polylines}_{new})}{\sum(\text{length of all polylines}_{reference})} * C_{reference}$$

3.2 Estimating Positional Accuracy

In this section, we describe QGM's approach to estimating the accuracy of the data provided by the new source. The accuracy of the data source is measured using two attributes: (1) *accuracy bounds* and (2) *features within accuracy bounds*.

As QGM does not have access to the actual location of a feature, it utilizes the features from the reference source to approximate the actual location of the feature and computes the values for the accuracy of the data provided by the new source. First, QGM retrieves the values for the accuracy attributes for the reference source. QGM utilizes the values of the accuracy attributes to generate a buffer around all sampled features from the reference set. Figure 2 shows examples of buffers for points, polylines, and polygons.

Next, QGM determines the percentage of features from the sample data retrieved from the source with unknown quality that fall within the buffer. If the source provides point data, QGM counts the number of points that are within the buffer using the following formula.

$$Acc_{new} = \frac{\#\text{ of features within the buffer}}{\#\text{ of features}}$$

As QGM has computed the percentage of features from the source with unknown quality that fall within the accuracy bounds used by the reference set, it utilizes the value for the accuracy bounds from the reference set as the value for the accuracy bounds attribute. QGM utilize the Acc_{new} value as the *percentage of features within accuracy bounds*.

If the source provides polylines, QGM computes the total length of all parts of the polylines that are within the buffer using the following formula.

$$Acc_{new} = \frac{\sum(\text{length of all polylines within buffer})}{\sum(\text{length of all polylines})}$$

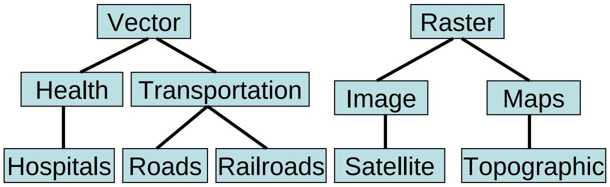

Figure 3: Domain Concepts Hierarchy

Vector(type, source, bbox, format, cs, vectorobj)
Transportation(type, source, bbox, format, cs, vectorobj)
Raster(type, source, bbox, format, size, cs, rasterobj)
Image(type, source, bbox, format, size, cs, rasterobj)

Table 4: Example Domain Relations

If the source provides polygons, QGM computes the total area of all parts of the polygons that are within the buffer.

$$Acc_{new} = \frac{\sum(\text{area of polygons within buffer})}{\sum(\text{area of all polygons})}$$

QGM utilizes the computed values as the values for the accuracy of the new source.

4. REPRESENTING GEOSPATIAL DATA

In this section, we describe our approach to describing the content and the quality of geospatial sources in QGM. We begin by reviewing the previous work on representing domain concepts and available sources. Next, we describe our approach to representing the quality of data provided by the sources.

4.1 Previous Work: Representing Domain Concepts and Sources

QGM builds on the research in data integration systems [10, 15] that represent various entities in the domain as relations with attributes. The set of domain relations form the mediated schema of the integration framework. In addition, we organize the domain concepts in a hierarchy by manually merging the domain concepts from the FGDC list of geographic concepts,[4] the hierarchy of geospatial data types from the National Atlas,[5] and the National Geospatial Agency (NGA) spatial data types[6]. Figure 3 shows the partial domain concepts hierarchy focusing on the domain concepts relevant to our running example. Table 4 shows the attributes associated with different domain concepts.

Similar to the domain concepts, QGM also represents the sources as relations with a set of attributes. For example, the *NavteqRoads* data source, which accepts a bounding box of an area and provides road vector data object containing road segments in the area, is represented as a relation with two attributes: *bbox* and *vectorobj*. The image sources in our example accept the size of the image and the bounding box of an area as inputs and provide an image object that contains the image of the area as output.

QGM utilizes the Local-As-View approach [14, 15] to describe the relationship between the available sources and domain concepts. In the Local-As-View approach, each source is defined as a view over a one or more domain relations. Figure 4 shows the source descriptions for the *NavteqRoads* and *TerraServerImages* data sources. The source descriptions specify the type of data a source provides and the coverage limitation of the data source. For example, the rule *S1* in Figure 4 states that the *NavteqRoads* data source provides road vector data covering the bounding box '[[33.5,-117],[34,-118]]'.

```
S1:NavteqRoads(bbox, vectorobj):-
    Roads(type, source, bbox, format, cs, vectorobj)ˆ
    format = ‘GML’ˆ  cs = ‘EPSG:4326’ˆ
    type = ‘Roads’ˆ source = ‘Navteq’
    vectorobj coveredby ‘[[33.5,-117],[34,-118]]’
S2:TerraServerImages(bbox, size, rasterobj):-
     SatelliteImage(type, source, bbox, format, size,
        cs, rasterobj)ˆ
     format = ‘JPG’ˆ cs = ‘EPSG:4326’ˆ
     rasterobj coveredby ‘[[33,-116],[34,-118]]’ˆ
     source = ‘TerraServer’ˆ type = ‘SatelliteImage’
```

Figure 4: Example Source Descriptions

4.2 Describing the Quality of Geospatial Data

In QGM we represent the quality of geospatial data in four steps. First, we create quality relations for each domain relation to allow users to specify restrictions on the quality of data. For example, corresponding to the *Vector* domain relation, we have a *VectorQuality* domain relation that describes the quality of a vector data object with the following attributes: (1) vector type, (2) source, (3) completeness, (4) accuracy bounds, and (5) vectors within accuracy bounds. Simiarly, we also have a quality relation for the *Raster* domain relation with the following attributes: (1) raster type, (2) source, (3) date collected, and (4) resolution collected. We create similar quality relations for all domain relations.

Second, we also define a quality relation for each source. The quality relations for the sources have all attributes from the corresponding domain quality relation, except the attribute for the data object. For example, the source quality relation for the *NavteqRoads* source has the following attributes: (1) completeness, (2) accuracy bounds, and (3) vectors within accuracy bounds. The image sources have the following two attributes for the quality: (1) date collected and (2) resolution collected.

Third, we provide QGM descriptions of the source quality relations as views over the quality relations for the domain concepts. These descriptions are similar to the source descriptions that relate the sources and the domain concepts. Below is an example description for a source quality relation:

```
NavteqRoadsQuality(completeness, accbounds,
      featureswithinaccbounds):-
 RoadsQuality(type, source, completeness,
      accbounds, featureswithinaccbounds)ˆ
 type = ‘Roads’ ˆ source = ‘Navteq’
```

Fourth, we also provide QGM a set of facts that constitute the tuples in each source quality relation. For the sources with unknown quality QGM generates the facts about the quality using techniques described in Section 3. Table 5 shows list of facts about the quality of data for our motivating example. The quality facts for the *TigerRoadsQuality*

[4] http://clearinghouse1.fgdc.gov/servlet/FGDCWizard
[5] http://www.nationalatlas.gov
[6] http://earth-info.nga.mil/publications/specs/

Relation	Quality
NavteqRoadsQuality(85, 3.6, 91)	Provided
NavteqHospitalsQuality(89, 3.6, 93)	Provided
TigerRoadsQuality(76, 3.6, 68)	Estimated
CasilRoadsQuality(93, 3.6, 88)	Estimated
TerraserverImageQuality('1/1/2001', 0.3)	Provided
GoogleMapsImageQuality('1/1/2004', 0.5)	Provided
TerraserverTopoMapsQuality('1/1/1999', 2)	Provided

Table 5: Quality Facts for our Running Example

and the *CasilRoadsQuality* are generated using the automatic quality estimation technique described in Section 3.

The domain hierarchy rules, source descriptions, source quality descriptions, and facts about the quality relations together make up QGM's domain model.

5. EXPLOITING QUALITY IN QUERY ANSWERING

The assessment and representation of quality of data provided by the available sources allows QGM to exploit the quality of data to provide more accurate answers for user queries. In this section, we describe QGM's process of providing high quality data in response to user queries. We begin by reviewing the previous work on using the view integration approach to answering user queries [6]. Then, we describe QGM's query answering algorithm that exploits the quality information.

5.1 Previous Work: View Integration

Data integration systems provide a uniform interface to a large number of sources. Users can specify their queries using logic rules containing the domain relations and order constraints. For example, a query to obtain the road vector data and a satellite image covering the area specified by the bounding box '[[33,-116][34,-118]]' can be specified using the query shown below:

```
Q1Data(vectorobj, imageobj, vtype, vsource, rtype,
   rsource):-
 Roads(vtype, vsource, bbox, format, cs, vectorobj)^
 SatelliteImage(rtype, rsource, bbox, format, size,
    cs, rasterobj)^
 vformat = 'GML'^ iformat = 'JPG'^
 cs = 'EPSG:4326'^ bbox = '[[33,-116],[34,-118]]'^
 vectorobj coveredby bbox^
 imageobj coveredby bbox
```

Given the user query, a data integration system utilizes a query reformulation algorithm to generate a datalog program to answer the query using the source descriptions and the domain rules. Second, the system optimizes the generated datalog program to reduce the execution time. Third, the system executes the generated plan and returns the answers to the user.

One of the well-known algorithms to generate a datalog program to answer a given user query is called the Inverse Rules algorithm [6]. The intuition behind the Inverse Rules algorithm is to obtain executable rules from the source descriptions by inverting the descriptions. For every source description, $S(X) : -P_1(X_1), ..., P_n(X_n)$, where X and X_i refer to set of attributes in the corresponding view or relation, the Inverse Rules algorithm generates n inverse rules, for $i = 1, .., n$, $P_i(X_i) : -S(X)$, where if $X_i \notin X$, X_i is replaced by a function symbol. The key advantages of the Inverse Rules algorithm are the ability to handle recursive user queries, functional dependencies, and access pattern limitations. The inverted rules, any domain rules, and the user query together form a datalog program that can answer the user query.

The datalog program generated by the Inverse Rules algorithm often contains many source requests that do not contribute to the answers of the query. Kambhampati et al. [13] describe an optimization technique that checks the constraints in the source descriptions and the constraint in the user query to remove all sources that have constraints that conflict with the constraints in the user query. For example consider a data source that provides road vector data for the bounding box '[[25,-74],[27,-76]]'. As the coverage of data source does not intersect with the area specified in the user query, the optimization technique would remove this source request from the generated datalog program. Once the datalog program is optimized, the data integration system utilizes a datalog interpreter to evaluate the generated program and obtain answers to the user queries.

5.2 QGM's Extensions to Exploit Quality

QGM's query answering algorithm builds on the query answering techniques used in the traditional data integration systems. In particular, QGM allows user to specify a quality criteria and utilizes the quality information for the sources to identify sources in the generated plans that do not satisfy the given criteria. QGM removes the requests to sources that do not satisfy the quality criteria. The quality criteria is specified using a logic rule similar to the query for the content. The logic rule describing the quality restrictions contains one relation for each type of data retrieved in the query, any applicable order constraints, and if needed one or more aggregate or *Skyline* operations. Skyline queries [3] are used to find all points in a multi-dimensional space which are not dominated by any other point. The *Skyline* queries are important in describing the quality restrictions as the users often need to optimize multiple attributes of quality. In our running example, users have a trade off between the completeness and the positional accuracy of vector datasets and the ability to perform a *Skyline* query frees the user from having to define a function that combines both the completeness and the positional accuracy.

For our running example the quality restrictions in the query, *find the most accurate and complete road vector data set and a satellite image collected at resolution better than 1 meter/pixel for the area covered by the bounding box '[[33,-116][34,-118]]'*, are specified using the rule below:

```
Q1Quality(vtype, vsource, rtype, rsource,
   resolution, completeness, featuresinaccbounds):-
 RoadsQuality(vtype, vsource, completeness,
   accbounds, featuresinaccbounds)^
 SatelliteImageQuality(rtype, rsource, date,
   resolution)^
 resolution < 1 ^
 SkylineMax(completeness, featuresinaccbounds)
```

In our example, the quality query contains the *RoadsQuality* and the *SatelliteImageQuality* predicates representing the quality of the road vector data and the satellite image retrieved in the query. The order constraint on the *original resolution* attribute specifies that all satellite imagery in the answers to the user query must be collected at a resolution better than 1 meter/pixel. We use the *SkylineMax* operator to ask QGM to maximize the values of the *completeness* and the *features within accuracy bounds* attributes. The user also specifies a join between the content (Q1Data) and the quality (Q1Quality) portion of the queries as shown below:

```
Q1(vectorobj,imageobj, resolution,
      completeness, featuresinaccbounds):-
   Q1Quality(vtype, vsource, rtype, rsource,
     resolution, completeness, featuresinaccbounds)^
   Q1Data(vectorobj, imageobj, vtype, vsource,
     rtype, rsource)
```

Here we have specified a join between the quality and the content rules using the data source name and the type of data the source provides for each type of data object retrieved in the query. This is due to the fact that we have modeled quality at the level of type of data provided by a source. If we need to model quality at a finer granularity, we can specify a different set of attributes for the quality relations and different join conditions. For example, if the sources provided different quality data depending on the area, we would have the bounding box attribute in the quality relations and in the join condition.

Given the user query and the quality criteria, QGM utilizes the following query answering algorithm.

1. Invert both the content and quality descriptions of the sources using the Inverse Rules [6] algorithm.
2. Utilize the optimization techniques described in [13] to remove rules containing unnecessary source requests.
3. Collect source quality facts for all sources that appear in the relevant rules.
4. Execute the quality query using only the quality facts gathered in third step.
5. Utilize the results of the quality query to remove rules containing requests to sources that do not satisfy the quality criteria.
6. Execute the generated datalog program and return the high quality answers to the user.

In our running example, the first two steps result in a datalog program containing requests to three road vector data sources (*NavteqRoads*, *CasilRoads*, and *TigerRoads*) and the two image sources (*TerraServerImages* and *GoogleMapsImages*). From the quality facts shown in Table 5, QGM collects the facts for the three vector data sources and two image sources.

Once QGM has the quality facts for all relevant sources, it evaluates the quality query using the inverted quality definitions and the facts for the quality of data. Both image sources provide resolution better than 1 meters/pixel. Therefore, both image sources satisfy the quality criteria. Next, QGM performs the *SkylineMax* operation using the values of the *completeness* and the *features within accuracy bounds* attributes for the three vector data sources. The value for both attributes for the *TigerRoads* data source is lower compared to the other vector data sources, while both *CasilRoads* and *NavteqRoads* have the highest value for at least one of the two attributes. Therefore, only the *CasilRoads* and the *NavteqRoads* data sources satisfy the constraint of the quality of the road vector data specified using the *SkylineMax* operation. All possible combinations of road vector data and image data that satisfy the quality requirements are shown in Table 6.

Roads	Image
NavteqRoads	TerraServerImages
NavteqRoads	GoogleMapsImages
CasilRoads	TerraServerImages
CasilRoads	GoogleMapsImages

Table 6: Sources that Satisfy Quality Criteria

As a result of evaluation of the quality query, QGM can remove the rules containing requests to the *TigerRoads* data source as it does not satisfy the quality criteria. The pruning of the source not only reduces the source request(s), but also ensures that the answer returned to the user satisfies the quality criteria.

Finally, QGM executes the datalog program consisting of the inverted definitions of the *NavteqRoads*, the *CasilRoads*, the *TerraServerImages*, and the *GoogleMapsImages* data sources and the user query. The result of the query are four combinations of image and road vector data objects that satisfy the content and quality requirements specified in the user query.

6. EXPERIMENTAL EVALUATION

We conducted experiments to show that (1) QGM's automatic quality estimation technique with sampling provides accurate estimates of the quality of data provided by sources and (2) QGM can utilize the quality estimates to provide high quality data in response to user queries and reduce the response time of user queries.

We searched the Internet for shapefiles containing the following types of data covering parts of USA: (1) roads, (2) rivers, (3) hospitals, (4) schools, and (5) lakes. As a result of the search we found 1268 shapefiles containing different types of data. We used QGM to estimate the quality of those data sources using all the data provided by sources and using three sampling techniques. As a reference set, we used the vector data provided by Navteq.[7] As shown in Table 7, QGM can estimate the value for the features within the accuracy bounds attribute with less than 10% error and the value for the completeness attribute with about 20% error by sampling only 20% of the data. The major reason behind the large error bound for the completeness attribute is due to the fact that often the density of features is not strong in the sampled area resulting in a very small sample size, which is not enough to estimate completeness of the data source regardless of the sampling pattern. However, the difference in the true values of completeness between two data sources that provide the same type of data for the same region is usually larger than the average error in the estimation process. Therefore, the estimated quality results are good enough to determine the relative quality of sources to answer user queries.

The second set of experiments are to show that QGM significantly improves the quality of answers for the user queries. We used the 1268 sources used in the first set of experiments as available sources and randomly generated 20 bounding boxes where there was at least one type of geospa-

[7] http://www.navteq.com

Type	% Data	Avg. Comp. & Acc. Without Sampling		Avg. % Error with Sampling	
		Completeness	Accuracy	Completeness	Accuracy
Point	10	91.76	95.6	17.54	12.27
Point	20	91.76	95.6	14.27	7.95
Polyline	10	38.09	80.28	24.69	8.68
Polyline	20	38.09	80.28	20.74	7.95
Polygon	10	68.12	87.15	25.01	11.20
Polygon	20	68.12	87.15	20.51	10.97

Table 7: Results Estimating Quality

Type	QGM		Average		Std. Deviation	
	% Comp.	% Acc.	% Comp.	% Acc.	% Comp.	% Acc.
Constraint	**59.81**	87.61	47.71	83.12	17.36	9.31
Aggregate	**68.19**	**89.97**	47.71	83.12	17.36	9.31
Skyline	**64.03**	87.90	47.71	83.12	17.36	9.31

Table 8: Quality of Data

tial data available. We asked QGM to retrieve one type of geospatial data available in each bounding box using three different types of quality restrictions. The first quality restriction was on either completeness or features within accuracy bounds attribute. It contained a constraint that the selected attribute must be greater than 50. The second quality query asked QGM to find data with the highest value for completeness or features within accuracy bounds. Finally, the third quality query asked QGM a skyline query with the completeness and features within accuracy bounds.

QGM always returned sources with the best quality given the quality criteria. As shown in Table 8, on average compared to the average of all objects returned, QGM provided 12% more complete results for constraint queries, 20% more complete results for aggregate queries, and 16% more complete datasets for skyline queries. The data returned by QGM on average had 5% more features within the accuracy bounds compared to the average of all relevant sources. For all queries the improvement in completeness was statistically significant using a two-tailed t-test with α =0.05, while the improvement in the accuracy was also statistically significant for aggregate queries.

In addition to providing better quality data, QGM was able to answer queries in 33.7% less time compared to answering queries by returning data from all relevant sources regardless of quality. The 33.7% less time on average resulted in reduction of 221 seconds. This was due to the fact that on an average, QGM was able to remove two source requests out of six due to the quality constraints.

7. RELATED WORK

This paper is related to four major areas of work. The first area of related research is on integrating geospatial data using a data integration system. Hermes [1], MIX [12], VirGIS [8], and Geongrid [20] are examples of data integration systems that have been used to integrate geospatial data.

The Hermes mediator system [1] integrates multimedia data and spatial data was one type of data it integrated. The MIX system [12] is an XML-based mediator system that supports integration of geospatial data and some grouping and aggregation operations using the Global-As-View approach. VirGIS [8] is a mediator system that utilizes a limited form of Local-As-View approach to integrate data from various geospatial data sources. VirGIS supports one to one mapping between source and domain relations. The Geongrid [20] describes grid-enabled mediation services (GEMS) architecture for integrating geospatial data. The GEMS architecture focuses on providing the best results by selecting sources with the best quality using a pre-defined ranking of sources based on the quality metadata. While these systems address the challenges involved in representing geospatial sources and answering user queries, they do not allow users to specify any quality constraints. The key advantages of QGM are the support for automatic quality estimation to quickly add quality descriptions for sources, a declarative specification of the quality information, and the flexibility for the users to specify their own quality criteria.

The second area of related work includes work on semantic integration of geospatial sources [4, 9, 18]. Ontology-driven Geospatial Information System (ODGIS) [9] is an integration framework that allows users to browse various classes in the geospatial ontology, reasons based on ontology classes, and utilizes terms from Wordnet to resolve linkage issues between the sources. The ODGIS framework assumes that it has access to a semantic mediator that generates answers to the user queries by retrieving and integrating geospatial information from the relevant sources. QGM would be an ideal choice to work as a semantic mediator within ODGIS.

Arpinar et al. [4] describe the process of manually developing a geospatial ontology and modeling sources using the manually developed ontology in a framework titled Geospatial Semantics and Analytics (GSA). In [18] authors describe SWING, a semantic framework for geospatial services. The SWING framework shows the feasibility of utilizing an ontology-based reasoner to overcome the semantic heterogeneity in the names of layers and attributes in different geospatial services. Their approach is similar to the mediator approach as they use a simple form of description logic to encode the rules. The goal of QGM, GSA, and the SWING is to provide a unified interface to a large number of geospatial data sources. However, our work also addresses the quality of the integrated data and ease of estimating the quality of a large number of sources.

The third research area related to this paper is the work on quality-driven data integration [2, 7, 16, 17]. Berti-Equille and Eckman et al. [2, 7] describe an approach to ensure maximally complete answers for queries on life science data sources by analyzing all possible plans to compute data. QGM's representation of quality is more expressive as it can represent multiple attributes of quality and user-specified quality criteria. Naumann et al. [16, 17] use the quality metrics to generate a plan to answer the user query in three steps: (1) prune based on source specific quality criteria, (2) generate plan based on the logic rules for content, and (3) plan selection based on the logic rules for content and attribute-specific criteria. As geospatial datasets often have limited coverage pruning first based on only the source specific quality attributes may result in answers that do not cover the query area. Therefore, QGM's query answering algorithm first selects only the relevant sources based on the content and then explores the search space based on the quality requirements. As the number of sources that provide data for the layers and areas in the query tend to be small in general, the search space is relatively small. Moreover, QGM automatically estimates the quality of sources to enable integration from a large number of sources.

Finally, GIS researchers have worked on different approaches and ontologies to model and visualize accuracy in geospatial data [11, 5, 19]. The proposed framework would allow users to model various concepts and characteristics, such as completeness or alignment, as described in [19, 5], and pose queries to retrieve geospatial data that meets the accuracy requirements based on the given characteristics. The accuracy of data retrieved using our framework can be visualized using approaches described in [11].

8. CONCLUSIONS AND FUTURE WORK

In this paper, we described a framework to support quality-driven large-scale geospatial data integration. The key contributions of our framework are: (1) the ability to automatically estimate quality of data provided by a source by using the information from a source of known quality, (2) declarative representation of both the content and the quality of geospatial data provided by sources, and (3) a quality-driven query answering technique for geospatial data. Our experimental evaluation using over 1200 real-world sources show that QGM not only provides better quality data compared to the traditional data integration systems, it also has lower response time.

In the future, we plan to add the capability to automatically generate source descriptions for sources that support OpenGIS standards. Moreover, we plan to investigate automatic estimation of other quality attributes, such as date collected and original resolution. We believe that we can use information retrieval techniques to search the text or XML documents for the information about those attributes. Once QGM can automatically generate source descriptions and estimate the quality of data provided by the sources, we can build a truly automatic geospatial data integration system that can find geospatial sources by searching the Internet, automatically determine the type of data provided by the source and its coverage, estimate the quality of data provided by the source, and add the source to the list of available sources for future user queries.

Acknowledgments

This research is based upon work supported in part by the National Science Foundation under Award No. IIS-0324955, and in part by the Air Force Office of Scientific Research under grant numbers FA9550-04-1-0105 and FA9550-07-1-0416. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any of the above organizations or any person connected with them.

9. REFERENCES

[1] S. Adali and R. Emery. A uniform framework for integrating knowledge in heterogeneous knowledge systems. In *Proc of the Eleventh IEEE International Conference of Data Engineering*, 1995.

[2] L. Berti-Equille. Integration of biological data and quality-driven source negotiation. In *ER*, pages 256–269, 2001.

[3] S. Borzsonyi, D. Kossmann, and K. Stocker. The skyline operator. In *Proc. of the 17th International Conference on Data Engineering*, pages 421–430, 2001.

[4] I. Budak Arpinar, A. Sheth, C. Ramakrishnan, E. Lynn Usery, M. Azami, and M.-P. Kwan. Geospatial ontology development and semantic analytics. *Transactions in GIS*, 10(4):551–575, 2006.

[5] R. Devillers and R. Jeansoulin. *Fundamentals of Spatial Data Quality*. ISTE, 2006.

[6] O. M. Duschka. *Query Planning and Optimization in Information Integration*. PhD thesis, Stanford University, 1997.

[7] B. A. Eckman, T. Gaasterland, Z. Lacroix, L. Raschid, B. Snyder, and M.-E. Vidal. Implementing a bioinformatics pipeline (bip) on a mediator platform: Comparing cost and quality of alternate choices. In *ICDE Workshops*, page 67, 2006.

[8] M. Essid, O. Boucelma, F.-M. Colonna, and Y. Lassoued. Query processing in a geographic mediation system. In *Proc of ACM-GIS*, pages 101–108, 2004.

[9] F. Fonseca, M. Egenhofer, P. Agouris, and G. Camara. Using ontologies for integrated geographic information systems. *Transactions in GIS*, 6(3):231–257, 2002.

[10] H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. Integrating and accessing heterogeneous information sources in tsimmis. In *Proc. of the AAAI Symposium on Information Gathering*, Stanford, CA, 1995.

[11] M. GoodChild. Models for uncertainty in area-class maps. In W. Shi, M. F. Goodchild, and P. F. Fisher, editors, *Proc of the Second International Symposium on Spatial Data Quality*, pages 1–9, 2003.

[12] A. Gupta, R. Marciano, I. Zaslavsky, and C. Baru. Integrating gis and imagery through xml-based information mediation. In *Proc of NSF International Workshop on Integrated Spatial Databases: Digital Images and GIS*, 1999.

[13] S. Kambhampati, E. Lambrecht, U. Nambiar, Z. Nie, and S. Gnanaprakasam. Optimizing recursive information gathering plans in EMERAC. *Journal of Intelligent Information Systems*, 2003.

[14] M. Lenzerini. Data integration: A theoretical perspective. In *Proc of ACM Symposium on Principles of Database Systems*, Madison, WI, USA, 2002.

[15] A. Levy. Logic-based techniques in data integration. In J. Minker, editor, *Logic Based Artificial Intelligence*. Kluwer Publishers, 2000.

[16] F. Naumann. From databases to information systems - information quality makes the difference. In *Proc of the International Conference on Information Quality (IQ)*, Cambridge, MA, 2001.

[17] F. Naumann, U. Leser, and J. C. Freytag. Quality-driven integration of heterogenous information systems. In *Proc of 25th International Conference on Very Large Data Bases*, pages 447–458, 1999.

[18] D. Roman and E. Klien. Swing - a semantic framework for geospatial services. In K. T. Arno Scharl, editor, *The Geospatial Web. In the Advanced Information and Knowledge Processing Series*, pages 227–237. Springer, 2007.

[19] M. F. Worboys and E. Clementini. Integration of imperfect spatial information. *Journal of Visual Languages and Computing*, 12:61–80, 2001.

[20] I. Zaslavsky and C. Baru. Grid-enabled mediation services for geospatial information. In *Proc of workshop on Next Generation Geospatial Information*, 2003.