Planning for Geospatial Data Integration

Snehal Thakkar
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Ray, CA 90292
thakkar@isi.edu
Advisor: Dr. Craig A. Knoblock

Introduction

In the recent past there have been significant research and industrial efforts to support access to large amount of geospatial data sources. Examples of industrial efforts include different data access standards proposed by the Open Geospatial Consortium (OpenGIS)[1] and the availability of satellite imagery and other data on websites, such as TerraServer[2].

In order to better understand the research challenges involved in designing a geospatial data integration system, I co-developed an application called the Building Finder (Michalowski *et al.* 2004). The Building Finder application integrates satellite imagery from TerraServer, road network information from the Tiger Line files[3], and building information from various assessor's web sites.

Our experiences with the Building Finder showed that geospatial data integration involves six major research aspects: (1) accessing geospatial data, (2) modeling geospatial data sources and operations, (3) query reformulation, (4) optimization, (5) execution, and (6) visualizing the results. However, all the efforts have mainly focused on providing access to various geographic datasets and visualizing the available geospatial data. In my thesis plan to focus on modelling geospatial data sources and operations, reformulating user queries into an integration plan consisting of a set of queries on available data sources and operations, and optimizing the generated plan.

In this abstract, I present my research on developing a geospatial data integration framework that can be utilized to rapidly generate applications, such as the Building Finder. Figure 1 shows the architecture of my framework. In particular, I focus on the work that I have done on modelling data sources and complex operations, query reformulation, and optimization. I conclude this abstract with a discussion of the work that remains to be done.

Geospatial Data Modeling

All the previous geospatial data integration systems, such as the mediator systems developed by Gupta et al. (Gupta *et al.* 1999) and Essid et al. (Essid *et al.* 2004), have used the Global-As-View (Garcia-Molina *et al.* 1995) approach to model the available data sources. While the GAV approach allows easy reformulation of user queries, addition of new data sources may require changing entire domain model. I utilize the Local-As-view (LAV) (Levy 2000) approach to model available data sources. In the LAV model each data source is described as a view over the domain predicates, therefore it is easier to add sources in a LAV model.

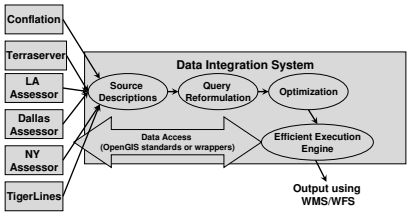

Figure 1: Architecture

I explicitly model the characteristics of geospatial data, such as alignment, accuracy, and coverage. Moreover, I model complex geospatial operations, such as conflation that change the characteristics of the geospatial data. My data integration framework utilizes the characteristics of the requested data to determine which geospatial operations should be performed to answer the user query.

Modeling Data Sources: If we have a imagery data source, such as TerraServer, which provides satellite imagery for the United States, we can provide the following description to the integration system.

```
Terraserver(imagetype, toplat,toplon, botlat, botlon,
        height, width, imageurl, resolution):-
    Image(imagetype, toplat, toplon, botlat,
        botlon, height, width, imageurl, resolution),
    (toplat>17.84),(botlat<71.55),
    (toplon>-168.67),(botlon<-65.15),
    (imagetype='satellite image')
```

Modeling Complex Operations: In addition to modeling data sources, we also need to provide descriptions of available operations, such as *ImageVectorConflation*. This operation accepts a satellite image and a road vector data of the same region and provides a conflated road vector data that is aligned with the given satellite image. By explicitly modeling the operations, the data integration system can determine based on the user query which operations should be per-

[1] http://www.opengis.org
[2] http://terraservice.microsoft.net
[3] http://www.census.gov/geo/www/tiger

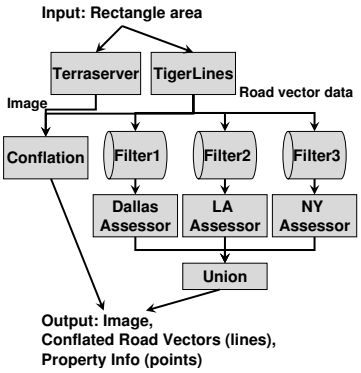

Figure 2: Example Integration Plan

formed to answer the user query. As some operations may take a long time to execute, it is important for the integration system to only use the operations that are required.

```
ImageVectorConflate(imageurl, vectordata, conflatedvectordata):-
    Image(imagetype, toplat, toplon, botlat,
        botlon, height, width, imageurl, resolution),
    Vector(vectortype, toplat, toplon, botlat,
        botlon, vectordata, accuracy, coverage),
    (imagetype='satellite image'), (vectortype='road'),
    Vector(vectortype, toplat, toplon, botlat,
        botlon, conflatedvectordata, accuracy, coverage),
    Aligned(imageurl, conflatedvectordata)
```

Query Reformulation and Optimization

In order to answer the user queries the data integration system must reformulate the user query to an integration plan containing a set of source queries. As requests to geospatial data sources and operations can be very time consuming, the integration framework must optimize the generated plan.

Query Reformulation: In previous work (Thakkar, Ambite, & Knoblock 2004) we have built a data integration system called Prometheus that extends the Inverse Rules algorithm (Duschka 1997) to generate an integration plan that can answer the user queries.

Optimization Using Tuple-level Filtering: The idea behind Tuple-level filtering (Thakkar, Ambite, & Knoblock 2004) is to utilize the constraints in the source descriptions to reduce the number of requests sent to a data source. Consider the integration plan for the Building Finder, which is shown in Figure 2. We have three data sources that provide assesment information: (1) *LAAssessor*, (2) *DallasAssessor*, and (3) *NYAssessor*. All the data sources provide property assesment information, but they have different coverages. The initial integration plan generated using the Inverse Rules algorithm would contain requests to all three data sources to find assessment information. My framework would utilize the Tuple-level filtering technique to insert filters before each source request to ensure that the integration plan only requests information from relevant data source. For example, it only queries the *LAAssessor* data source if the area of interest is in the county of Los Angeles.

Proposed Work

A useful geospatial data integration system must reason about the completeness of various data sources. Consider a scenario where the integration system has access to the following road vector data sources: (1) Tiger Line files for the state of California, (2) Tiger Line files for the county of Los Angeles, and (3) Tiger Line files for the Orange county in California. When the integration framework gets a query to get conflated road vector data for the region covering Los Angeles and Orange counties, it would query all three data sources. Moreover, it would then have to conflate the results from all three data sources. However, if the integration framework could reason that all three data sources are complete, i.e. they provide information about all the roads in their respective coverage area, then it can decide to query only the first data source.

I am currently working on including the completeness information for each source in the domain model. Next, I will work on extending the query reformulation algorithm to utilize the completeness information to generate alternative plans that answer the user query. For example, for in the above-mentioned scenario, two alternative plans would be to query the Tiger Line data source for the state of California or to query the other two Tiger Line data sources. Next, I will utilize cost-based optimization techniques to evaluate the cost of the generated integration plans and find the most cost-efficient integration plan.

References

Duschka, O. M. 1997. *Query Planning and Optimization in Information Integration*. Ph.D. Dissertation, Stanford University.

Essid, M.; Boucelma, O.; Colonna, F.-M.; and Lassoued, Y. 2004. Query processing in a geographic mediation system. In *12th ACM International Workshop on Geographic Information Systems(ACM-GIS 2004)*, 101–108.

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

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

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

Michalowski, M.; Ambite, J. L.; Knoblock, C. A.; Minton, S.; Thakkar, S.; and Tuchinda, R. 2004. and semantically integrating heterogeneous data from the web. *IEEE Intelligent Systems* 19(3):72–79.

Thakkar, S.; Ambite, J. L.; and Knoblock, C. A. 2004. A data integration approach to automatically composing and optimizing web services. In *Proceedings of 2004 ICAPS Workshop on Planning and Scheduling for Web and Grid Services*.