GENERATING QUERY EVALUATION PLANS WITHIN A SPATIAL MEDIATION FRAMEWORK

Amarnath Gupta, Ilya Zaslavsky and Richard Marciano
San Diego Supercomputer Center, University of California San Diego

1 Introduction

The goal of a spatial mediation system is to give a user the ability to issue a single query that would access multiple sources to retrieve different pieces of the result and would assemble these pieces to provide a composite response to the query. In (Gupta *et al* 1999) we described the general architecture of a system to accomplish a similar integration for geographic information stored in a GIS system and georeferenced images stored in an image database management system. In this paper we investigate the issues involved in mediating among different GIS sources in the process of query evaluation.

Our specific goal is to investigate how a spatial mediation system takes into account the representational heterogeneity of measured or derived spatial information stored in the GIS to be integrated. This heterogeneity may arise from a number of reasons. Different sources may store coordinates in one of many projections, with different precision. They may present data accumulated at different times, measured with different accuracy through different procedures. Data layers may be raster or vector, they may have or lack topological organization, and may use different and possibly incompatible data structures and indexing schemes to store and access spatial data.

Integration of such heterogeneous spatial sources within a mediator system becomes possible only if:

a) Every source exports a minimal set of "capability" information to the integration system. We will present a capability specification scheme for a spatial data source.

b) The query evaluation engine of the system plans for explicit data transformation steps in addition to data retrieval, data combination and data restructuring steps that are usually performed in non-spatial information integration systems.

We posit that an optimal query plan process will strongly depend upon how the retrieval costs are balanced against the transformation costs. While we leave a detailed cost analysis outside the scope of this paper, we illustrate the influence of transformation costs on the overall query processing through a concrete example scenario.

We now present a brief description of the wrapper-mediator model of information integration, followed by an account of our reference architecture.

1.1 Information Integration in the Wrapper-Mediator Architecture

A mediator-based system is a data integration architecture that supports homogeneous views (in a common data model) over heterogeneous data sources. Mediator systems are based on a 3-level architecture, which includes a "foundation" layer (data sources with *wrappers*), a mediation layer (which supports exchange of queries and results between wrapped legacy data sources and applications), and an application/user interface layer (Wiederhold, 1992). The advantage of this architecture is its modularity and scalability. These systems support combining query results from individual sources rather than combining the data. In addition, the use of a semistructured data model at the mediator enables the modeling of sources with no structure or implicit structure. Examples of such semi-structured mediator-based systems include TSIMMIS (Papakonstantinou *et al* 1995, 1996), DISCO (Tomasik *et al* 1998), and Information Manifold (Levi *et al* 1996). Examples of the use of this approach for geospatial data are the Aquarelle project (INRIA) and the research described in (Shimada and Fukui 1999) and (Bishr *et al* 1999).

The MIX (Mediation of Information using XML) project develops a type of a wrapper-mediator system where data sources expose themselves as XML sources, user queries and query fragm ents are expressed in an XML query language, and query results are presented as XML documents. The XMAS query language (Baru *et al* 1999) developed as part of the MIX project, is an XML query language for expressing user queries and relaying them to an application mediator. used to . It builds upon ideas from languages such as XML-QL (Mather 1995), Yat (Cluet *et al* 1998), MSL (Papakonstantinou *et al* 1996), and UnQL (Buneman *et al* 1997). XMAS allows object fusion (e.g., combining an image reference from one source and a map reference from another source into a new composite object) and pattern matching on the input XML data. Additionally, XMAS features powerful grouping and order constructs for generating new integrated XML "objects" from existing ones.

In (Gupta *et al* 1999) we have extended the MIX wrapper-mediator architecture to perform spatial data integration. If the application mediator detects that a query evaluation requires accessing geographic data sources or services, it delegates a particular query fragment to a spatial mediator (Figure 1). This happens in two situations: either variables declared in a particular XMAS query match data elements from a schema exported to the mediator by a geographic source, or the predicates used in a query are contained in a source capabilities description. Using the query fragments supplied by the application mediator, and information about source capabilities, the spatial mediator determines an optimal query evaluation plan. This plan is formulated in a common language understood by GIS source wrappers, which translate the query fragments into the language of a particular source, initiating data retrieval or processing at each of them. According to the query execution plan, responses to query fragments are exchanged between sources, so that eventually a virtual map is assembled and sent to the user application.

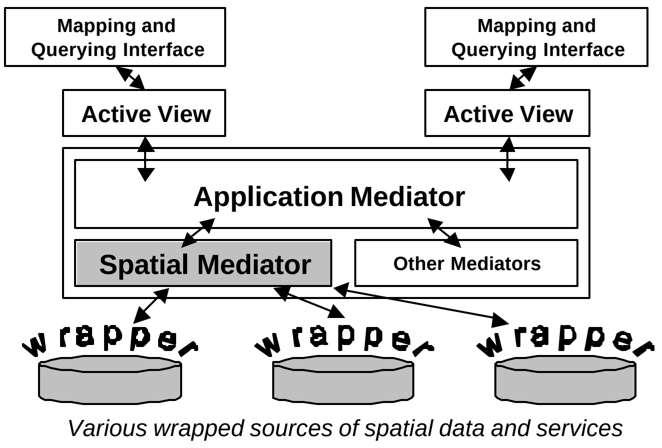
Mapping and Querying Interface
Mapping and Querying Interface
Active View
Active View
Application Mediator
Spatial Mediator
Other Mediators
wrapper
wrapper
wrapper
Various wrapped sources of spatial data and services

Figure 1: The architecture of a system of mediators and wrappers for heterogeneous information integration.

Example 1. Consider a query: *"Find all parcels with Total Assessed Value (TAV) > $500K located within neighborhood Carmel Valley"*. Assume that there are five data sources in the system as shown in Table 1. For the moment, we disregard the column headings. Note that data sources are logically distinct in the table, while in reality different data layers may be available at one physical server, often combined with a range of processing functions.

Schema element	**Collection Name**	**Spatial extent**	**Source**
TAV	Parcel layer	San Diego County	A
Neighborhood name	Beat layer	San Diego City	E
Elevation	7.5' DEM	San Diego Quadrant	C
Street speed limits	Street layer	San Diego County	B
City coordinates	Gazetteer	United States	D

Table 1: Heterogeneous information sources with different spatial extents, accuracy and data types.

The mediator goes through the following steps to formulate the query.

- Source Selection: The mediator only needs sources A and E to construct the query result. The mediator can select the source in a number of ways. In the first method, the system designer may have made explicit rules (or views) to encode one-to-one mappings between attributes and sources.

This is the approach taken in the original MIX system (Baru *et al* 1999). A more general approach is to dynamically evaluate the minimal combination of sources that satisfy the query attributes and predicates. This requirement becomes important if several sources can supply similar and possibly redundant data, and is adopted in this paper. Clearly, in either method sources A and E would be selected for the example query.

- Subquery Formation: Once the sources are selected, the mediator uses a set of rewrite rules that convert the appropriate fragment of the user's query for each selected source. The rewrite rule constructs the subquery by selecting the attributes to be returned and the predicates to be satisfied.
- Query Planning: In practice, there will be a number of different ways in which sources can be selected. Hence, the mediator needs to formulate different sets of subqueries and determine the one that can be performed with the least cost, where the cost is determined by expected time and resources necessary for evaluating the subquery, estimated error propagation, etc. In addition, one subquery may be dependent on the result of another, forcing them to be executed in sequence. The query plan is a partial order on subqueries so that the complete query is executed in an optimal fashion.

In this paper we will formalize the process of query formulation taking into account the fact that when data is passed between a source and the mediator, or between two sources, it may need to be transformed.

1.2 Organization of the Paper

In Section 2 we develop a capability language for spatial information sources with which a spatial data source can inform a mediator about its data and query handling properties. We describe the process by which the mediator dynamically integrates the capabilities of multiple spatial sources for a user query, and show that this process is equivalent to a graph construction problem. In Section 3 we use an example to trace the steps by which a mediator builds an evaluation plan from source schemas and query and transformation capabilities of the sources. Finally, we conclude in Section 4, with an outlook into our future work.

2 Capabilities of a Spatial Information Source

The capabilities of an information source are a specification of its information content, and of the methods it provides to an external entity to access that information. The information content is specified as the ***export schema***, while the methods can either be ***query capabilities*** or ***transformation capabilities***. In our usage, query capabilities refer to the methods by which a source gives access to its own stored data, and transformation capabilities refer to the methods by which an information source can temporarily accept non-resident data and return a computed result without updating its own content.

2.1 The Export Schema

The mediator uses the export schema to select a source and pass on a subquery to it. Hence the two goals of a source's export schema are to provide information about all the data items and item groups on which the user may formulate a query, and provide enough information to help the mediator determine when the source should be selected. For mediators used in the database literature (Cluet *et al* 1998), this selection is often made by choosing the smallest subset of sources that export all query attributes. For spatial information, however, such an approach will be inadequate because the semantics and representation of an attribute will affect its selection for specific queries. For example, if, using data in Example 1, we just request a parcel having the highest total assessed value, we do not need to transform the data provided by the source A. However, if a query requests the average elevation of all regions whose elevation is less than 500ft., we need to realize that this may trigger an interpolation operation in source C, and the interpolation cost may affect the choice of the source. We believe that the spatial mediator needs to have a built-in framework to handle not only the geometric component but also the semantics of spatial information. In this paper we adapt Chrisman's (Chrisman 1997) "measurement framework" approach to specify a part of spatial semantics.

From the database point of view our schema model is simple. A *data element* has a *name*, a *value*, and a *measurement framework*. Quite possibly, the name of a data element can be chosen from a well-recognized *namespace* that standardizes the names with fixed definitions (such as SDTS (Spatial Data Transfer Standard), or the emerging GML (Geography Markup Language) which encodes Open GIS "simple features"). A value may belong to a system or user-defined *type*. Scalars, sets, lists, tuples, points and polygons are examples of system-defined types. The user may also define a tree-structured type, for example, for use in a hierarchical classification of remotely sensed imagery. A number of data elements can be grouped into a named *collection*. A collection of data elements can be unordered or structured. Specifically, a *layer* containing a tuple of polygons and other non-spatial data is an unordered collection and a *grid* or a *TIN* representation of spatial data is interpreted as a structured collection. A collection may have a number of collection metadata including, for example, the coordinate system used to describe the data elements in it. A *schema* is a set of collections.

The measurement framework of a data element is a statement that specifies the condition in which the value of the data element was measured. Chrisman's (1997) measurement semantics applies to data elements that represent geographic objects or properties. In our model, the schema engineer can declare a data element to have a *null* measurement framework to signify that it is not a geographic property. Otherwise, a measurement framework of a data element is a structure of the form {*element type*, *control type*, *measurement basis*}, where the **element type** states if the data element is spatial, non-spatial or relationship. A relationship element has a list of pointers to other data elements that it relates. For example, "driving speed" is a relationship element that points to two nodes of a road network and represents the driving speed between those two points. It is often likely that the type system of the user will further specialize an element type into more refined subtypes. The **control type** of a data element is the name of the data element (or elements) that is used to regulate the variation of the current element. For example, the data element called "population" may have the data element called "census tract" as its control attribute. Given the uniqueness of the "census tract" values, this represents the classic case of functional dependency in relational database theory. On the other hand, the control type of the spatial type "elevation contour" (a line through all points having the same elevation value) is the data element called elevation. Finally, the **measurement basis** of a data element refers to a (hopefully standard) procedure (method or function) that was used to derive the value of the data element. For example, if the control type of the data element is *spatial* and the spatial data element is a region, the data value may have been derived by a classification function, or a cascade of statistical operations (e.g., median of a 30% sampling) performed on the region. If the control of the data element is an *attribute*, the measurement basis typically is a condition on the attribute (e.g. condition "elevation=200ft" defining an elevation contour.) In the next section we will point out why the semantics specified by the measurement framework impacts the process of optimal query evaluation.

Although not specified here, a data element may also have a number of additional attributes, among which possibly the most significant is the timestamp on the data and a specification of its accuracy. It is also possible to attach to a measurement framework a value function that computes a favored value. Such a function could be as simple as a precedence function A < B < C < D, stating that if the framework were to choose among different values for the data element, D is the most preferred value assignment and A is the least preferred. A situation where this may arise is labeling a grid region by its land use. If the grid actually has multiple land use patterns, a preference criterion may be set by ranking by the relative area covered by each land use.

2.2 Query Capabilities

The query capabilities of an information source comprise a set of query operations, which, given a set of selection conditions, retrieve data items or collections that satisfy the conditions. In the information integration literature, query capabilities have been specified using a number of methods such as binding patterns, query templates (Papakonstantinou *et al* 1995), and capability records (Levy *et al* 1996). In all of these methods the primary idea is to identify a set of predicates and functions supported by the source, and enlist their input and output parameters using the elements of the schema. For example, *intersect(+polygon:$p1, +polygon:$p2, -polygon:$p3)* specifies a binding pattern. It states that the source supports an operation called "intersect" that takes two polygons as input (+) and returns a third polygon as output (-). Several of these methods do not work very well when the information source supports a powerful engine. For example, describing query capabilities of an ArcView source using a method such as a binding pattern, would require a very large number of attribute combinations.

We take an algebraic approach to specify query capabilities, primarily because most spatial sources can be better modeled with a procedural query language rather than a declarative one. The intuitive reason is that a GIS source is typically designed to be interactive, performing one step at a time, and the operations are typically governed by the type of object on which they are executed. To this end, we use existing algebraic models wherever possible, as summarized below:

- For all non-spatial data types we use common relational operators.
- For all spatial data types that are points, lines and polygons, we use the ROSE algebra operators (Güting and Schneider 1995). Mapping from the ROSE algebra to the internal operations supported by the source can be non-trivial. However, this is a wrapper design problem and we have, for a few sources, designed wrappers to translate ROSE algebra operations to native operations.
- For all grid spatial sources, including raster data, we use an array algebra (Libkin *et al* 1996). In this algebra one can not only access any subarray of a larger array, but can also compute analytic data elements such as histograms, which can then be used for a subsequent part of the query.
- For line networks, we assume a minimal set of operations. For a node, one can request for the IDs of the edges coming into the node, and those going out of the node. For an edge, one can request the ids of the two nodes it connects. We assume that the node and the edge elements have properties of other data types, and can be queried using the appropriate algebra.
- For all other data elements, the source has to explicitly declare a list of operators in the form of a signature: opName(type_1, type_2,…) $\rightarrow$ type_n, where the right hand side of the arrow specifies the return type of the operation.
- We assume that regardless of the type of the data element, the source allows variable assignment, and thus an expression like a = b op c is valid. We also assume that a collection allows set operation including membership checking and element insertion and removal.

2.3 Transformation Capabilities

Our notion of spatial data transformations is tuned to the context of spatial information mediation. Consider a single spatial data source S_1. Suppose the user sends a request $Q(r_q)$ to S_1 where r_q is a spatial region supplied as a constant parameter in the query, and the system responds by producing a result $R(Q(r_q))$ in the form of a map. The user only receives the map (that the user's viewer can interpret) as the final answer and has no access to the internal spatial data structure revealing the internal properties of R. In this case we ignore the fact that the source might have performed any internal transformations of r_q or of any other intermediate data in producing R. If there are two sources S_1 and S_2, such that the final result can be produced only by combining data from both, several situations, such as those depicted below, could arise:

- The mediator performs no post-processing of the partial spatial information returned by either source and the user's viewer can simultaneously interpret and present the combined result.
- The sources send their results using different data structures. The mediator needs to convert them to a common structure that the user's viewer can interpret
- Under the mediator's guidance, partial results from one source are transmitted by its wrapper to the second source. The second source converts these data into a common structure, performs some operation to combine the two partial results and transmits the combined result to the mediator. As another variation, even the first source can first perform the conversion before transmitting it to the receiver of the data.

We consider the last two instances to be examples of data transformation, because the receiver of the data (the mediator or another source) cannot accept the nominal form of the data produced by the producer data source.

In this framework, a transformation is a function f that operates on data item d to produce data item $d¢$. If $d= d¢$ then f is called an *identity transform*. We do not include identity transforms in our discussions in the paper. Different kinds of transforms, ranging from simple coordinate transforms to more complex conversion between different spatial data structures, have been used in the GIS and cartography literature (Tobler 1979, Clarke 1995, Chrisman 1997). From the data management point of view we take the following approach to transformation modeling.

A transformation function f is completely specified by the following parameters:

- Domain ***dom****(f)*: Type of the source data item
- Range ***ran****(f)*: Type of the target data item
- Constant Parameters [**P**]: a list of parameters necessary for the computation of *f*, dependent on the measurement frameworks of the source and target data items. These parameters are constant in the sense that they do not

depend on the properties of the data. An example would be the 3×3 kernel required for transforming spatial coordinates.

- Data-dependent Parameters [**V**]: a list of parameters that are dependent on the data. This list is also controlled by the measurement frameworks of the source and the target. We distinguish between two categories of data-dependent parameters. For the simpler category only the value of the parameter depends on the data. In the second category, the size of the parameter is also variable and depends on the size of the data.
- The computation function itself.

From the mediator's perspective we can also model data transformations as graph elements. In this model, given a network **N**={**V**, **E**} of information producers and consumers, data transformation takes place along a directed edge $e \in \mathbf{E}$ from node v_1 to node v_2 ($v_1, v_2 \in \mathbf{V}$), iff:

- *data*(v_1) is transmitted from producer node
- *data*(v_2) is the form in which it is accepted at the consumer node
- *data*(v_2)=*f*(*data*(v_1)), where function f is not an identity transform. In general, we do not require that any inverse function f^{-1} be defined.

In this framework a query Q is modeled as a subgraph such as the one shown in Figure 2. In this graph M_{qg}, the subquery generator of the mediator, decomposes the original query Q into three subqueries SQ1, SQ2 and SQ3.

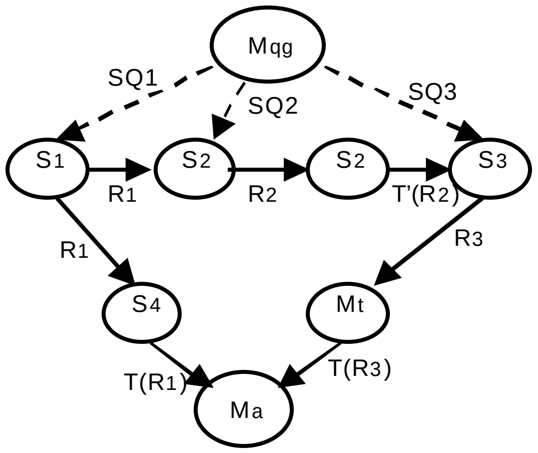

Figure 2: A query can take the form of a subgraph in which results are retrieved, transformed and collected

Note that source S_2 needs R_1, the result of SQ1, to produce its own result R_2. Source S_3 needs a transformed version of R_2 along with the subquery SQ3 to produce its partial result R_3. Source S_4 is shown in the role of a pure data transformation agent, in contrast with source S_2, which can perform data retrieval as well as data transformation. M_t illustrates the fact that the mediator itself can serve as a data transformation agent converting R_3 to the form $T(R_3)$ before merging the result with $T(R_1)$ in M_a, its assembly component.

At any edge showing a transformation in the subgraph above, the computation of the spatial data transformation may be affected by a number of factors. These include:

- The complexity of computing the function *f* itself
- The size of any parameter necessary to compute the function. In simple cases, such as bicubic interpolation, the parameters are independent of the data. In more involved computations, the parameters may need to be derived using the data to be transformed. For example, the spatial density of data points may influence the approximation functions used to perform an interpolation.
- The size of the support set. Let **D** be the total set of data sent by one node to another in a single unit. The support set *S(d)* of a data element $d \in \mathbf{D}$ is the set of other data elements in **D** such that the function *f* can be computed only if *S(d)* is available. For an operation like reprojection of data points, the support set is 0, while for an operation like neighborhood computation, the support set can be a $k \times k$ matrix around the data point. If computing *f* needs statistical aggregates such as mean and variance, the support set for any data element can be the entire data set **D**.
- The number of iteration (or recursion) steps needed to transmit **D** to the destination node. This often depends on the buffer capacities available for data transfer both at the source and destination.

3 Generation of query evaluation plans based on source capabilities: an example

Generally speaking, any changes in information content during GIS processing can be treated as geographic transformations, using the measurement framework scheme mentioned above. While some of the transformations (like projections, for example) require little in terms of accompanying metainformation (since this is a transformation within one object dimension and doesn't involve attributes), others may need explication of attribute assumptions or explicit construction of feature neighborhood to make the transformation possible. In our example with the "polygon in polygon" request, the user has to be alerted about which particular version of the "within" request is being used, since this will affect any measures of the output. The following sample query explores a more complex situation involving data structure transformation.

<u>Example 2</u> Find all parcels with TAV>500K located on slopes over 15% within 30 minutes drive from downtown San Diego.

Spatial mediator receives this query expressed in some declarative query language (XMAS, in our architecture), and generates the schema and the function graphs for the query. The **schema graph** is generated based on schema information exposed by data sources, as in Table 1.

These schema elements are found either because they are explicitly requested in the query (such as TAV), or are functionally related to query variables (such as the elevation map which is used to produce a slope map; or street speed limits used to produce a 30-minute isochrones map), or are semantically equivalent within a namespace that a source subscribes to (in our example, downtown location is expressed through city coordinates found in a gazetteer).

A schema graph generated for this query (see Figure 3) follows the traditional cartographic modeling (Tomlin 1990, Burrough 1986) techniques showing a sequence of data transformations leading to the final map. An important difference is that the query integrates information from different sources, and mediator-directed transformations take place at different sources as well.

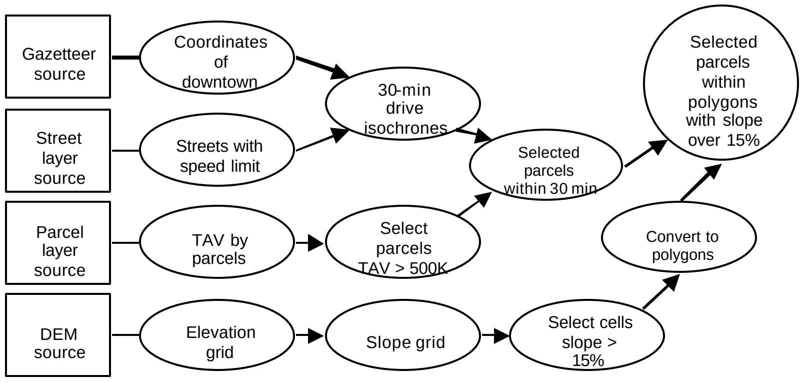

Figure 3: A schema graph for the example query

Note that this graph is just one from a set of possible schema graphs that are generated in the mediator for a given query. The graph is generated by selecting data elements with their containers from a list of data elements which sources expose to mediator as part of schema metadata. Alternate schema graphs generated for the same query would reflect other query evaluation strategies. For example, in one strategy the DEM would be converted to TIN at the first step, so that triangles with slope over 15% can be overlaid with the selected parcels. Or, one could compute the proportion of grid cells with slope over 15% within each parcel, to classify it as either located on a slope or not.

Using the generated collection of schema graphs, the mediator represents data elements to be used in actual query evaluation, and identifies candidate data sources. However, the mediator hasn't yet examined whether these sources are capable of supporting queries and transformations expressed in a particular schema graph, and therefore cannot determine how sources should exchange and transform the data. To reflect the transformation and query capabilities of sources in the evaluation plan, the mediator assembles the appropriate sources in a companion graph which we called a **function graph**.

This graph is generated from a list of all transformations and query types supported by the sources known to the mediator. For our situation, this list would contain:

- **f_1: T_SlopeFromDEM;**
  T_{in} = elevation grid;
  T_{out} = slope grid;
  sources: $S_i(f_1)$
- **f_2: Q_SelectByTheme;**
  Q_{in} = {target polygon theme; mask theme; selection condition= "within"}
  Q_{out} = polygon theme;
  sources: $S_i(f_2)$
- **f_3: T_GridToPolygon;**
  T_{in} = binary grid;
  T_{out} = polygon theme;
  sources: $S_i(f_3)$
- **f_4: Q_SelectByAttribute;**
  Q_{in} = {theme, selection condition}
  Q_{out} = polygon theme;
  sources: $S_i(f_4)$
- **f_5: T_isochrones;**
  T_{in} = {streets with street speed attribute, coordinates of origin);
  T_{out} = polygon theme;
  sources: $S_i(f_5)$

For each transformation and query in the graph, we know which sources S_i can implement it. One of possible function graphs for the sample query is shown in Figure 4.

Each operation in this graph references a list of sources where the operation is implemented. However, operations may be implemented differently at different sources, and therefore may have different computation costs, assumptions and error propagation functions. While computational complexity of several core GIS operations has been discussed in GIS literature (Chrisman *et al* 1992, Worboys 1995, etc.) and error propagation functions have been proposed for some of them (Lanter and Veregin 1992, Heuvelink 1998), this research theme is by no means exhausted. In our sample query, slope can be computed by fitting a plane through four or eight neighboring points on the elevation grid, or by taking maximum or average of four or eight adjacent gradients. A specific best-fit plane model implemented in software, or a neighborhood used to compute the slope, are typically not accessible. Also, these techniques will perform more or less accurately depending on terrain conditions. Inaccuracy of raster to vector conversion, and especially difference in accuracy between the parcel layer and DEM, may render overlay results unreliable. However, if a query plan involves converting DEM to TIN as an intermediate step, this could result in a particularly unreliable result. Clarke (1995, p. 267) mentions that there are no simple algorithms yet for DEM to TIN conversion, other than dividing every grid cell into two triangles.

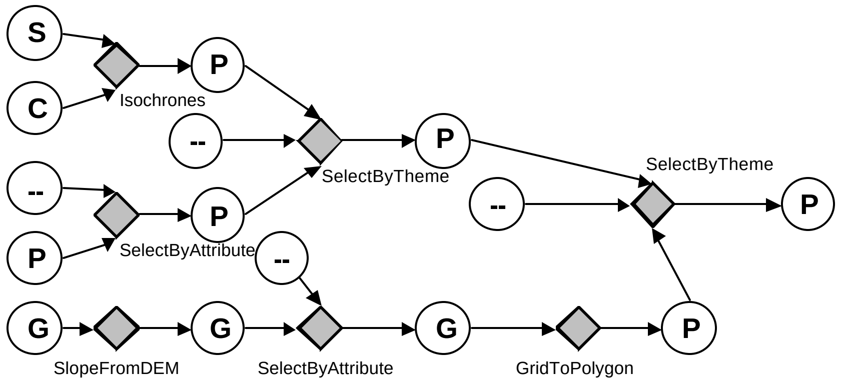

Figure 4: Function graph for the sample query. In this graph, "G" stands for grid, "S" for street; "C" for point; "P" for polygon data container types. The "—" indicates that the operation input is supplied by the XMAS query directly.

Several ambiguities need to be resolved in formulating the query. We will discuss this using the ***within*** predicate of the SelectByTheme query, which is expressed above in software-independent syntax. Mediator checks this syntax against the available descriptions of source capabilities, which include lists of supported predicates, to formulate query fragments interpretable by the source wrappers. When the query is parsed into fragments, spatial predicates are re-specified depending on the capabilities of the source and using the set of spatial algebra operations implemented in the mediator. The ***within*** predicate, in particular, can translate into Avenue's #FTAB_RELTYPE_ISCOMPLETELYWITHIN predicate of the SelectByTheme request, which is the ArcView's expression of the *Area-Inside* predicate of the mediator's algebra (here, we use constructs of the ROSE Algebra described in (Güting and Schneider 1995). A particular GIS wrapper may not support the *Edge-inside* or *Vertex-inside* predicates of the ROSE Algebra (e.g. they do not directly map into ready Avenue constructs). If a source supports a variety of "within" flavors, it exposes this capability to mediator, which in turn alerts the user asking to specify which particular version of "within" to use. For example, if one of sources underneath the mediator may additionally support function *centroid_within(region1, region2)*, the spatial mediator will export it to application mediator and on to the user interface, to make it available for the user.

Once a collection of query schema and function graphs are generated, an optimal merge of them is computed by the mediator, using information about servers associated with branches of the graphs (i.e. servers containing particular data elements – in the schema graph, and servers implementing particular query and transformation functions – in the function graph). The result will be a graph similar to the one shown in section 2.3 of the paper. In this example query, we consider parcel map residing on source A, a street map residing on source B, a USGS DEM residing on source C, and a gazetteer residing on D. We also know that source A is an ArcView source (i.e. it supports SelectByTheme and SelectByAttribute queries from our function graph), source B is a transportation GIS supporting the Isochrone request, source C is a raster-based GIS, and source D supports simple coordinate extraction requests. Our joined query graph is then expressed as a sequence of procedures relayed by the mediator to each source, as shown below:

R_1 *= Q(source = "D", name="San Diego")*
$R_2 = T_{isochrones}$ *(source = "B", obj = "Streets", center =* R_1*, time = "30")*
$R_3 = T_{slope\_from\_DEM}$*(source = "C", obj = "Elevations")*
R_4 *= Q(source = "C", obj =* R_3*, condition = "R3.slope > 15")*
$R_5 = T_{polygon\_from\_DEM}$*(source = "C", obj =* R_4*)*
R_6 *= Q(source = "A", layer = "parcels", condition = "TAV > 500000")*
R_7 *= SQ(source = "A", obj1 =* R_6*, obj2 =* R_2*, condition = "completely_within")*
R_8 *= SQ(source = "A", obj1 =* R_7*, obj2 =* R_5*, condition = "completely_within")*
Return Map(source = "A", obj=R_8*)*

In this sequence of operations, each operation is assigned to a particular GIS source.

4 Conclusion

This paper describes an extended wrapper-mediator architecture, which we call the spatial mediation framework. In this framework, constructing a response to a spatial query to multiple sources is driven by a mediator which is expected to perform spatial source selection, subquery formation, and query planning. These tasks are facilitated by a source's ability to "publish" or export its capabilities. Capabilities can express both the underlying querying mechanisms as well as the transformational abilities of a source.

Future research will further refine the embedding of these methodologies into the spatial wrapper mediator framework and explore query graph optimization (on the plan graph), looking at issues of scalability and performance.

References

Baru C, Gupta A, Ludäscher B, Marciano R, Papakonstantinou Y, Velikhov P and Chu V. (1999) *XML-Based Information Mediation with MIX*. Proc. of the *ACM SIGMOD'99*, June 1-3, Philadelphia, PA, USA, pp597-599.

Bishr Y A, Pundt H and Rüther C (1999) *Proceeding on the Road of Semantic Interoperability - Design of a Semantic Mapper Based on a Case Study from Transportation*. In Vckovski A, Brassel K E and Schek H-J (eds) Proc. *Interoperating Geographic Information Systems*, Second International Conference, INTEROP '99, Zurich, Switzerland, March 10-12, 1999, pp203-215.

Buneman P, Davidson S B, Fernandez M F and Suciu D (1997) *Adding Structure to Unstructured Data*. In *6th Intl. Conference on Database Theory* (ICDT), Delphi, Greece, 1997. Lecture Notes in Computer Science , Vol. 1186, pp336-350, Berlin, Springer-Verlag.

Burrough P A (1986) *Principles of Geographical Information Systems for Land Resources Assessment*. Oxford, Oxford University Press.

Chrisman N R, Dougenic J A and White, D (1992) *Lessons for the design of polygon overlay processing from the ODYSSEY WHIRLPOOL algorithm*. In Proceedings of the *5th International Symposium on Spatial Data Handling*, Charleston, SC: August 3-7, 1992 (Columbia, SC: University of South Carolina), V 2, pp401-410.

Chrisman N (1997). *Exploring Geographic Information Systems*. John Wiley & Sons.

Clarke K C (1995) *Analytical and Computer Cartography*. Prentice Hall.

Cluet S, Delobel C, Simeon J and Smaga K (1998) *Your Mediators Need Data Conversion!* In Proceedings of the *1998 ACM SIGMOD International Conference on Management of Data*, pp177-188.

Gupta A, Marciano R, Zaslavsky I and Baru C (1999) *Integrating GIS and Imagery through XML-based Information Mediation* In Agouris P and Stefanidis A (eds) *Integrated Spatial Databases: Digital Images and GIS*, Lecture Notes in Computer Sci ence, Vol. 1737, pp211-234, Berlin, Springer-Verlag.

Güting R H and Schneider M (1995) Realm-Based Spatial Data Types: The ROSE Algebra, *VLDB Journal*, 4, 100-143.

Heuvelink G B M (1998) *Error Propagation in Environmental Modeling with GIS*. London, Taylor & Francis.

Lanter D P and Veregin H. (1992) A research paradigm for propagating error in layer-based GIS. *Photogrammetric Engineering and Remote Sensing*, 58, 825-833.

Levy A Y, Rajaraman A and Ordille J J (1996) *Querying Heterogeneous Information Sources Using source Descriptions*, In Proceedings *of the 22nd VLDB Conference*, Mumbai (Bombay), India, pp251-262.

Libkin L, Machlin R and Wong L (1996) *A Query Language for Multidimensional Arrays: Design, Implementation, and Optimization Techniques*. In *SIGMOD Conf. Proc.*, pp228-239.

Mather P M (1995) Map-image registration accuracy using least-squares polynomials. *Int. J. Geographical Information Science* 9, 543-554.

Papakonstantinou Y, Gupta A, Garcia-Molina H and Ullman J D (1995) *A Query Translation Scheme for Rapid Implementation of Wrappers*. In Proceedings of the *International Conference On Deductive and Object Oriented Databases*, pp161-186.

Papakonstantinou Y, Abiteboul S, and Garcia-Molina H (1996) *Object Fusion in Mediator Systems*. In *International Conference on Very Large Data Bases (VLDB)*, 1996.

Shimada S and Fukui H (1999) *Geospatial Mediator Functions and Container-based Fast Transfer Interface in SI^{3}CO Test-Bed*. In Vckovski A, Brassel K E and Schek H-J (eds) Proc.

Interoperating Geographic Information Sys tems, Second International Conference, INTEROP '99, Zurich, Switzerland, March 10-12, 1999, pp265-276.

Tobler W. (1979) The Transformational View of Cartography. *The American Cartographer* 6, 101-106.

Tomasic A, Raschid L and Valduriez P (1998) Scaling Access to Heterogeneous Data Sources with DISCO. *IEEE Transactions on Knowledge and Data Engineering*, 10, 5, 808-823.

Tomlin C D (1990) *Geographic Information Systems and Cartographic Modeling*. Englewood Cliffs, New Jersey, Prentice Hall.

Wiederhold G (1992) Mediators in the Architecture of Future Information Systems. *IEEE Computer*, 25, 3, 38-49.

Worboys M F (1995) *GIS: A Computing Perspective*, London, Taylor & Francis.