On Aggregation Issues in Spatial Data Management

M. Indulska, M. E. Orlowska

School of Computer Science and Electrical Engineering

The University of Queensland

Brisbane, Australia

marta@csee.uq.edu.au, maria@csee.uq.edu.au

Abstract

Large amounts of information can be overwhelming and costly to process, especially when transmitting data over a network. A typical modern Geographical Information System (GIS) brings all types of data together based on the geographic component of the data and provides simple point-and-click query capabilities as well as complex analysis tools. Querying a Geographical Information System, however, can be prohibitively expensive due to the large amounts of data which may need to be processed. Since the use of GIS technology has grown dramatically in the past few years, there is now a need more than ever, to provide users with the fastest and least expensive query capabilities, especially since an approximated 80% of data stored in corporate databases has a geographical component. However, not every application requires the same, high quality data for its processing. In this paper we address the issues of reducing the cost and response time of GIS queries by pre-aggregating data by compromising the data accuracy and precision. We present computational issues in generation of multi-level resolutions of spatial data and show that the problem of finding the best approximation for the given region and a real value function on this region, under a predictable error, in general is NP-complete.

Keywords: Geographical Information Systems, Spatial Data Aggregation, Data Warehousing

1 Introduction

Geographical databases have been extensively studied over many years and many sophisticated systems have been developed (Orlowska, Lister and Fogg 1992). Geographic Information Systems (GIS) are "tools that allow for the processing of spatial data into information, generally information tied explicitly to, and used to make decisions about, some portion of the earth" (Demers 1997). Thus a typical GIS stores various data about geographical areas represented as maps.

Research also shows that an approximated 80% of data stored in large corporation databases has a geographical component (Gonzales 2000). "The full benefits of using spatial data can be achieved by combining the data from different sources covering a common region" (Orlowska and Zhou 1999). Many real life applications would substantially benefit from introducing spatial representations of the data, rather than the currently available textual references to the names and descriptions of the regions, their sub-regions and so on. The introduction of a spatial component would undoubtedly be most powerful in a Data Warehousing environment, where vast amounts of historical data are queried for decision making purposes. The decision making process could thus be further enhanced by the addition of spatial representation of data and its patterns. Currently, however, the cost and response time of GIS queries is a limiting factor for the GIS itself let alone for Spatial Data Warehouse performance.

The increasing importance of spatial data analysis (Shekhar, Chawla and Ravada 1999), the need for efficient and cost effective data warehousing and the rapid growth of GIS usage suggests that there is a need for a solution which will reduce the query cost and time in Geographical Information Systems in turn making it feasible to integrate a GIS with a Data Warehouse.

A map in a GIS is made up of a set of pre-defined regions. We assume that for each spatial region on a given map, data such as pollution, precipitation, temperature or vegetation cover is collected from various sources (such as surveys, aerial photos and sensors) and stored in the GIS. Such data normally is expressed as a real value function(s) defined over the specified region, often with a single value for a component polygon of the region. The granularity of the data is naturally dictated by the way the data is collected and how it is intended to be later used in applications. The data can then be queried with different levels of precision subject to specific application requirements.

Querying the original data, however, can be an expensive and time-consuming task. In practice, there may be a large number of maps (regions) stored in a GIS, furthermore each such region may be made up of vast amounts of sub-regions (polygons). Each polygon on such a map will have associated with it a value for each function being measured (e.g. temperature) at a given point in time. If a user, for example, wants to compute the average temperature of the entire region depicted in the temperature map at a given point in time, then the computation of the result would involve retrieving the value for each region and then calculating the average. In practice, this can be 'serious' computation since vast amounts of data must be processed. The question is whether there is an effective way to reduce this computation requirements at run time and provide controlled quality of service for the user.

In this paper, we approach the problem of aggregation query cost at run time. The response time reduction can be achieved by storing materialised views of the original

data. This in turn requires additional computation prior to the query execution and additional memory requirements for storing the pre-defined data views. We focus on computational analysis of methods to achieve the best data approximation within the pre-specified range of data precision.

1.1 Related Research

Current research into the problem of Spatial Data Aggregation or Spatial Data Warehouse Design is not extensive (Han et al 1998, Stefanovic 1997, Zhou et al 1999). This area, however, draws on several well established streams of research; relational Data Warehouse design (Baralis et al 1997, Golfarelli et al 1998, Golfarelli and Rizzi 1999, Gupta 1997, Indulska 2000, Inmon 1996, Theodoratos et al 1999, Yang et al 1997), Data Warehouse maintenance (Colby at al 1996, Gupta et al 1993, Mumick et al 1997, Quass and Widom 1997) and geographical database systems (Demers 1997, Gonzales 2000, Orlowska et al 1992, Orlowska and Zhou 1999, Shekhar et al 1999).

Han and Stefanovic (1998) propose an algorithm which attempts to reduce the number of polygons which have to be processed in order to answer a query. Only queries which require the spatial merge operation of a region are considered. The authors make the observation that pre-merging for all records in all views is too expensive but not pre-merging is also not a desirable solution. The aim is to discover which sets of polygons should be pre-merged and stored and which can be merged on-line.

The algorithm presented by Han and Stefanovic takes as input a partially materialised spatial data cube, the expected view access frequency and a region depicting the connectivity of the polygons. For each view and each record in the view, the 'spatial region' cell containing the polygons for merging is intersected with all other 'spatial region' cells in the view and with 'spatial region' cells in the remaining views in the data cube. This process is repeated for every cell in every view. The method is not efficient when a large number of views is considered. Also intersection of cells from different views can be meaningless in some cases, resulting in materialisation of polygons which in fact are never queried together in isolation from other polygons. Additionally, this method is not very accurate since it relies on query frequency data which cannot be fully predicted.

Zhou, Truffet and Han (1999) propose a method for efficient merging of a set of selected polygons in order to obtain the boundary of the set. The authors make the observation that not all polygons belonging to the set of mergeable polygons actually contribute to the boundary of the target polygon. The method thus avoids fetching all polygons in the set from the database, fetching only those which contribute to the boundary. Since neighbouring polygons will share some identical line segments, the algorithm selects boundary cells by selecting polygons which consist of lines for which no identical lines exist. The boundary polygons are then merged by removal of identical line segments between neighbouring polygons. This algorithm significantly reduces the cost of polygon amalgamation. This approach, however, merges only the selected set of polygons and does not determine which polygons should actually be merged, this selection is determined by the user. Although, the problem of spatial data aggregation is related to amalgamation of polygons it is different from our focus. Our focus is on the selection of polygons will need to be aggregated without consideration of how they will actually be amalgamated in the end.

1.2 Contribution and Organization of the Paper

In this paper, we propose a concept for aggregation of spatial data based on the 'similarity' of values of a function defined over the region. Initially, we limit our consideration to only one real function and assume that the function may have only one value for each polygon in the region. We prove that the best aggregation with respect to the number of resulting new polygons for a given function, in general is an NP-complete problem.

The method presented aggregates polygons and computes a new aggregated function of a region, based on the initial values of the function which is being measured over that region (for example temperature or precipitation). We show that such aggregation can be performed at different levels of granularity resulting in a number of aggregations for a single function, each with a known guaranteed error measure. These pre-computed data views can then be used to provide a choice (within the available selection of views) of Quality of Service (QoS) to the user thereby reducing the computation cost and time of certain queries. It is important to note that the aggregation is performed without consideration of queries or query frequencies which are difficult to predict. Rather, it is based on the behaviour of the function being measured as well as the physical connectivity of the polygons.

The remainder of the paper is organized as follows; Section 2 provides the motivation for this work. Section 3 introduces the problem scope, the formalisms and the problem statement. In Section 4 the aggregation process is presented (along with an example) and it is shown to be NP-complete in Section 5. We conclude our findings in Section 6 and discuss planned future work in this area in Section 7.

2 Motivation

Let us first consider an example. We assume a relation PRECIPITATION(pID, date, prec_amount) exists, which stores the daily precipitation (in millimetres) for each polygon in a given region R..

The region being considered is depicted in Figure 1(a). It is divided into a number of polygons, where the value depicted for each polygon is the value of the precipitation function for that polygon at a particular point in time. Naturally, a region can be partitioned into thousands of polygons and it may have many different functions defined over it; however this simple example will suffice in order to demonstrate the general problem scope. Let us now consider a query which requires the average

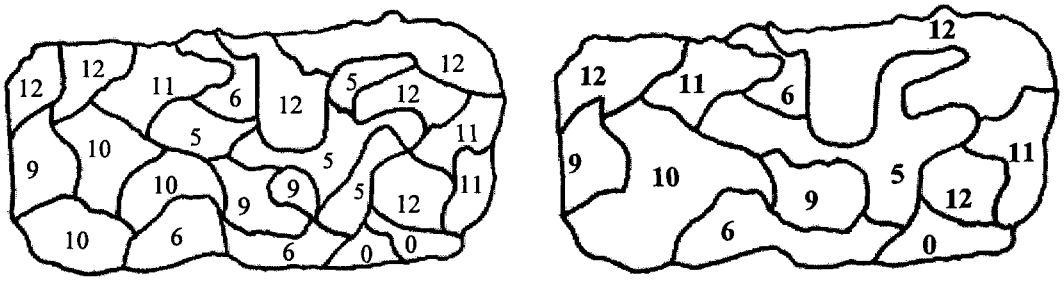

Figure 1: a) Initial region R partition, b) Region R partition after polygon aggregation

precipitation on a given day of areas that received more than 9 millimetres of rain.

In order to evaluate this query, all polygons which received more than 9 millimetres of rain on the specified day are accessed and then the average precipitation is calculated. This requires accessing 12 polygons and calculating the average of 12 values. However, after closer observation, it can be seen that some neighbouring polygons actually have the same precipitation values. These neighbouring polygons can therefore be aggregated without loss of accuracy. This results in a new partition of polygons for the region R (see figure 1(b)) and requires storage of the new polygon identifier values and their respective precipitation values, however the original partition of region R is still retained in the database. If we assume that such aggregation took place, in order to answer the above query only 6 polygons need to be accessed and the average of 6 values calculated as opposed to 12 polygons and their respective values.

In this example, only the polygons with equal values of the function were aggregated. We can however introduce a different granularity of aggregation which will result in an aggregation with some loss of accuracy. For example, one can aggregate neighbouring polygons where the difference of precipitation is no more than one millilitre. One such possible aggregation is shown if Figure 2. In this case the precipitation value for the aggregated polygons is the average value of the precipitation function of polygons which now form the new polygon. However it can be seen that there are many possible aggregations in this case, the problems associated with finding an aggregation (also referred to as a "cover") will be addressed in Section 4.

The above described process results in a hierarchy of aggregations for the region R, where each aggregation has a different error measurement associated with it (discussed in Section 7). When the user issues a query which concerns the region R and the function which was used in the aggregation of the polygons in that region, the user is presented with a number of available and already computed aggregations for the region and the particular function. As previously stated, each such level of aggregation has associated with it an error measurement. For some user queries the data accuracy level required may be 100%, however for many queries complete accuracy is not required and may be relaxed to an extent. In either case, depending on the requirement of data accuracy for a given user query, the user chooses the appropriate aggregation level (appropriate according to the error measurement the user is prepared to accept) for region R from the aggregation hierarchy in order to answer the query, as depicted in Figure 3.

Thus the lower the accuracy of the result that the user is prepared to accept, the lower the cost and response time of the query and the "quality" of evaluation. The creation of such aggregation hierarchies, however, has a cost associated with it. Once such a hierarchy is created it will also have to be maintained as the underlying data changes. Therefore this process is beneficial when the total cost reductions of all queries executed on the hierarchy of aggregations outweighs the cost of creation and maintenance of the aggregation hierarchy. However, considering that once such a hierarchy is computed it will support multiple queries as well as multiple users, the cost of computation should be offset by the savings in query cost (this of course depends on the frequency of query execution).

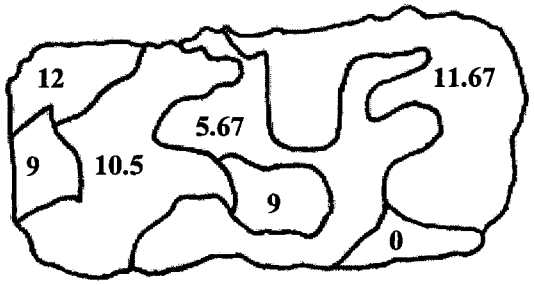

Figure 2: Region R partition after another level of polygon aggregation

3 Problem Definition

Let us now introduce the formalism necessary to reason about levels of aggregations.

We define a region R with an initial (also referred to as level zero) non-overlapping partition of polygons $\boldsymbol{P}^0$, where $\boldsymbol{P}^0 = \{P^0_1, P^0_2, ..., P^0_{n_0}\}$ is a collection of polygons which together form region R. That is,

$$R = \cup P^0_{i0} \text{ and}$$

$$\forall(i, j)\ i, j \in I^0 \mid P^0_{i0} \cap P^0_{j0} = \varnothing;$$

where $I^0 = \{1, 2, ..., n_0\}$ is the level zero index value.

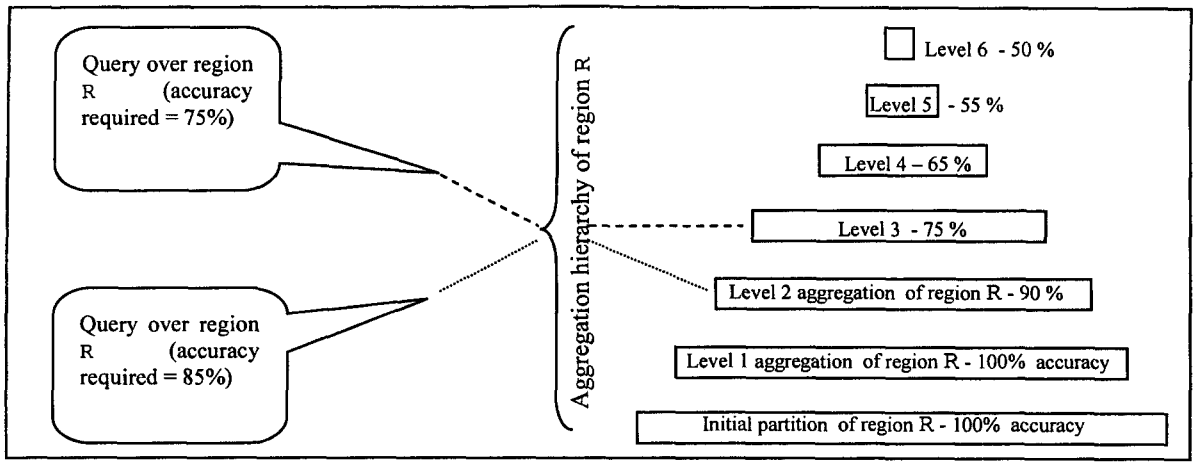

Figure 3: A basic depiction of the querying process.

The index is required since aggregations at different levels may consist of a different number of polygons (i.e. different values of n).

We can therefore introduce the next level partition (the level one aggregation) in a similar manner;

$$\boldsymbol{P}^1 = \{P^1_1, P^1_2, \ldots, P^1_{n_1}\} \quad \text{such that}$$

$$R = \cup P^1_{i\,1} \quad \text{and}$$

$$\forall (i, j)\ i, j \in I^1 \mid P^I_{i\,1} \cap P^I_{j\,1} = \varnothing;$$

where $I^1 = \{1, 2, \ldots, n_1\}$.

Naturally, the level one partition $\boldsymbol{P}^1$ is obtained through aggregation of the polygons in the initial $\boldsymbol{P}^0$ partition and it is important to note that a polygon can in itself be a collection of polygons without any loss of generality. Also, since aggregation of polygons at any level cannot produce an aggregation level with more polygons than in the underlying level, there is a restriction on the indexes of each aggregation level such that the number of polygons at the level k partition is greater or equal to the number of polygons at level k+1 partition, i.e. $n_k \geq n_{k+1}$.

Let us now illustrate the above concept through the use of an example. We consider a region R which has an initial partition consisting of five polygons,

i.e.

$$\boldsymbol{P}^0 = \{P^0_1, P^0_2, P^0_3, P^0_4, P^0_5\},$$

as depicted in Figure 4(a) below. Let us also assume that a polygon aggregation has taken place resulting in a level one cover of region R consisting of three polygons, that is

$$\boldsymbol{P}^1 = \{P^1_1, P^1_2, P^1_3\},$$

as depicted in Figure 4(b) below. It is of course important to note that the values of the function defined over these polygons were also aggregated according to some aggregation function (in this paper only the average function is considered).

Hence, the aggregation produced a new level where polygon P^0_1 still exists in the same form as in the level zero partition, but where polygons P^0_2 and P^0_3 were merged to form a new polygon P^1_2 and their vales were aggregated. Likewise, polygons P^0_4 and P^0_5 were merged to form a new polygon P^1_3 and also had their values aggregated. While the purpose of this paper is not to state how the polygons are merged but which polygons are actually merged, we would like to point out that the method presented by Zhou, Truffet and Han (1999) can be used to accomplish this merging process.

$$P^0_1 \rightarrow P^1_1$$

$$\{P^0_2 \cup P^0_3\} \rightarrow P^1_2$$

$$\{P^0_4 \cup P^0_5\} \rightarrow P^1_3.$$

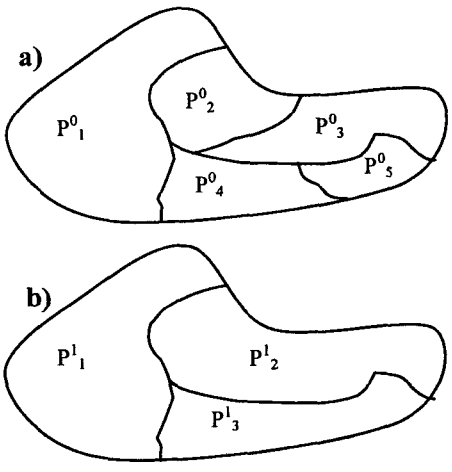

Figure 4. a) Level zero partition of region R b) Level one aggregation of region R

Up to this point we have not specified any conditions which manage the merging process but rather focused on

what merging is and how subsequent levels will be constructed.

Let us now introduce the next level of formalisation. We define a function

$$f: \{P^k_i\} \rightarrow \Re, \forall\, i \in I^k\, f(P^k_i) = v^k_i,$$

which for each level k polygon P^k_i of a region R returns the real number value v^k_i which is the value of the measured function f for polygon P^k_i.

We introduce a partition or aggregation level as a partition $\boldsymbol{P}^k_{f\,\varepsilon}$, which corresponds to a level k partition which was obtained by aggregating polygons of level k-1 based on the values of the function f with an "similarity value" of ε. The similarity value is used to determine whether any two neighbouring polygons can actually be aggregated to form one new polygon. Thus if we, for example, assume that ε =m where $m \in \Re$ (where $\Re$ is the set of all real numbers), then only neighbouring polygons where the difference of the values of function f is less than or equal to m can be aggregated. Thus, for any partition $\boldsymbol{P}^k_{f\,\varepsilon}$, the following must always hold for each element P^k_j in partition $\boldsymbol{P}^k_{f\,\varepsilon}$:

$$\forall (i, j)\ i, j \in I^{k-1} \mid f(P^{k-1}_i) - f(P^{k-1}_j) \mid \leq \varepsilon \quad \textit{and}$$

$$(P^{k-1}_i, P^{k-1}_j) \text{ are connected.}$$

Connectivity in this context is assumed to exist if any two polygons share a polygon boundary which is at least one point.

Additionally, each polygon, at any level k of the aggregation hierarchy, has associated with it a value of its physical area, such that

$$A: \{P^k_i\} \rightarrow R\ \forall\, i \in I^k\ A(P^k_i) = a^k_i;$$

which represents the area of polygon P_i belonging to a level k aggregation. This physical area can be used in the aggregation process to increase accuracy by calculating the new polygon's function value based on the area weighted underlying polygon values.

We also define an average function $f^{\varepsilon}_{AVG}(P^k_m)$ for each element P^k_m of a partition $\boldsymbol{P}^k_{f,\varepsilon}$, which calculates the value of the function f for the element P^k_m. This is calculated as the average (weighted by area) of the function f values of the underlying polygons at level k-1 which were aggregated to form P^k_m based on an similarity value of ε.

$$f^{\varepsilon}_{AVG}(P^k_j) = \sum a^{k-1}_i * v^{k-1}_i \,/\, \sum a^{k-1}_i \text{ where } i \in J \subseteq I^{k-1}$$

Finally, we define the error of a partition $\boldsymbol{P}^k_{f,\varepsilon}$, as $E(\boldsymbol{P}^k_{f,\varepsilon})$ which is the average difference in function values weighted by polygon area (see Section 7).

Thus given a region R, a function f and similarity values $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k$, we are now required to find a sequence of partitions (level 1, f_{AVG}, ε_1), (level 2, $f^{\varepsilon_1}{}_{AVG}, \varepsilon_2$), ..,(level k, $f^{\varepsilon_{k-1}}{}_{AVG}, \varepsilon_k$) such that each partition provides the best approximation of the function f, $f^{\varepsilon_1}{}_{AVG}$, $f^{\varepsilon_{k-1}}{}_{AVG}$ respectively.

The "best approximation" can be viewed from three different perspectives however. First of all, one may be interested in finding a partition such that there exists a minimum number of polygons. This requires finding ALL possible partitions and selecting the ones with the minimum number of polygons. Such a task is very hard for a region with even 10 polygons let alone a larger region in a GIS.

For instance, starting at the initial partition, we require a level one partition (level 1, f, ε_1) such that it is a minimum partition and it satisfies the following condition:

$$\min \sum_j \left(\sum_i \mid f^{\varepsilon}{}_{AVG}(P^1_j) - v^0_i \mid \right).$$

Secondly, one may chose to impose a restriction on the above, by specifying the number of polygons which is to be present in the generated cover. Therefore instead of finding the minimum cover, the problem now becomes finding a cover with k elements while still minimizing the difference of values between the newly aggregated polygons and their underlying polygons.

$$\min \sum_k \left(\sum_i \mid f^{\varepsilon}{}_{AVG}(P^1_k) - v^0_i \mid \right).$$

Lastly, one may specify a range of acceptable error of the generated cover. While the first two variations of the problem are similar to each other, this one is quite opposite. In this problem variation the importance shifts from the number of elements in the cover to the actual acceptable error of the cover.

In this paper we concentrate on finding the minimum partition of a region.

4 Generating Minimum Region Partitions

A minimum partition of a region is a partition which has the smallest number of elements, or polygons, from all other possible partitions which are created based on the same conditions. Thus, if we were to create all possible partitions at a particular level of aggregation, the minimum partition would be the partition(s) which had the smallest number of polygons among all generated partitions.

In this section we explain and show in detail (through the use of an example) the process of finding a minimum partition without generating all possible partitions. This is done through the generation of a number of graphs and matrices. Initially the connectivity graph and the ε-graph (as well as their corresponding adjacency matrices) are generated which show the physical connectivity of the polygons and the function behaviour respectively. These are then used to construct the ε-connectivity-matrix and it's corresponding graph, which shows both the connectivity and function behaviour. The ε-connectivity-graph is then extended, due to reasons explained later on, to allow partition generation.

We illustrate the process with the use of the following example. We consider a region R with a polygon partition $\boldsymbol{P}^0 = \{P^0_1, P^0_2, P^0_3, P^0_4, P^0_5\}$ shown in Figure 5 below where the values of a function f being measured over R are shown in bold. Let us also assume that the value of ε is 1 and that all polygons are of the same size.

It is important to note that not all of the presented graphs/matrices have to be generated in order to find a minimum partition. One could for example start the process by constructing the ε-connectivity-matrix without constructing the connectivity and ε-graphs. The generation of these, however, is shown because it is easier to construct the ε-connectivity-graph by constructing it from the ε-connectivity-matrix which can in turn be automatically generated from the connectivity and the ε-matrices.

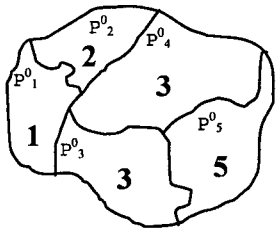

Figure 5. A region R with an initial level zero polygon partition $\boldsymbol{P}^0$.

4.1 Partition Construction Process

Given a region R with a polygon partition $\boldsymbol{P}^k$, a graph can be constructed which represents the connectivity of the polygons in $\boldsymbol{P}^k$. We call such a graph a connectivity graph $G_C(\boldsymbol{P}^k)$.

$$G_C(\boldsymbol{P}^k) = (V, E)$$

where each v_i in V represents a polygon P^k_i in $\boldsymbol{P}^k$, and where vertices (v_i, v_j) are connected by an edge in E iff the polygons P^k_i and P^k_j are directly connected to each other in $\boldsymbol{P}^k$.

Having constructed the connectivity graph $G_C(\boldsymbol{P}^k)$ we generate an adjacency matrix from $G_C(\boldsymbol{P}^k)$, called a connectivity matrix $M_C(G_C(\boldsymbol{P}^k))$.

To illustrate the above, we construct a connectivity-graph $G_C(\boldsymbol{P}^0)$ as well as its adjacency matrix $M_C(G_C(\boldsymbol{P}^0))$ for the region shown in Figure 5. This graph and its corresponding matrix is shown in Figure 6 below. As previously stated, there exists and edge between any two nodes in this graph if the polygons represented by those nodes are physically connected in Figure 5. We also construct another graph which represents the difference of values between the polygons with respect to ε. This graph is called the ε-graph and is referred to as

$G_\varepsilon(\boldsymbol{P}^k{}_{f\,\varepsilon})$.

$$G_\varepsilon(\boldsymbol{P}^k{}_{f\,\varepsilon}) = (V, E)$$

where any two vertices (v_i, v_j) in $G_\varepsilon(\boldsymbol{P}^k{}_{f\,\varepsilon})$ are connected by an edge in E iff $|f(p_i) - f(p_j)| \leq \varepsilon$, without any assumption of connectivity. Once again we generate an adjacency matrix $M_\varepsilon(G_\varepsilon(\boldsymbol{P}^k{}_{f\,\varepsilon}))$, called the ε-matrix.

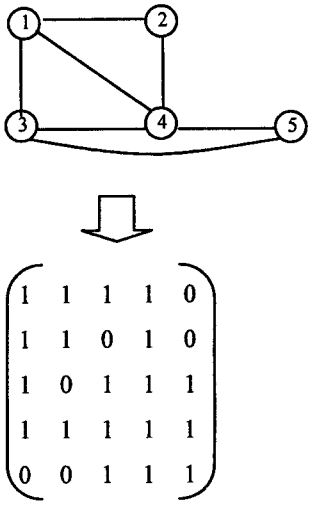

Figure 6. The connectivity-graph $G_C(\boldsymbol{P}^0)$ and it's corresponding connectivity-matrix $M_C(G_C(\boldsymbol{P}^0))$.

Applying this process to the example, we proceed to construct the ε-graph $G_\varepsilon(\boldsymbol{P}^0{}_{f\,1})$. After the ε-graph is constructed, we construct the adjacency matrix $M_\varepsilon(G_\varepsilon(\boldsymbol{P}^0{}_{f\,1}))$; both of which are shown in Figure 7. Note that edges in this graph exist only if the difference of function values between the two polygons is less than or equal to 1.

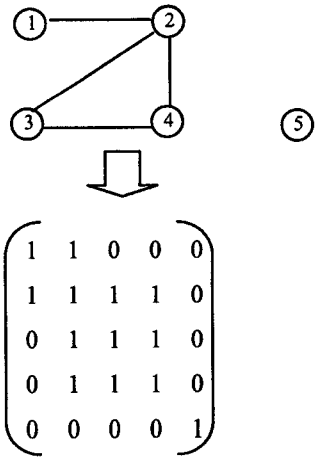

Figure 7. The ε-graph $G_\varepsilon(\boldsymbol{P}^0{}_{f\,1})$ and its corresponding ε-matrix $M_\varepsilon(G_\varepsilon(\boldsymbol{P}^0{}_{f\,1}))$.

Having generated the connectivity-matrix $M_C(G_C(\boldsymbol{P}^k))$ and the ε-matrix $M_\varepsilon(G_\varepsilon(\boldsymbol{P}^k{}_{f\,\varepsilon}))$, we can now generate the ε-connectivity-matrix $M_{C\varepsilon}(G_{C\varepsilon}(\boldsymbol{P}^k{}_{f\,\varepsilon}))$.

The ε-connectivity-matrix is a n*n matrix where n = |V| in $G_C(\boldsymbol{P}^k)$. Each element a_{ij} in $M_{C\varepsilon}(G_{C\varepsilon}(\boldsymbol{P}^k{}_{f\,\varepsilon}))$ will have a value of either of (0,1);

$a_{ij} = 0$ iff (v_i, v_j) are not connected in the connectivity-graph $G_C(P^k)$) ***OR*** if (v_i, v_j) are not connected in the ε-graph $G_\varepsilon(P^k{}_{f\,\varepsilon})$;

$a_{ij} = 1$ iff (v_i, v_j) are connected in the connectivity-graph $G_C(P^k)$) ***AND*** if (v_i, v_j) are connected in the ε-graph $G_\varepsilon(P^k{}_{f\,\varepsilon})$.

The ε-connectivity-matrix $M_{C\varepsilon}(G_{C\varepsilon}(P^k{}_{f\,\varepsilon}))$ can be easily generated by a simple operation on matrices $M_\varepsilon(G_\varepsilon(P^k{}_{f\,\varepsilon}))$ and $M_C(G_C(P^k))$. Based on the above definition of $M_{C\varepsilon}(G_{C\varepsilon}(P^k{}_{f\,\varepsilon}))$, it can be seen that the only time a_{ij} will be equal to 0 will be when the product of a_{ij} in $M_C(G_C(P^k))$ and a_{ij} in $M_\varepsilon(G_\varepsilon(P^k{}_{f\,\varepsilon}))$ is equal to 0. Likewise, the only possibility of a_{ij} being equal to 1 is when the product of a_{ij} in $M_C(G_C(P^k))$ and a_{ij} in $M_\varepsilon(G_\varepsilon(P^k{}_{f\,\varepsilon}))$ is equal to 1.

Thus we introduce an operation ⊛ which is a 'special' multiplication of two n*m matrices. This operation reflects the conditions of polygon merging. Namely, it expresses the conjunction of the two conditions introduced above.

Let us consider a n*m integer matrix A and an equally sized integer matrix B. We can define operation ⊛ as

$$A \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix} \circledast B \begin{pmatrix} b_{11} & a_{12} & \dots & b_{1m} \\ b_{21} & a_{22} & \dots & b_{2m} \\ \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & b_{nm} \end{pmatrix}$$

$$= C \begin{pmatrix} a_{11}*b_{11} & a_{12}*b_{12} & \dots & a_{1m}*b_{1m} \\ a_{21}*b_{21} & a_{22}*b_{22} & \dots & a_{2m}*b_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1}*b_{n1} & a_{n2}*b_{n2} & \dots & a_{nm}*b_{nm} \end{pmatrix}$$

follows;

We now generate $M_{C\varepsilon}(G_{C\varepsilon}(P^k{}_{f\,\varepsilon}))$ by applying ⊛ to $M_\varepsilon(G_\varepsilon(P^k{}_{f\,\varepsilon}))$ and $M_C(G_C(P^k))$;

$M_{C\varepsilon}(G_{C\varepsilon}(P^k{}_{f\,\varepsilon})) = M_\varepsilon(G_\varepsilon(P^k{}_{f\,\varepsilon})) \circledast M_C(G_C(P^k))$.

From the ε-connectivity-matrix we are now able to generate the ε-connectivity-graph. $G_{C\varepsilon}(P^k{}_{f\varepsilon}) = (V, E)$; where any two nodes (v_i, v_j) are connected by an edge in E iff $a_{ij} = 1$ in $M_{C\varepsilon}(G_{C\varepsilon}(P^k{}_{f\,\varepsilon}))$.

We apply the ⊛ operation to the matrices generated in our example to obtain the ε-connectivity-matrix $M_{C\varepsilon}(G_{C\varepsilon}(P^k{}_{f\,\varepsilon}))$ shown in Figure 8. From this matrix we construct the ε-connectivity-graph $G_{C\varepsilon}(P^0{}_{f1})$ also shown in Figure 8.

Finally, having obtained the ε-connectivity-graph, $G_{C\varepsilon}(P^k{}_{f\,\varepsilon})$, we can generate an extended ε-connectivity-graph, $G_{EC\varepsilon}(P^k{}_{f\,\varepsilon}) = (V, E)$ where $E = E^1 \cup E^2$. A pair of nodes (v_i, v_j) in $G_{EC\varepsilon}(P^k{}_{f\,\varepsilon})$ can be connected by either an edge in E^1 or an edge in E^2 (but not both).

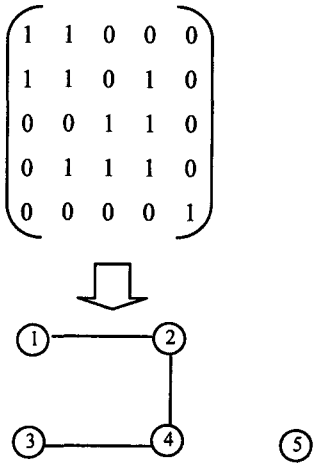

Figure 8. The ε-connectivity-matrix $M_{C\varepsilon}(G_{C\varepsilon}(P^k{}_{f\,\varepsilon}))$ and its corresponding ε-connectivity-graph $G_{C\varepsilon}(P^0{}_{f1})$.

Edges in E^1 (represented in the graph as solid edges) are edges of the type defined in graph $G_{C\varepsilon}(P^k{}_{f\,\varepsilon})$. Edges in E^2 (represented in the graph as dashed edges) are edges of the type defined in graph $G_\varepsilon(P^k{}_{f\,\varepsilon})$. Therefore a solid edge between any two nodes in an extended ε-connectivity-graph, signifies that the polygons are physically connected *and* that $|f(p_i) - f(p_j)| \le \varepsilon$. A dashed edge on the other hand signifies that $|f(p_i) - f(p_j)| \le \varepsilon$ however no direct physical connectivity exists between the two polygons.

We are now able to construct the extended ε-connectivity-graph for our example. Nodes in the graph are connected by solid edges only if the polygons are connected AND the difference of values of the function on those polygons is less than or equal to 1. They are connected by dashed edges only if they are not connected by a solid edge AND if the difference of values of the function on those polygons is less than or equal to 1. The extended ε-connectivity-graph for this example is shown in Figure 9.

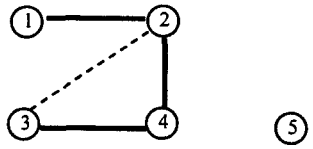

Figure 9. The extended ε-connectivity-graph $G_{EC\varepsilon}(P^k{}_{f\,\varepsilon})$.

Based on this graph, we can find covers for the particular region by finding cliques (subsets of the graph in which all nodes are connected to every other node). Cliques and their relevance to this problem are discussed in more detail in section 5. However, let us first explain the significance of having two types of edges.

If we only have edges of type E^1 in the graph, by finding a clique we find a set of polygons where each polygon is connected to every other polygon and the difference of values of the function on those polygons is less than ε.

However, it is possible (and very likely) to have a grouping of polygons where all values of the function on those polygons are within ε of one another but some of the polygons are not directly connected.

Thus we could, for example, have a case where there are three polygons, the first connected to the second and the second connected to the third. In this case, by having edges of just type E^1, we would not be able to find that full clique. However, by adding edges of type E^2, we add additional information to the graph, which allows us to find cliques of polygons even if they are not directly physically connected to each other in $\boldsymbol{P}^k_{f\,\varepsilon}$. As long as there exists a series of solid edges between the nodes in a subset of the graph, we know that those nodes are physically connected to each other in some way. We also know that the difference of function values between two polygons connected by an edge is less than ε. However this still does not mean that a group of polygons can be formed.

There can be a case where even though the difference of function values between the first polygon and the second is less than ε and the difference of function values between the second polygon and the third is less than ε, a group of polygons still cannot be formed. This is because there is no transitivity in the ε-graph, therefore the above case does not imply that the difference between the first and third polygon is also less than ε.

By introducing edges of type E^2 we can observe in the graph whether such transitivity exists. Thus if there is a graph for the above described case, and there is an edge of type E^2 between polygon one and polygon three, then we know that the difference of function values of these two polygons is also less than ε.

Therefore, if we find a clique in the extended ε-connectivity-graph which is connected by a continuous series of solid edges, then we can be certain that all the polygons in that clique are connected in some fashion and that the difference of function values between any two polygons is always less than ε.

Having generated the extended ε-connectivity-graph for our example, we can now reason about the possible level 1 covers for the region R.

Since the graph is disconnected into two parts, each part is considered separately. We can refer to these two graphs as $G^1_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$ and $G^2_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$, where $G^1_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$ is the graph with four nodes and $G^2_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$ is the graph with node 5.

Starting with the first graph, $G^1_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$, we look for cliques in the graph which are connected by a continuous series of solid edges. Nodes 2, 3 and 4 form a clique in this case, since each node is connected to every other node and there exists a continuous series of solid edges between all three nodes. Therefore, polygons 2, 3 and 4 can be considered for grouping together. Alternatively, nodes 1 and 2 form a clique as do nodes 3 and 4. Note however that nodes 1, 2, 3 and 4 do not form a clique even though there is a continuous series of solid edges. This is because node 1 is not connected to every other node in graph $G^1_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$, which means that the difference of function values between P^0_1 and P^0_3, as well as P^0_1 and P^0_4 is greater than 1.

Therefore on examination of $G^1_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$, we can state that the possible groupings of polygons are either of the following two possibilities.

$P^0_1 \rightarrow P^1_1$
$\{P^0_2 \cup P^0_3 \cup P^0_4\} \rightarrow P^1_2$ **OR**

$\{P^0_1 \cup P^0_2\} \rightarrow P^1_1$
$\{P^0_3 \cup P^0_4\} \rightarrow P^1_2$

On examination of $G^2_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$, we see that there is one possibility since there is only one node.

In this case there are two possible covers for this example, they are shown in Figure 10 below.

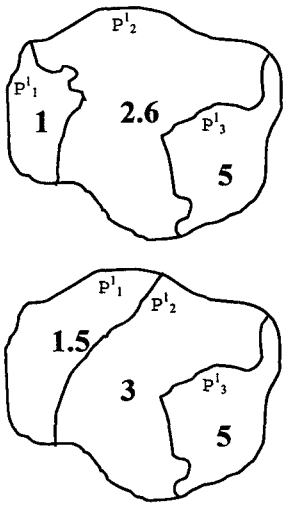

Figure 10. Two possible covers $\boldsymbol{P}^1$.

5 Graph Cliques

Definition (Garey and Johnson 1979) : Given a graph G = (V, E) and a positive integer J ≤ |V|; does G contain a ***clique*** of size J or more; i.e. a subset V' ⊆ V such that |V'| ≥ J and every two vertices in V' are joined by an edge in E? An example is shown in Figure 11 below.

Considering an extended ε-connectivity-graph $G_{EC\varepsilon}(\boldsymbol{P}^k_{f\,\varepsilon})$, we look for subsets of vertices where each vertex in the subset is connected to another by an edge (either in E^1 or in E^2). Thus we are looking for maximal subsets of vertices connected by a continuous set of edges E^1. Moreover, for each such subset, for each pair of vertices (v_i, v_j) which is not connected by en edge in E^1, there must exist an edge in E^2.

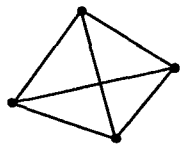

Figure 11. A four element clique

This problem is a variant of the CLIQUE problem, which is one of well known NP-complete problems in literature (Garey and Johnson 1979, Bomze, Budinich, Pardalos and Pelillo 1999, Karp 1972, Hastad 1996, Boppana and Halldorsson 1992, Bomze 1997). It is in fact one of the first problems shown to be NP-complete (Karp 1972). Since it is NP-complete, exact algorithms are guaranteed to return a solution to the problem only in a time which increases exponentially with the number of vertices in the graph (Garey and Johnson, 1979). The difference between the problem presented here and the CLIQUE problem is that of an added restriction caused by the need to check that a subset V' is connected by a continuous set of solid edges. This however only adds to the complexity of the clique problem which in itself is already a difficult problem to solve. So far the best polynomial-time approximation algorithm for the clique problem has an approximation ratio of $n^{1-o(1)}$ (Boppana and Halldorsson 1992). The problem is thus somewhat more complex than the maximum clique problem and thus still NP-complete (see proof below).

Proof.

We transform the CLIQUE $\in$ NP problem to minimum partition generation problem by using the extended ε-connectivity-graph $G_{EC\varepsilon}(\mathbf{P}^k_{f\varepsilon})$. Given a graph $G = (V, E)$ and a positive integer $J \leq |V|$, we transform E to E' where $E' = E^1 \cup E^2$. Since we need to find maximum cliques, $J = |V|$ initially. If no clique is found of size $|V|$, then $J = J - 1$ until a clique is found. Additionally, a restriction is placed on the CLIQUE problem which states that the set of vertices in the clique have to be connected by a series of continuous edges in E^1, thus adding to the complexity of the problem.

6 Conclusions

In this paper we have discussed the background and motivation for aggregation of spatial data. We have illustrated in detail the minimum partition generation process and have shown that the problem is in general NP-complete.

7 Future Work

Because the computational complexity of the maximum clique problem is hard to approximate (Bomze, Budinich, Pardalos and Pelillo 1999), there has been substantial research towards devising heuristics (Pelillo 2001). Future work on Spatial Data Aggregation will include evaluation of the maximum clique heuristics or implementation of new heuristics for the purpose of finding minimum partitions.

Additionally, in this paper we have only considered aggregations for queries on single functions. However conjunctive queries will be considered in the future, along with the two other problem formulations introduced in Section 3. Different options for error measurement will also have to be considered in the future.

8 References

BARALIS, E., PARABOSCHI, S. and TENIENTE, E. (1997): Materialized View Selection in a Multidimensional Database, *Proc. 23rd VLDB Conference*, Greece.

BAILEY-KELLOG, C., ZHAO, F. AND YIP, K. (1996): Spatial Aggregation: Language and Applications. *Proc. of the Thirteenth National Conference on Artificial Intelligence,* Portland, Oregon.

BOMZE, I., BUDINICH, M., PARDALOS, P. and PELILLO, M. (1999): The Maximum Clique Problem. In *Handbook of Combinatorial Optimization.* Vol 4. DO, D. and PARDALOS, P. (eds.). Kluwer Academic Publishers.

BOMZE, I. (1997): Evolution towards the maximum clique. *Journal Glob. Optim.,* Vol. 10:143-164.

BOPPANA, R. and HALLDORSSON, M. (1992): Approximating maximum independent sets by excluding subgraphs. *BIT,* Vol. 32: 180-196.

COLBY, L., GRIFFIN, T., LIBKIN, L., MUMICK, I., and TRICKEY, H. (1996): Algorithms for Deferred View Maintenance. *ACM SIGMOD Conference,* Montreal.

DEMERS, M. (1997): Fundamentals of geographic information systems. New York, J. Wiley and Sons.

GOLFARELLI, M., MAIO, D. and RIZZI, S. (1998): Conceptual Design of Data Warehouses from E/R Schemes. *Proc. 31st International Conference on System Sciences*, Vol. 7, Hawaii.

GOLFARELLI, M. and RIZZI, S. (1999): Designing the Data Warehouse: Key Steps and Crucial Issues. *Journal of Computer Science and Information Management, vol. 2(3).*

GONZALES, M.L. (2000): Seeking Spatial Intelligence. *Intelligent Enterprise Magazine*, 3(2).

GUPTA, A., MUMICK, I. and SUBRAHMANIAN, V.S. (1993): Maintaining Views Incrementally. *SIGMOD Conference*, pp. 157-166, Washington.

GUPTA, H. (1997): Selection of Views to Materialize in a Data Warehous., *Proc. International Conference on Database Theory*, Greece.

GUTING, R. (1994): An Introduction to Spatial Database Systems. *Special Issue on Spatial Database Systems, VLDB Journal, Vol.3, No.4 .*

HAN, J., STEFANOVIC, N. and KOPERSKI, K. (1998): Selective Materialization: An Efficient Method for Spatial Data Cube Construction. *Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining*, New York.

HASTAD, J. (1996): Clique is hard to approximate within $n^{1-\varepsilon}$. *Pro. 37th Ann. Symp. Found. Comput. Sci.:627-636.*

INDULSKA, M. (2000): Shared Result Identification for Materialized View Selection. *Proc. 11th Australasian Database Conference*, Canberra, 2000.

INMON, W.H. (1996): Building the Data Warehouse, 2nd Ed, John Wiley.

KARP, R. (1972): Reducibility among combinatorial problems. In *Complexity of Computer Computations,* MILLER, R. and THATCHER, J. (eds.). Plenum Press.

MUMICK, I., QUASS, D. and MUMICK, B. (1997): Maintenance of Data Cubes and Summary Tables in a Warehouse. *ACM SIGMOD Conference,* Arizona.

ORLOWSKA, M.E., LISTER, A.M. and FOGG, I. (1992): Decentralising Spatial Databases, in *Networking Spatial Information Systems* (revised edition). 63-76. NEWTON, P.W., ZWART, P.R. and CAVILL M.E. (eds). John Wiley and Sons.

ORLOWSKA, M.E. and ZHOU, X. (1999): A Spatial Database as a Component of Integrated Database System. International Symposium on Database Applications in Non-Traditional Environments (DANTE'99) , Japan.

PELILLO, M. (2001): Heuristics for Maximum Clique and Independent Set. In *Encuclopedia of Optimization,* FLOUDAS, and PARDALOS, P. (eds.). Kluwer Academic Publishers.

QUASS, D. and WIDOM, J. (1997): On-Line View Maintenance. *ACM SIGMOD Conference*, Arizona.

SAMET, H. (1995): Spatial Data Structures. *Modern Databse Systems: The Object Model, Interoperability and Beyond*, W. Kim, ed., Addison Wesley/ACM Press, Reading, MA.

SHEKHAR, S., CHAWLA, S. and RAVADA S. (1999): Spatial Databases: Accomplishments and Research Needs. IEEE Transactions of Knowledge and Data Engineering.

STEFANOVIC, N. (1997): Design and Implementation of On-Line Analytical Processing (OLAP) of Spatial Data. Masters Thesis, Simon Fraser University.

THEODORATOS, D., LIGOUDISTIANOS, S. and SELLIS, T. (1999): Designing the Global Data Warehouse with SPJ Views. *Proc. CAiSE*, Germany.

YANG, J., KARLAPALEM, K. and LI, Q. (1997): Algorithms for Materialized View Design in Data Warehousing Environment. *Proc. 23rd VLDB*, Greece.

ZHOU, X., TRUFFET, D. and HAN, J. (1999): Efficient Polygon Amalgamation for Spatial OLAP and Spatial Data Mining. *Proc. 6th International Symposium on Large Spatial Databases (SSD'99)*, Hong Kong, Springer-Verlag.