

Article

Towards Improving Query Performance of Web Feature Services (WFS) for Disaster Response

Chuanrong Zhang ^{1,*}, Tian Zhao ² and Weidong Li ¹

¹ Department of Geography & Center of Environmental Sciences and Engineering, University of Connecticut, Storrs, CT 06269-4148, USA; E-Mail: weidong.li@uconn.edu

² Department of Computer Science, University of Wisconsin–Milwaukee, Milwaukee, WI 53201, USA; E-Mail: tzhao@uwm.edu

* Author to whom correspondence should be addressed; E-Mail: chuanrong.zhang@uconn.edu; Tel.: +1-860-486-2610.

Received: 6 January 2013; in revised form: 21 January 2013 / Accepted: 31 January 2013 /

Published: 6 February 2013

Abstract: While OGC's WFS facilitates disseminating heterogeneous spatial data over the Web and allows feature-level geospatial information sharing and synchronization, performance issues challenge the efficient and effective utilization of WFS for disaster response. Literature shows that obtaining spatial information becomes very slow when querying WFS systems from large geospatial databases over the Internet. Solutions on how to improve the WFS system performance so that spatial data can be delivered to disaster responders within a reasonable amount of time are needed. This paper proposes a parallel approach based on Voronoi diagram indexing and data/task parallelism for improving the query performance of WFS systems for disaster applications. Experimental results show that the parallel approach can significantly improve the response time needed to process the spatial queries from a massive volume of spatial data for disaster response.

Keywords: disaster response; WFS; performance; Voronoi diagram index; parallel computation

1. Introduction

The magnitude and frequency of disasters seems to be increasing, for example the recent Wenchuan earthquake in China, the Indonesian tsunami, Hurricane Katrina, and California wildfires. Because disasters have a temporal and geographic footprint, geospatial data and tools are important for disaster response. Geospatial data and tools can help disaster responders to determine where damaged buildings or injured residents located and where impacts are greatest, so that they can act more quickly to help save lives, reduce damages and costs of dealing with disasters. The disaster response relies heavily on the ability to discover and use accurate up-to-date geospatial information to prepare for and respond to disasters. Decision makers need spatial information such as demographic data, road network data, critical facilities, infrastructure, and emergency shelter locations to coordinate response efforts during disasters. Fast obtaining the spatial information is critical to ensure that emergency supplies and resources would be able to reach the impacted areas in the most efficient manner. In fact, literature has shown that disaster response has greatly benefited from the recent advancement in information technology, especially GIS and remote sensing [1,2]. However, findings have shown that the use of GIS for disaster response can readily fail due to the inability to obtain and integrate spatial data rapidly and the lack of an interoperating GIS [3,4]. According to a critical report released by US House Select Bipartisan Committee [5], the federal government's ineffective response to Hurricane Katrina was partly a failure of sharing and accessing information in a timely manner [6]. The experience suggests that the real barriers to disaster response are not lack of data [6,7] but are in most cases the difficulties in obtaining and integrating heterogeneous spatial information in a timely manner [8]. The institutional barriers and issues preventing spatial data sharing between federal, state and, local government institutions and the private and government data providers have been discussed in the literatures [9,10]. However, even the willingness to share data does not remove the technological barriers to dynamic and effective utilization of distributed data sources for improving emergency response and homeland security efforts [11]. With the rapid development in GIS and its applications, more and more geographical databases have been developed by different programs and applications, but data sharing and acquisition is still a big problem for disaster response. Not that data are not available, there is a huge amount of geographical data stored in different places among multiple jurisdictions and in different formats, but data reuse and sharing for disaster response are daunting tasks because of the heterogeneity of existing systems in terms of data modeling concepts, data encoding techniques and storage structures, *etc.* [12].

The development of the Internet creates a unique environment for sharing geospatial data. Users can use the Internet to download data for viewing, analysis or manipulation. However, data sharing facilitated by the advances of network technologies is still hampered by the incompatibility of the variety of data models [13]. To facilitate the exchange and sharing of spatial data by building on initial expenditures, Spatial Data Infrastructures (SDIs) have been developed in many countries in the past two decades. Spatial Data Infrastructures (SDIs) provide access, reuse, and integration of heterogeneity geographic information from multiple sources over the Internet in support of disaster response [14]. To facilitate the sharing and reuse the heterogeneous spatial data, recent SDI development is based on the open standards and Open Geospatial Consortium (OGC) web service technologies [15,16].

The OGC's web services such as Web Feature Services (WFS) provide a way to achieve interoperability and a solution to solving problems arising from syntactic and structural heterogeneity between data sources. Studies have shown that OGC's WFS played important roles in real-time geospatial data sharing and exchange from heterogeneous sources over the Web for time-critical applications [15]. Another important advantage of OGC's WFS is that they provide feature-level data searches, access, and exchange in real time over the Web. OGC's WFS overcome the problem and allow disaster response applications to quickly access to the most up-to-date feature level data. While OGC's WFS facilitate disseminate heterogeneous spatial data over the Web and allow feature-level geospatial information sharing and synchronization, performance issues challenge efficient and effective utilization of WFS for the disaster response. Literature shows that obtaining spatial information becomes very slow when query the WFS systems from large geospatial databases over the Internet [16–19]. This is because the WFS transport text-based GML data over the network. When GML-coded geospatial data are transported, all the markup elements that describe spatial and non-spatial features, geometry, and spatial reference systems of the data are also transported to the recipient. This is important for data interoperability, because the GML-coded data could be saved and used by any other client-side applications that can read GML data. However, this also greatly slows down the performance for the system. Compared with some binary GIS data formats, the size of GML data files is large. Large file sizes may hinder the use of GML files as a means of data transport over the Internet. Although solutions such as using compression or sending the GML data to the client in stages or progressively have been proposed in literature [17], issues still exist for improving the performance. Solutions on how to improve the WFS system performance so that spatial data can be delivered to the disaster responders within a reasonable short time span are needed.

In addition, disaster response applications require many users concurrently access spatial databases through highly intensive geocomputation processes. The spatial data objects are generally nested and complex, and spatial queries are based not only on the attributes of spatial objects but also on the spatial location, extent and measurements of spatial objects in a reference geographical system. Therefore, spatial query requires intensive disk I/O accesses and spatial computation. This brings further challenges for efficiently and fast conducting spatial queries from the WFS systems.

With the development of SDI and GIS, the spatial data are growing exponentially year by year and they are becoming more diverse. While WFS facilitate spatial data sharing and provide feature-level data search, access, and exchange over the web thus decision makers need not download a whole data file for analysis, performance is becoming an important issue for disaster responders or decision-makers concurrent accesses to spatial information, which may be obtained by involving highly intensive computation. This has been acknowledged widely in literature. However, there are a few studies in literature to investigate the ways to improve the performance of WFS [18,19]. This paper proposes a parallel approach for improving query performance of the WFS system for disaster response applications. By improving query performance of the WFS systems, we expect that the needed spatial data can be delivered to the first disaster responders, managers and decision-makers in a timely manner.

2. A Parallel Approach for Improving Performance

To improve WFS performance and enhance disaster responders or decision-makers’ ability to make timely and accurate decisions, we propose a parallel approach to provide an efficient spatial query processing over large spatial databases. The parallel approach should allow numerous users perform concurrent queries. The proposed parallel approach makes full use of the Voronoi diagram for creating a spatial index and data/task parallelism for concurrent spatial queries. We expect that the “Voronoi diagram indexing + data/task parallelism” processing architecture reduces individual spatial query execution time by taking advantage of parallel and distributed processes thus can afford a large number of concurrent spatial queries for disaster response applications. Figure 1 illustrates the main query processing steps of the proposed parallel approach. To obtain the needed information from diverse distributed data sites, a user’s query will be first decomposed into a sequence of sub-queries. The decomposed spatial queries are then analyzed and translated into parallel programs for locating data fragments in data sites, query optimization, and query execution. After that, the parallel query results from multiple WFS servers are retrieved and combined by performing spatial calculations. Finally, the combined geospatial features are delivered as a single response to disaster responders or decision-makers. The basic idea of the proposed approach is to locate all distributed data sources that might contain answers to a user query. In a distributed system, data required for query processing need to be located since it may be present across several data sites.

Figure 1. The main query processing steps of the proposed parallel approach.

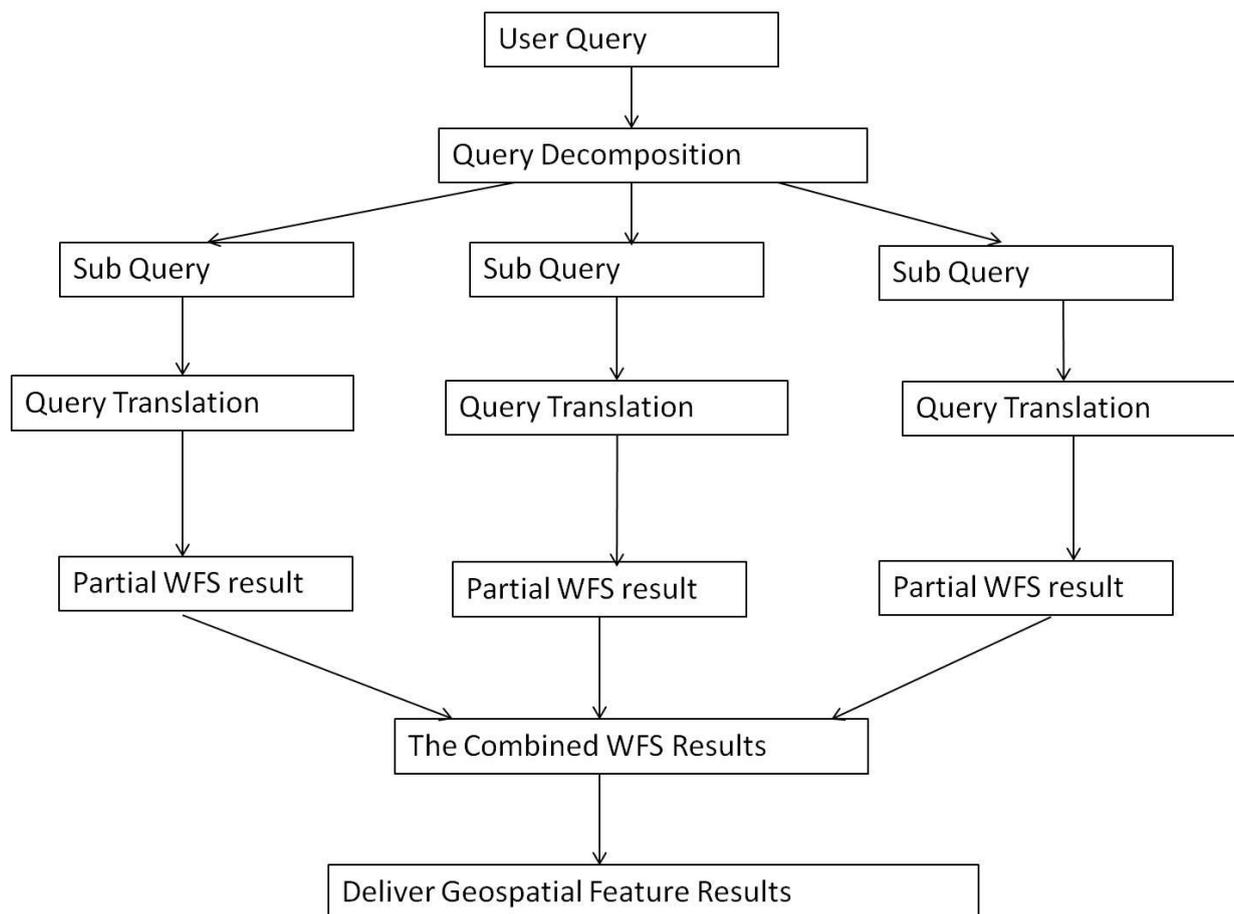
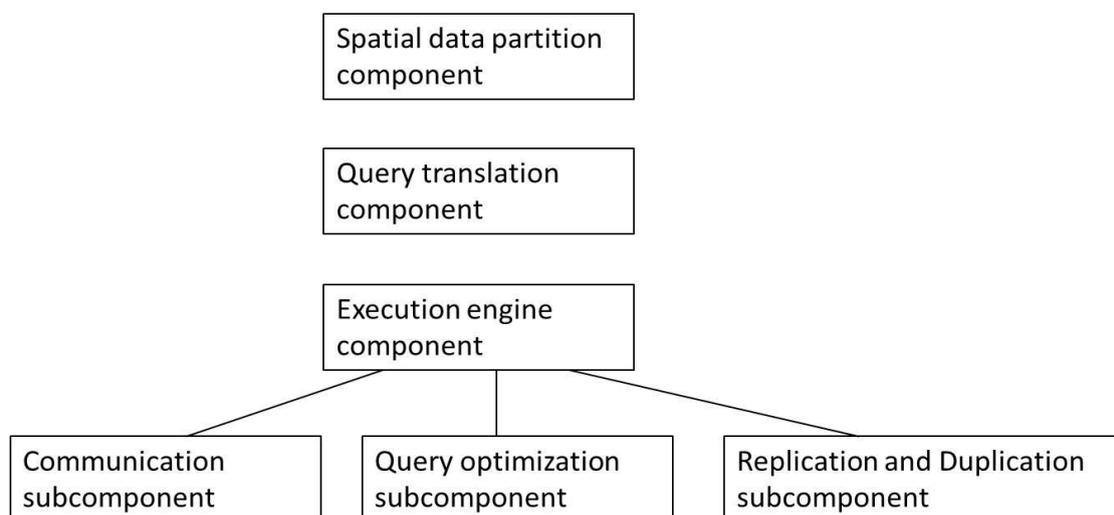


Figure 2 shows the main components of the proposed parallel approach. It includes three components—spatial data partition component, query translation component, and execution engine component. The execution engine component is composed by communication subcomponent, query optimization subcomponent, and replication and duplication subcomponent. Spatial data partition component dynamically fragments the data based on their characteristics and allocates fragments to data sites. Query translation component parses the global query and generates site specific query execution plans. Execution engine component parallelizes and executes queries on parallel WFS servers. Communication subcomponent maintains all communication requirements among data sites, for example, data delivering, collecting and combining query results. Query optimization subcomponent optimizes the site queries for efficient processing. Replication and duplication subcomponent enables parallelization in system and makes the system fault-tolerant. All these components work in complete synchronization to get the overall quest. In addition, the system also contains algorithms capable of performing spatial computations over geospatial data. In the following paragraphs, we introduce the key technologies used in the proposed parallel approach—data/task parallelism programming model with Voronoi diagram based indexing.

Figure 2. Components of the proposed parallel approach.



The proposed approach takes advantage of the inherent data and task parallelism of spatial queries by translating each user query into sub-queries that can be executed in multiple spatial data servers in parallel, of which the results are later combined. Data parallelism exists in a spatial query when the same query task is performed on partitioned data in parallel without the need to synchronize. Task parallelism exists in a spatial query when the query task is transformed into several sub-query tasks that are performed independently in parallel data servers. A spatial query that exhibits data or task parallelism can be answered more efficiently with parallel processors or distributed servers. A restricted form of data/task parallelism is supported by the MapReduce programming model [20], where independent tasks (mappers) are computed in parallel on partitioned data and the results of the mappers are aggregated by reducers into the final outcome. However, the existing MapReduce frameworks such as Hadoop have high overhead for joint queries and are more restrictive in how data are assigned to parallel tasks and how the tasks are synchronized. These limitations make them more

suitable for batch processing jobs instead of real-time spatial join queries. In this study, we propose to use a high-level programming language X10 to provide the parallel and distributed computing environment. With X10, the partitioning, synchronization, and aggregation of tasks and data are performed at language level. Thus, we are able to implement more flexible parallel query algorithms. Since X10 language provides built-in support for parallel and distributed computing, programmers are able to focus on developing concurrent programs using high level programming constructs without dealing with low level communication details of parallel and distributed processors. In X10 language, a unit of parallel computation is called a task. A program can start multiple tasks to run on a number of parallel threads using a construct called “async”. The threads are scheduled using the work-stealing algorithm to run unfinished tasks to achieve good load balancing among all processors. A program can specify a synchronization point using the “finish” construct to wait for a group of parallel tasks to complete before the next phase of computation starts. Note that parallel tasks should be chosen carefully so that they do not have dependency on shared data since explicit synchronization on shared data to prevent race conditions can significantly decrease performance. In reality, a large number of spatial queries can be parallelized without dependence on shared data. For example, consider the query Q1:

Select the K-nearest emergency shelters to each elementary school in Connecticut.

A straightforward implementation will answer the query by first retrieving all of the spatial objects for emergency shelters (denoted by the variable A) and spatial objects for elementary schools (denoted by the variable B) then performing a spatial join of A and B to find the K -nearest shelters to each school. If there are N_A number of shelters and N_B number of schools, then the time complexity would be $O(K \times N_A \times N_B)$. However, we can have better performance if we change the order of the query so that we first find all of the elementary schools and then for each school we query the K -nearest emergency shelters. Under this scheme, we can execute the sub-queries in parallel to find the K -nearest shelters for each school since the sub-queries are independent of each other. Suppose there are P number of parallel processors. Then the time complexity of the query would be $O(K \times N_A \times \frac{N_B}{P})$, which is certainly faster than the former approach. In addition, we can further improve the query performance if the emergency shelters are indexed using a Voronoi diagram, which reduces query time of K -nearest shelters to $O(\log N_A)$. That is to say, with a Voronoi diagram in place, it only takes logarithmic time to find the nearest shelter S and the next K nearest shelter can be found in constant time by searching the Voronoi neighbors of S . Thus, with a Voronoi diagram and parallelization, we can improve the computation time of query Q1 to $O(\log N_A \times \frac{N_B}{P})$. In comparison to the original query solution, this is a substantial performance improvement.

Note that there are two assumptions to this kind of performance improvement. One is that the Voronoi diagram-based indexing only helps queries that involve spatial joins of the Voronoi diagram sites such as the emergency shelters for query Q1. This is probably a reasonable assumption for emergency response applications, which tend to conduct spatial join queries based on emergency shelters or other important locations. The second assumption is that the queried data are shared by all parallel processors. This is also a reasonable assumption if the spatial data sets are moderate in sizes and each parallel processor and its corresponding storage system have a sufficiently large bandwidth to handle the required throughput. When data sets are too large to fit into main memory and spatial data have to be read from storage systems to answer queries, it would be more efficient to partition large

data sets into several segments and store them into several distributed data servers. In this scenario, we cannot divide the query tasks for locating nearest shelters for each school and send them to different servers since a single server may not contain the data to provide the answer. Instead, all servers must receive all queries so that the runtime complexity is $O(\log N_A \times N_B)$. Fortunately, with Voronoi diagram-based indices, each server can quickly discard queries, for which the server does not have related data, so that the runtime complexity is in fact close to $O\left(\log N_A \times \frac{N_B}{P}\right)$ with P servers.

To support data or task parallelization for spatial queries, spatial indices will be created for geospatial objects. A spatial index allows a data server to quickly locate the queried spatial object without searching through the entirely collection of spatial objects in the server database. The performance of parallel processing of a spatial query is highly dependent on the way that the queried data is indexed. There are different approaches to construct spatial index in literature. For example, hierarchical indices such as R-tree and Quadtree [21] have been used to enable efficient parallel processing of a wide range of spatial queries. These approaches improve runtime performance by dividing the computationally expensive pieces of the algorithm across multiple servers. The hierarchical indices allow efficient look-up of a spatial object based on its coordinates. Objects that are within spatial proximity tend to be on the same server so that it is more efficient to process spatial joins in parallel. However, these indices are not specifically designed to find out whether a point falls within a region of interests such as the K-nearest neighborhood problem, which are common queries in emergency response applications. As explained earlier, Voronoi diagram (VD) [22] allows efficient query of nearest neighbors. Thus, Voronoi diagrams are used to index the spatial objects in the proposed parallel query system.

A Voronoi diagram decomposes a space into disjoint polygons based on a set of sites (*i.e.*, point features). Given a set of sites in the Euclidean space, Voronoi diagram associates all locations in the plane to their closest site. Each site has a Voronoi polygon consisting of all points closer to it than to any other point features. The set of Voronoi polygons associated with all the sites is called the Voronoi diagram with respect to the site set. The polygons are mutually exclusive except for their boundaries. Figure 3 shows an example of a Voronoi diagram and its polygons for 11 sites. There are many approaches to generate Voronoi diagram in Euclidean space. For example, in the “divide and conquer” approach, the input point features are first sorted in increasing order by their X coordinate [23]. Next, the point features are separated into several subsets of equal size. A partial Voronoi diagram is created for each subset. Finally, the created partial Voronoi diagrams are combined to one single final Voronoi diagram for the entire input.

Figure 4 illustrates the procedure of constructing Voronoi diagram-based index. A Voronoi diagram is first computed using a major spatial object type such as emergency shelters. Then the spatial objects of another WFS feature set such as school locations are indexed using the Voronoi diagram. To index a WFS feature set, it is first divided into multiple blocks of objects, then the spatial objects of each block are indexed with the Voronoi polygon that the object belongs to. After that, the indexed objects are aggregated based on their proximity to each other. Finally, the aggregated objects are divided into partitions for several data servers.

Figure 3. An example of Voronoi diagram.

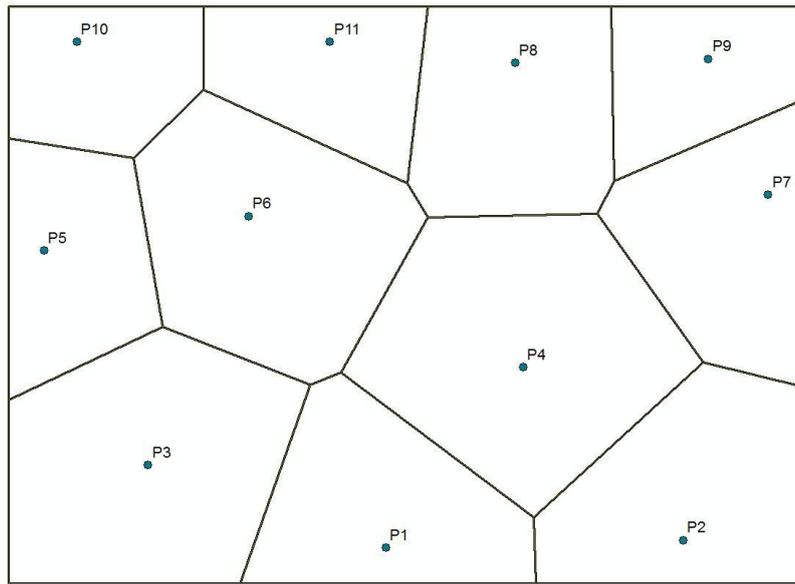
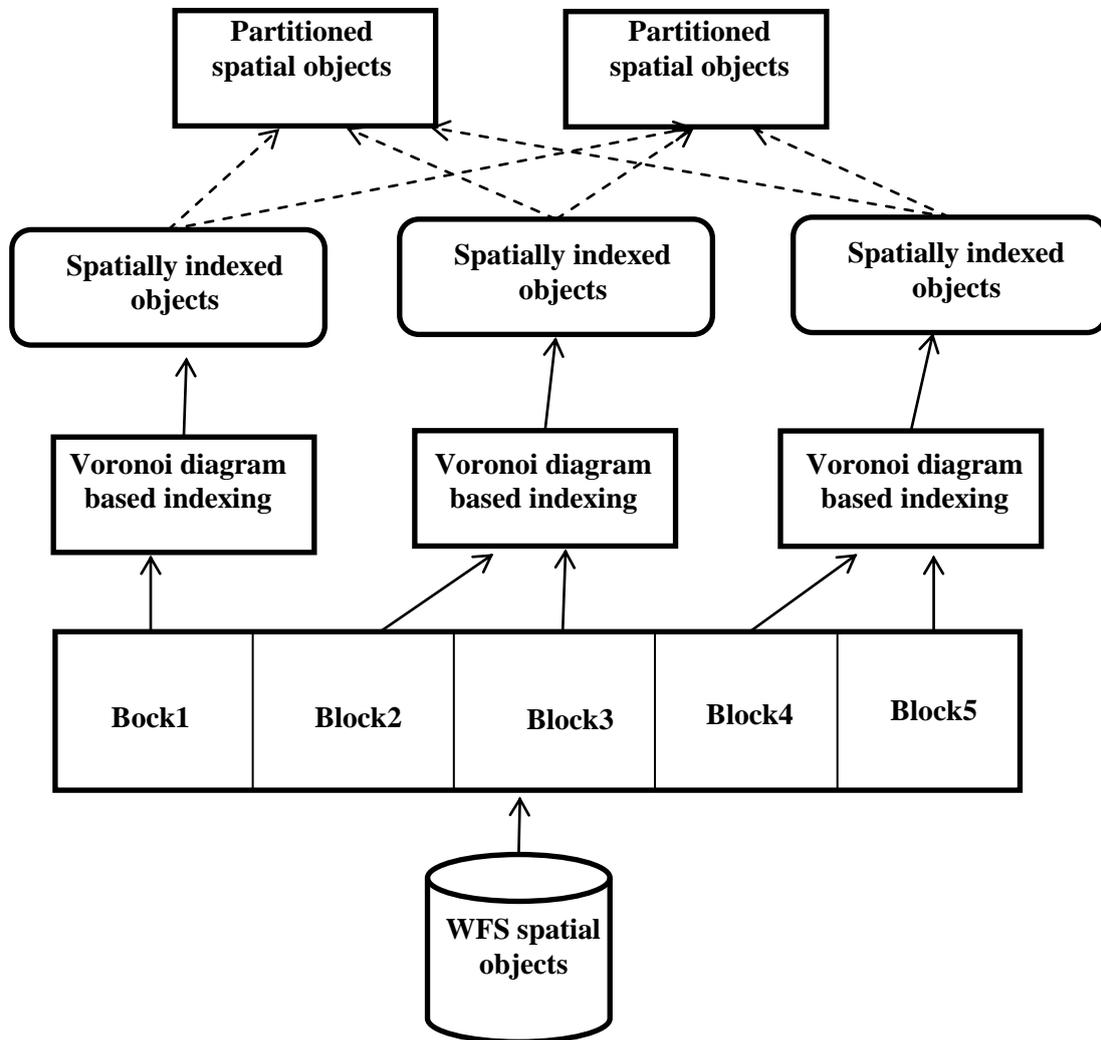


Figure 4. The procedure of indexing spatial objects using a Voronoi diagram.



3. Experiments

To evaluate the performance advantage of the parallelizing spatial query using the Voronoi diagrams, spatial query experiments were conducted on a shared memory cluster machines with 12 processors. For this experiment, 40,000 sites were randomly generated and several Voronoi diagrams were computed for the sites using the divide and conquer algorithm, which has $O(N \log N)$ runtime complexity. With P parallel processors, the runtime can be potentially lowered to $O(\frac{N}{P} \log N)$. Note that a sequential algorithm, such as SweepLine algorithm [24], may have the same runtime complexity as the Divide and Conquer algorithm does but the former cannot be easily parallelized. So in this study, we used the divide and conquer algorithm for computing the Voronoi diagram.

To allow efficient query of the nearest neighbor of a spatial point, the computed Voronoi diagram was indexed. Note that the nearest neighbor of a query point q is a site p such that the distance between q and p is less than the distance between q and any other site. Since we have the Voronoi diagram, the nearest neighbor of q is just the site of the Voronoi polygon that contains the point q . The problem of locating the polygon in a planar graph that contains a query point is also called the Point Location problem. The brute-force approach of locating the Voronoi polygon of a query point involves linear search of all of the Voronoi polygons, while an optimal search only needs $O(\log N)$ time with an existing index. In this study we implemented one of the optimal Point Location algorithms called Triangulation Refinement algorithm [25], which needs $O(N \times \log N)$ amount of space and time to compute the index. The triangulation refinement algorithm starts by adding a sufficiently large outer triangle to the Voronoi diagram to form a planar graph and then triangulates the graph so that all polygons in the graph are triangles. The second step involves picking independent vertices from the graph with degrees of each vertex no more than 8. A set of independent vertices are those that do not belong to the same faces in the graph. The third step is to remove the independent vertices from the graph and then re-triangulate the resulting graph and go back to step-two until all that remains is the single outer triangle. During the triangulation refinement steps, we recorded a hierarchical structure of triangles so that each triangle had one or more children and each the leaf triangle is part of a Voronoi polygon. To search for the Voronoi polygon that a point belongs to, we started with the root triangle and checked which child of the triangle contained the point. Then we recursively searched within that triangle until a leaf triangle was reached. The height of the hierarchy resulted from a triangulation refinement was $O(\log N)$ since each refinement step removed a constant fraction of the existing vertices of the graph. The time for locating a point using this triangle hierarchy was the height of the hierarchy, which is $O(\log N)$. Note that the triangulation refinement algorithm is much more complex than a straightforward algorithm such as the slab decomposition algorithm [26]. However, the latter algorithm needs $O(N^2)$ memory, which is not practical for large data set. For example, the index of a million site Voronoi diagram created from slab decomposition algorithm could require 10^{12} bytes.

In this study, the Voronoi diagram was indexed using the triangulation refinement algorithm. As an experiment, 10,000 KNN queries were run on Voronoi diagrams of 10,000, 20,000, and 40,000 sites using 1 to 12 shared memory processors. The queries were split up into 50 parallel tasks .where each task included 200 KNN queries. Each task was sent to a thread running on a parallel processor. The final results of parallel threads were accumulated after all parallel threads completed their computation. The query program was implemented in the programming language X10. The X10

programs can be compiled to run on either Java or C++ backend. For this experiment, Java backend was used since it has relatively better performance. Figure 5 shows the execution time of 10,000 KNN queries on three Voronoi diagrams generated from 10,000, 20,000, and 40,000 sites, respectively, using 1 to 12 parallel processors. A significant speed-up is observed with the parallel computation. From Figure 5 it can be seen that the speedup factor was almost linear when the number of processors increased from 1 to 4 and then the speedup factor decreased as more processors were added. This is due to the facts that each of the test runs included 50 parallel tasks and that when more processors were added, each processor got less workload and the overhead of the parallel architecture became more significant. Note that actual spatial queries have much higher workload than a simple KNN query does and performance gain of parallelizing spatial queries are expected to be higher in practice.

Figure 5. The execution time of 10,000 KNN queries on three Voronoi diagrams generated from 10,000, 20,000, and 40,000 sites, respectively, using 1 to 12 parallel processors.

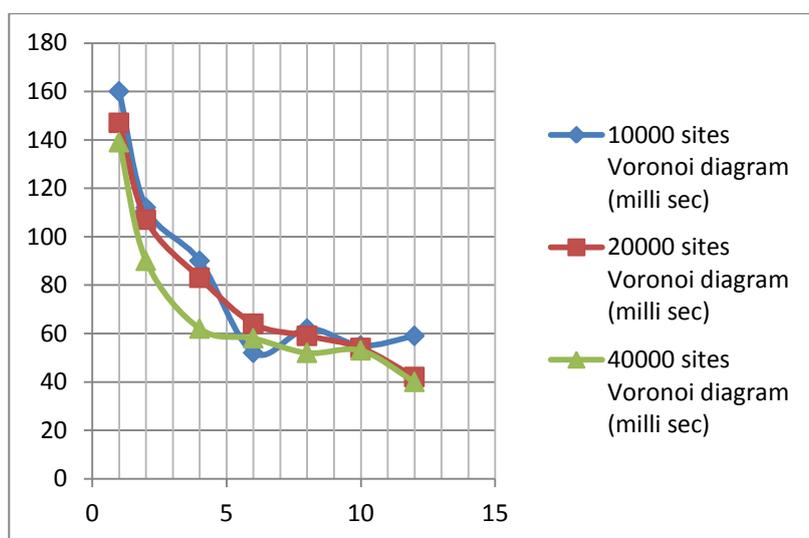
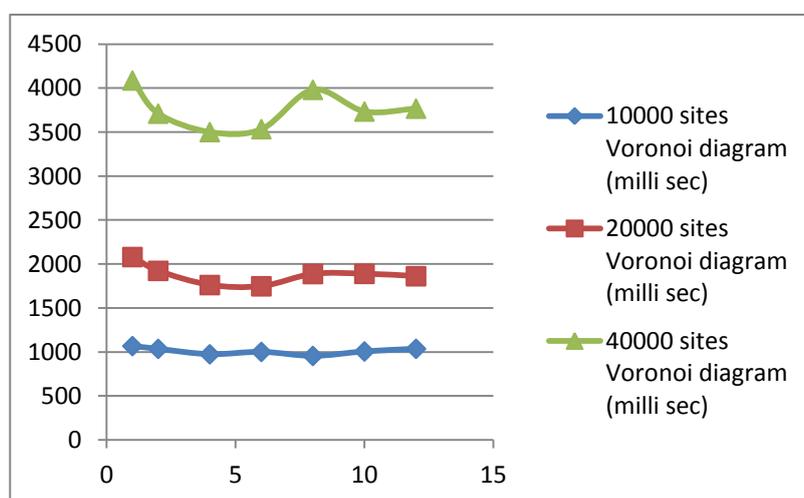


Figure 6. The execution time of Triangulation refinement of three Voronoi diagrams generated from 10,000, 20,000, and 40,000 sites, respectively, using 1 to 12 parallel processors.



We also parallelized a portion of the triangulation refinement algorithm for indexing Voronoi diagrams. Figure 6 shows the execution time of the triangulation refinement of three Voronoi diagrams generated from 10,000, 20,000, and 40,000 sites, respectively, using 1 to 12 parallel processors. However, since the indexing algorithm has a recursive structure and only a portion of the computation in each recursive step is parallelized, the speed improvement is rather limited. Also note that the runtime of indexing is proportional to the size of the Voronoi diagram.

4. Discussions

The disaster response often requires access to multiple distributed databases and large data sets for data processing, simulation, and decision-making. Obtaining and integrating heterogeneous spatial information in a timely manner is important for the efficient and effective disaster response. While OGC WFS facilitate spatial data sharing at the feature level for disaster response, the text-based GML data and concurrent and complex queries may take prohibitively long time to obtain the needed information from the WFS systems. Because a spatial query is a query that returns features based on their spatial relationship with query geometry such as searching for features given an extent or searching for features based on their relationship to other features, it is a complex service request and needs intensive geocomputation processes. This may cause a long delay in query processing or in communication without high-performance or parallel computing.

In addition, after disaster events communication is often limited because the existing infrastructure was destroyed or disasters occurred in an area without infrastructure. Thus, the disaster response may only have wireless communications including wireless ad hoc networks that can be formed among the disaster responders or relief workers carrying handheld devices such as mobile phones or laptops. Note that with the development of wireless networking and personal digital devices, it is expected that more and more wireless communication will be used for disaster response applications. However, the wireless communication has bandwidth constraints and limited communication ranges. This causes tremendous amount of traffic and service requests, thus causing more inefficient spatial queries of the WFS to get the needed information.

To improve the query performance of OGC WFS, in this research we propose a parallel approach, which makes full use of the Voronoi diagram for creating a spatial index and data/task parallelism for concurrent spatial queries, to execute the query processes on a large cluster of computers. The proposed approach is a geography aware method to partition a large map region. It splits and indexes the input data by constructing a Voronoi diagram. A Voronoi diagram is designed to guarantee that data within a partitioned region are stored on one node and all spatial data are distributed across clusters according to geographical space. A Voronoi diagram is extremely efficient in searching a nearest neighbor region because it divides the two dimensional geo-space into several parts, and each part records the nearest relationship through the shared edge between each pair of neighbor parts.

The proposed approach allows the disaster response applications to utilize multiple computational resources including high-performance computers or clusters of workstations, which may be distributed over a wide-area network, to extract the needed information from bulk geographic data sets efficiently. It, therefore, provides an ideal and practical solution to improve OGC WFS's query performance for disaster response applications. The proposed approach may be used to provide some degree of fault

tolerance. The proposed approach allows WFS systems divide running queries into sub-queries and replicates spatial data such that each sub-query can be rerun on a different server if the server initially responsible fails. This is a common situation during disaster events, when parts of infrastructure in the region may be destroyed by the disasters. Under this situation, if data were replicated on other servers that are not affected by disasters, then failed WFS queries can be rerun on those functional servers.

Experimental results show that the parallel approach can significantly improve the response time needed to process the spatial queries from a massive volume of spatial data for disaster response.

The parallel approach can also be combined with other strategies to further improve the query performance of WFS for disaster response. For example, with the growth and widespread use of smart phones, the future disaster response may use smart phones to query the needed spatial feature information from the WFS systems for updating and sharing the disaster information. With the *wireless* network for smart phones, a zipped binary XML (BXML) or the popular GZIP-based compression algorithm may be employed to decrease the data volume in the network transmission to further improve the efficiency of WFS performance.

The proposed parallel approach can also combine with the caching strategy to further improve the efficiency of query performance from the WFS systems for disaster response applications. The frequently queried results can be cached on the WFS servers to allow the systems to respond as quickly as possible. Caching reduces the data access time by storing results of frequent spatial joins in memory so that the same operation is not repeated for every data request.

In addition, the proposed parallel approach can also be combined with the progressive transmission technique to improve the efficiency of query performance from the WFS systems for the disaster response applications. With the progressive transmission technique, the spatial features in the WFS systems are extracted using cartographic principles to construct the multi-layer structure. Several methods have been proposed in literature for the vector data progressive transmission [27,28]. These methods can be used with the proposed parallel approach to further improve the efficiency of query performance from the WFS systems for disaster response applications.

Further, the proposed parallel approach can be easily adapted to the Map-Reduce model utilized by Cloud computing services. A Map-Reduce framework can execute many map-tasks in parallel and run reduce-tasks to integrate results of the map-tasks. For our approach, the parallel sub-queries generated from a WFS query can be converted to map-tasks while the partial results of the map-tasks can be summarized by a reduce-task. Even though the Map-reduce model of Cloud computing introduces large runtime overhead, it can provide distributed computing capability in elastic and on demand manners by virtualizing and pooling computing resources [29]. By providing “computing as a service” for end users in a “pay-as-you-go” mode, cloud computing may be more convenient and budget and energy consumption efficient for improve the performance of the WFS systems of heavy workload.

Finally, as mentioned in [16], the WFS systems provide a solution for real-time geospatial data sharing at the feature level. This study focus on improving query performance from the WFS system while other operations supported by the WFS-T protocol such as creating, deleting, and updating are not considered here since they are often not performance critical. In addition, to preserve the data integrity, they have to be processed sequentially, which makes it difficult to achieve performance gain in parallel framework. Further, although this paper focuses on improving the efficiency of query performance from the WFS systems, it should be pointed out that the proposed parallel approach can

also be used for discrete geospatial features such as online datasets provided by EPA and US Census Bureau [19]. As long as there is geographic location information in the data, we can use Voronoi diagrams to organize the spatial data and partition the spatial data into several subsets for parallel computation. The proposed approach is not suitable for raster data such as the map images returned by Web Map Services.

5. Conclusions

The disaster response applications need fast access to accurate and up-to-date spatial data to obtain an overview of the disaster situation, to assess the damages, and to supply local logistic teams with reliable information. No single government or private agency alone can provide and guarantee all of the needed geospatial information for effective disaster response. There is a need for searching the needed spatial information from multiple distributed data sources over the Web.

OGC's WFS facilitate integration and sharing geospatial information from multiple heterogeneous data sources and allow fast access to feature-level geospatial information over the Web. However, several inherent mechanisms of WFS such as text-based GML data and complex spatial queries through intensive geocomputation processes bring challenges for efficiently conducting spatial queries from the WFS systems. This paper proposes a parallel approach for improving query performance of the WFS systems for disaster response applications based on combination of the Voronoi diagram indexing and the parallel technique. The experiment results show that the proposed parallel approach can be used to reduce answer time for computationally intensive spatial queries from a WFS system, particularly for the spatial queries involving in gathering and processing the large amounts of spatial data from many different sources. The proposed parallel approach improves WFS query performance by making full use of high-performance computer servers distributed over a wide-area network. With unique merits of job parallelism, the proposed parallel approach may provide an effective solution to the big data analysis challenge in the disaster response applications.

In addition, the proposed parallel approach may allow many users such as disaster responders and decision-makers concurrently access spatial databases during disaster response processes. Although there is still a need for combining a variety of other strategies such as compression and cache methods with the proposed parallel approach to further improve WFS's query performance, we anticipate that the proposed approach can improve the efficiency of a WFS query by parallelizing intensive geocomputation. The proposed approach is capable of making full use of the increasingly wide available high performance computers and clusters for effective disaster response.

Acknowledgments

We thank the anonymous reviewers and the editor for their comments and suggestions on the manuscript.

References

1. Cova, T.J. GIS in Emergency Management. In *Geographical Information Systems: Principles, Techniques, Applications, and Management*; Longley, P.A., Goodchild, M.F., Maguire, D.J., Rhind, D.W., Eds.; John Wiley & Sons: New York, NY, USA, 1999; pp. 845–858.
2. Kwan, M.P.; Lee, J. Emergency response after 9/11: The potential of real-time 3D GIS for quick emergency response in micro-spatial environments. *Comput. Environ. Urban Syst.* **2005**, *29*, 93–113.
3. Williamson, R.A.; Baker, J.C. Lending a helping hand: Using remote sensing to support the response and recovery operations at the World Trade Center. *Photogramm. Eng. Remote Sens.* **2002**, *68*, 870–875.
4. Zerger, A.; Smith, D.I. Impediments to using GIS for real-time disaster decision support. *Comput. Environ. Urban Syst.* **2003**, *27*, 123–141.
5. US House Select Bipartisan Committee to Investigate the Preparation for and Response to Hurricanes Katrina and Rita. *A Failure of Initiative: The Final Report of the Select Bipartisan Committee to Investigate the Preparation for and Response to Hurricanes Katrina and Rita*; US Government Printing Office: Washington, DC, USA, 2006.
6. Donkervoort, S.; Dolan, S.M.; Beckwith, M.; Northrup, T.P.; Sozer, A. Enhancing accurate data collection in mass fatality kinship identifications: Lessons learned from Hurricane Katrina. *Forensic Sci. Int. Genet.* **2008**, *2*, 354–362.
7. Monmonier, M.; Giordano, A. GIS in New York State County Emergency Management Offices. *Appl. Geogr. Stud.* **1998**, *2*, 95–109.
8. Wiegand, N.; Garc ía, C. A task-based ontology approach to automate geospatial data retrieval. *Trans. GIS* **2007**, *11*, 355–376.
9. Harvey, F. Developing geographic information infrastructure for local government: The role of trust. *Can. Geogr.—Geogr. Can.* **2003**, *47*, 28–36.
10. Niemann, B.J.; Niemann, S. Allan H. Schmidt: GIS journeyman. *Geo. Inf. Syst.* **1998**, *8*, 42–45.
11. Grady, R.K. Homeland security, privacy, and data acquisition. *Earth Obs. Mag.* **2004**, *13*, 27–28.
12. Devogele, T.; Parent, C.; Spaccapietra, S. On spatial database integration. *Int. J. Geogr. Info. Sci.* **1998**, *12*, 335–352.
13. Choicki, J. Constraint-based interoperability of spatiotemporal databases. *Geoinformatica* **1999**, *3*, 211–243.
14. Mansourian, A.; Rajabifard, A.; Valadan Zoej, M.J.; Williamson, I.P. Using SDI and web-based system to facilitate disaster management. *Comput. Geosci.* **2006**, *32*, 303–315.
15. Zhang, C.; Li, W.; Peng, Z.-R.; Day, M. GML-based interoperable geographical databases. *Cartography* **2003**, *32*, 1–16.
16. Zhang, C.; Li, W. The roles of web feature service and web map service in real time geospatial data sharing for time-critical applications. *Cartogr. Geogr. Inform. Sci.* **2005**, *32*, 269–283.
17. Peng, Z.-R.; Zhang, C. The roles of geography markup language, scalable vector graphics, and web feature service specifications in the development of internet geographic information systems. *J. Geogr. Syst.* **2004**, *6*, 95–116.
18. Yang, C.; Wong, D.; Yang, R.; Kafatos, M.; Li, Q. Performance-improving techniques in web-based GIS. *Int. J. Geogr. Info. Sci.* **2005**, *19*, 319–342.

19. Yang C.; Wu, H.; Huang, Q.; Li, Z.; Li, J.; Li, W.; Miao, L.; Sun, M. WebGIS Performance Issues and Solutions. In *Advances in Web-based GIS, Mapping Services and Applications*; Li, S., Dragicevic, S., Veenendaal, B., Eds.; Taylor and Francis: New York, NY, USA, 2011; pp.121–138.
20. Dean, J.; Ghemawat, S. Mapreduce: Simplified data processing on large clusters. *Commun. ACM* **2008**, *51*, 107–113.
21. Cary, A.; Sun, Z.; Hristidis, V.; Rische, N. Experiences on Processing Spatial Data with MapReduce. In *Proceedings of the 21st international Conference on Scientific and Statistical Database Management*, New Orleans, LA, USA, 2–4 June 2009; Winslett, M., Ed.; pp. 302–319.
22. Akdogan, A.; Demiryurek, U.; Banaei-Kashani, F.; Shahabi, C. Voronoi-Based Geospatial Query Processing with MapReduce. In *Proceedings of 2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom)*, Indianapolis, IN, USA, 30 November–3 December 2010; pp. 9–16; Available online: http://infolab.usc.edu/DocsDemos/IEEE_Cloud_Computing_2010.pdf (accessed on 23 December 2012).
23. Okabe, A.; Boots, B.; Sugihara, K.; Chiu, S.N. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*; Wiley: New York, NY, USA, 2000.
24. Fortune, S. A Sweepline Algorithm for Voronoi Diagrams. In *Proceedings of the Second Annual Symposium on Computational Geometry*, Yorktown Heights, NY, USA, 2–4 June 1986; pp.313–322.
25. Kirkpatrick, D.G. Optimal search in planar subdivisions. *SIAM J. Comput.* **1983**, *12*, 28–35.
26. Dobkin, D.; Lipton, R.J. Multidimensional searching problems. *SIAM J. Comput.* **1976**, *5*, 181–186.
27. Battenfield, B.P. Transmitting vector geospatial data across the internet. *Geogr. Inform. Sci.* **2002**, *2478*, 51–64.
28. Hoppe, H. Progressive Meshes. In *Proceedings of SIGGRAPH'96*, New Orleans, LA, USA, 4–9 August 1996; pp. 99–108.
29. Yang, C.; Goodchild, M.; Huang, Q.; Nebert, D.; Raskin, R.; Bambacus, M.; Xu, Y.; Fay, D. Spatial Cloud Computing—How can geospatial sciences use and help to shape cloud computing. *Int. J. Digital Earth* **2011**, *4*, 305–329.