ABSTRACT

Analyzing and querying large volumes of spatially derived data from scientific experiments has posed major challenges in the past decade. For example, the systematic analysis of imaged pathology specimens result in rich spatially derived information with "GIS" characteristics at cellular and sub-cellular scales, with nearly a million derived markups and hundred million features per image. This provides critical information for evaluation of experimental results, support of biomedical studies and pathology image based diagnosis. However, the vast amount of spatially oriented morphological information poses major challenges for analytical medical imaging. The major challenges I attack include: i) How can we provide cost effective, scalable spatial query support for medical imaging GIS? ii) How can we provide fast response queries on analytical imaging data to support biomedical research and clinical diagnosis? and iii) How can we provide expressive queries to support spatial queries and spatial pattern discoveries for end users? In my thesis, I work towards developing a MapReduce based framework MIGIS to support expressive, cost effective and high performance spatial queries. The framework includes a real-time spatial query engine RESQUE consisting of a variety of optimized access methods, boundary and density aware spatial data partitioning, a declarative query language interface, a query translator which automates translation of the spatial queries into MapReduce programs and an execution engine which parallelizes and executes queries on Hadoop. Our preliminary experiments demonstrate that MIGIS is a cost effective architecture which achieves high performance spatial query execution. MIGIS is extensible and can be adapted to support similar complex spatial queries for large scale spatial data in other scientific domains.

Categories and Subject Descriptors
H.2.8 [Database Management]: Database Applications—Spatial Database and GIS, Scientific databases

General Terms
Design, Experimentation, Performance

Keywords
Spatial Query Processing, MapReduce, Scientific Data Management, Analytical Imaging

1. INTRODUCTION

1.1 Digital Pathology & Medical Imaging “GIS”

Analyzing spatially oriented scientific data becoming increasingly important for many scientific applications such as satellite imagery, sky survey [9], human atlas [13] and oil exploration etc. In the past decade, digitalized pathology images generated by high resolution scanners enabled microscopic examination of tissue specimens to support clinical diagnosis and biomedical research [6]. Pathology image analysis plays a critical role to support clinical diagnosis and biomedical research by providing quantitative and reproducible measurements of micro-anatomical features. For example, one common approach for computer aided diagnosis with pathology imaging is to segment boundaries of cellular and sub-cellular objects such as tumors, blood vessels, cells, nuclei and compute dozens of image features associated with the image regions of these objects. This generates rich spatially derived information with GIS characteristics – Medical Imaging “GIS” [15, 2].

1.2 Use Cases and Complex Queries

Algorithm Evaluation and Spatial Cross-Matching.

To support computer aided diagnosis of diseases with the emerging pathology imaging technology, it is essential to develop and evaluate high quality image analysis algorithms in following scenarios. i) Algorithm Validation. Algorithms are tested, evaluated and improved in an iterative manner. This involves a formal testing phase in which ground-truth segmentation results (labeled by pathologists) are compared to algorithm generated output. The results are evaluated to assess inter-observer variability between pathologists, between algorithms and humans, and between different algorithms. ii) Algorithm Consolidation. Multiple algorithms are developed in a study to solve the same problem. One algorithm may outperform another in certain regions of the feature space, given that they optimize different cost function. Studies showed that consolidating results from multiple algorithms (i.e., an ensemble approach) can lead to better analysis results. iii) Algorithm Sensitivity Study. An algorithm often involves a set of parameters that can be tuned to adapt to different image types, resolution and image quality. Exploring the sensitivity of analysis output with respect to parameter adjustments can provide a guideline for the best deployment of algorithms in different scenarios and for rapid development of robust algorithms. One essential task for such evaluation is
to cross-match millions of spatial boundaries of segmented micro-
anatomic objects such as nuclei for a single image, with a study of
hundreds of images. To evaluate the likeness of two or multiple
result sets, we can compute the ratio of the total area of the inter-
section divided by the total area of the union between two polygon
sets.

Spatial Proximity between Micro-anatomic Objects and Local Nearest Neighbor Density.
Micro-anatomic objects with spatial proximity often form groups
of cells that are close in both physical space and gene expression
space. For example, a shortest distance query will provide sum-
mary statistics of the proximity of stem cells to these regions of
interest: for each stem cell, find abnormal regions that are closest
to the cell and compute the mean distance and standard deviation
for all stem cells in an image. Another example query is to iden-
tify $K$ nearest nuclei for each blood vessel and return local density
measure. This is a type of query where every object is involved and
we could have millions of cells for a single image.

Global Spatial Pattern Discovery in Images.
The tumor growth comes with necrosis and vascular proliferation
which often forms spatial patterns during different stage of tumor
growth. For example, glioblastoma, the most common brain tumor,
often appears as ring-enhancing lesions where the rings have much
higher concentration of cells than adjacent regions (Figure 1 (c)).
Identification of regional co-location patterns and detection of spa-
tial outliers help automate the identification of tumor subtypes and
their characteristics in a more qualitative manner.

1.3 Challenges

High Spatial and Geometric Computation Complexity.
Cross-matching spatial objects is a typical spatial join which first
finds matching intersected polygon pairs and then measures the ra-
tio of overlapped areas. A naive brute force approach for such
matching is extremely expensive and may take hours or days to
compute even for a single image [17]. This is mainly due to the
polynomial complexity of common computational geometry algo-
rithms [1] used for verifying intersections of polygon pairs which
most often contains hundreds of points to represent each shape.
While spatial access methods provide efficient matching of poly-
gons based on their Minimum Bounding Rectangle (MBR), the
expensive intersection verification and spatial measurement is often
ignored in the previous studies. Moreover, statistical measures
such as mean-nearest neighbor distance and spatial autocorrelation
are computationally expensive and they are used frequently in anal-
ysis tasks. These bring new challenges for the algorithm evaluation
and analysis tasks.

“Big Data” Challenge.
Pathology images such as whole-slide images (WSI) made by
scanning microscope slides at diagnostic resolution are very large:
a typical WSI contains 100,000x100,000 pixels. One image may
contain millions of objects such as cells or nuclei. Hundreds of
image features for each object could be extracted. A study may in-
volve hundreds of images obtained from a large cohort of subjects.
For large scale interrelated analysis, there may be dozens of algo-
rithms – with varying parameters – to generate many different re-
sult sets to be compared and consolidated. Thus, derived data from
images of a single study is often in the scale of tens of Terabytes,
and Petabytes of data is likely when analytical pathology imaging
is adopted in the clinical environment in the future. Such large vol-
ume of data combined with the complexity of spatial queries poses
new challenges for developing effective solutions.

2. REQUIREMENTS

2.1 Architecture Requirements
Real-Time Spatial Queries with Fast Response. Algorithm eval-
uation often requires on the fly evaluation of temporary results and
such results maybe discarded after the query results are obtained.
Initial analysis results and algorithm output are generated quickly.
Therefore fast response time is needed and indeed critical for the
efficiency of the whole analysis pipeline.

Cost-Effectiveness. While it’s possible to have a multi-million dol-
lar parallel SDBMS package with reasonable performance for some
use cases, it’s unrealistic to require a research environment to have
such an expensive up-front budget. Therefore a cost-effective ar-
chitecture is required.

Extensible Framework. Query needs may change over time and
new queries can be complex. For some query tasks, users may
want to use complex algorithms which they developed specifically
for that task. Thus, flexible and extensible architecture is needed
to support new queries and UDFs.

Declarative Query Language. Most of the users (biomedical re-
searchers and doctors) have very limited programming skills if they
have at all. Historically, they are more comfortable with simple
query languages such as SQL. Thus a declarative query language
support is essential for usability.

Scalable HPC architecture. For analytical tasks such as result
analysis, mining, comparison and evaluation a scalable high per-
formance computing architecture is needed.

2.2 Query Requirements
Pathology analytical results are mostly spatially oriented, and
most queries indeed involves certain types of spatial operations.
Some of the query cases are illustrated in Figure 1.

Spatial Join Query: To compare and consolidate (segmenta-
tion) algorithm results, two result sets will be cross-matched to
calculate the extent of intersections. For example, a query to com-
pute the distance and intersection ratio of intersected spatial objects
segmented from an image by different algorithms (green vs. red
in Figure 1 (a)) is a common but most expensive query type. i.e.
$\sigma_{\text{dist,overlap}}(Alg_1 \bowtie \text{Intersects} Alg_2)$.

Multi-way Spatial Join Query: Multiple result sets will be cross-
matched to compute the extent of intersections. A natural extension
of two-way join to multiple algorithm results. $Alg_1 \bowtie Alg_2 \bowtie \ldots \bowtie Alg_n$.

Nearest Neighbor Query: To characterize the nuclei/cell features
around certain types of tissues. For example, find all cells (Figure 1
(b) green) close to blood vessels (Figure 1 (b) red) and compare
their characteristics with the other cells.

Global Density Pattern Query: Similar to window query but with
extra predicate. That is the regions selected should satisfy local
density conditions specified in the query. For example, finding all
regions with size at least $c$ and density is higher than $d$.

2.3 Current Available Solutions
The problem is studied in the context of spatial databases. In this
approach [16], spatial boundaries are managed as spatial data types
through spatially extended data types on top of object-relational
databases, and spatially extended indexing is provided for finding
matching spatial objects based on MBRs. Spatial extended func-
tions are then used in SQL queries to provide spatial operator sup-
port. While this approach achieves certain level of scalability, it’s not practical due to the long overall response time, the complexity of the architecture and the expensive cost of both software and hardware license.

3. OUR APPROACH: MIGIS

We are developing a MapReduce based framework MIGIS to support expressive and cost effective, high performance spatial queries. The framework includes a real-time spatial query engine RESQUE consisting of a variety of optimized access methods, boundary and density aware spatial data partitioning, a declarative query language interface, a query translator which automates translation of the spatial queries into MapReduce programs, and an execution engine which parallelizes and executes queries on open source MapReduce platform – Hadoop. Figure 2 shows an architectural overview of MIGIS. It relies on the HDFS for data storage, Hadoop for real-time query execution and utilizes a stand-alone spatial query engine for spatial query processing.

RESQUE: Real-Time Spatial Query Engine.

To support high performance spatial queries, a stand-alone spatial query engine is needed and it should poses following capabilities: i) spatial relationship comparison, such as intersects, touches, overlaps, contains, within, disjoint; ii) spatial measurements, such as intersection, union, convexHull, difference, distance, centroid, area, etc.; iii) spatial access methods for efficient query processing; and iv) optimization for real-time processing environment. The engine should also be easily executed across multiple cluster nodes for parallelization and should have certain degree of extensibility. RESQUE is developed with such design and performance requirements in mind.

Spatial operations can be expensive to compute compared to the operations in a non-spatial relational database. For example, to compare two image set results, these images “joined” with each other with predicates such as “\(P_1\) intersects \(P_2\)” (\(P_1\) and \(P_2\) represent spatial objects in each image set). Therefore spatial query processing techniques traditionally employees a “partition-filter-refine” approach without creating index [11, 18]. In RESQUE, we take a hybrid approach in which spatial indices are created on the fly and used to assist spatial calculation on partitions. Our experiments further indicate that the overhead of index creation is only accounts for tiny fraction of overall query response time, while it significantly increases query performance.

Spatial Query Language and MapReduce Translator.

Users interact with the system using a SQL-like query language with spatial extension. This requires translation of SQL queries into Map-Reduce to execute on the system. YSmart [7], a SQL-to-MapReduce translator developed in our group, is extended with spatial query capabilities. After users submit a query, YSmart parses the query into a tree structure and replaces the spatial predicates with spatial operators. It also analyzes operator correlation between query tree nodes and optimizes query for faster execution.

Query Execution Engine.

Depending on the query types, different execution strategies are selected for optimized performance. Query Execution Engine analyzes types of queries, carries on further optimization and triggers the right engine to process queries. To evaluate the example “join” query we mentioned above, the execution engine follows following steps. First, bulk spatial index building is performed on spatial data of each corresponding tile and we use R*-Trees [4] for spatial index construction. The spatial join component performs MBR based spatial join filtering with the two R*-Trees, and refinement on the spatial join condition is further performed on the polygon pairs through geometric computations. The spatial measurement step is performed on intersected polygon pairs to calculate results required, such as centroid distance for each pair of intersecting spatial objects. We refer interested readers to [17] for details.

4. PRELIMINARY RESULTS

Experiment Setup. We use a dataset of whole slide images for brain tumor studies provided by Emory University Hospital, with two results computed from two algorithms. We have dataset of size from 1X(18 images, 44GB) to 10X(180 images, 440GB) for different testings and all the performance tests are conducted on a small dedicated cluster with 10 nodes (192 cores overall). The average number of nuclei per image is 0.5 million, and each nucleus has 74 derived features.

4.1 RESQUE

Query Performance. To test the performance of RESQUE itself, we run it on a single node as a single thread application. We run
the spatial join query with RESQUE, as it is a commonly used expensive query type. We test the spatial indexing effectiveness by taking a single tile with two result sets (5506 vs 5609 spatial objects). Figure 3 summarizes evaluation results. A brute-force approach compares all possible pairs of boundaries using a computational geometry function without any index, and takes 673 minutes. Such slow performance is due to polynomial complexity on pairwise comparisons and expensive computation on geometric computation. An optimized brute-force approach will first filters out non-intersecting polygons with MBR intersection test. This approach takes 4 minutes 41 seconds, thanks to minimized geometric computations. Using RESQUE with indexing based spatial join, the number of computations is significantly reduced, and it only takes 16 seconds.

**Data Loading.** In terms of data loading performance, MapReduce outperforms other approaches by large margin as observed in [12]. Therefore RESQUE also enjoys the light loading cost compared to SDBMS approach. As shown in [17], RESQUE dominates on the overall efficiency and makes it possible for building a fast response query system. Especially, the loading time is minimal compared to others.

### 4.2 MIGIS

Without loss of generality, we continue to use the “join query” example to demonstrate the performance of executing spatial queries in MIGIS. Figure 4 shows the results with different data sizes: 1X, 3X, 5X, and 10X (18 images) data sets, with varying number of reducers. Note that the reduce phase dominates the cost as map phase is performing simple record grouping based on keys. We can see a continuous drop of execution time when the number of reducers increases, and the time can not be further reduced after all cores on the cluster are fully utilized. It achieves a nearly linear speed-up, e.g., time is reduced to about half when the number of reducers is increased from 50 to 100. The average querying time per image is 15 seconds (with 1X dataset and all cores), comparing with 22 minutes 12 seconds in a single partition PostGIS, and 89 minutes 30 seconds on a single partition SDBMS X. The experiments also show a nearly linear increase of time versus data sizes and it testifies that MIGIS is a very scalable approach. We refer interested readers to [17] for more detailed experiments on a larger cluster.

### 5. RESEARCH DIRECTIONS

#### 5.1 Optimized Spatial Query Engines for Different Use Cases.

Decades of past research suggests that good query plans can improve query performance by orders of magnitude. Current systems such as Hive and Pig [14, 10] have limited query optimization capabilities. As we explored in [7], intra and inter query correlations can be utilized to generate more efficient query plans. However, complex query cases in medical imaging, as we illustrate next, requires a more intelligent optimizer which can pick the best plan according to query type and data placement.

**Nearest Neighbor Query.** Nearest neighbor query is an important but expensive query type. Figure 1 (b) illustrates one example where for each cell (green), nearest blood vessel (red) and distance between them need to be returned. Spatial access structure such as R*-Tree or KD-Tree, and space partitioning approaches such as Voronoi Diagram can be used to facilitate the search process. However, the number of spatial objects are simply too large which makes those methods still inefficient. Moreover, after images partitioned into tiles the spatial co-location information will be lost. That is, nearest neighbor of one object may reside in another tile and sharing information between computation nodes in a shared nothing environment is expensive. One simple approach is to solicit the neighboring tiles also participate in the calculation of nearest neighbors for the same query and filter the redundant results. But this approach requires extra batch of computation overhead.

**Multiway Spatial Join.** One straightforward way is to implement it as nested two-way join. However, naïve implementation may not be efficient and the query graph in multi-way join can be complex (clique query, chain query, star query and ring query). Meanwhile, the spatial predicates are generally expensive to compute and join reordering between participating relations may result in different query performance. How to efficiently support multi-way join is an interesting direction and we are planning to extend the system to support multiway in the next step.

**Density Query.** Characterizing a dense regions is not trivial and even trained professionals may produce false alarm or miss some important regions. Supporting a density query would also enables biomedical research professionals to characterize tumor regions more accurately in terms of spatial features.
5.2 Intelligent Data Partitioning

While all of our experiments are based on fixed tile partition size, it's interesting to see how the partitioning effects system performance. In some cases, larger partitioning is preferred (i.e. KNN). Moreover it's desirable to have a intelligent system which automatically infers optimal partition size depending on the dataset characteristics and query type. In our experiments we find that dynamically adjusted tile partitioning can help alleviate the "Curse of the Last Reducer".

5.3 Declarative Query Interface

MIGIS relies on Hadoop for spatial query execution. Thus users need to submit their job as MapReduce programs. However, it's unrealistic to request users to adjust their technical skills and write MapReduce programs instead of writing SQL which they are comfortable with. Therefore, a unified query interface is needed. There are several SQL to MapReduce translators such as Hive [14] and YSmart [7]. But they only support subset of standard SQL and none of them provides spatial query translation functionality. How to automatically translate a spatial SQL query into efficient MapReduce program and how does it compare to hand written code is another interesting problem we are planning to investigate in the future.

5.4 Hardware Accelerated RESQUE

Most of the low level algorithms which SDBMS relies on, have been well studied in computational geometry context. Under current computer architecture algorithmic advances are less likely significantly reduce the cost of geometrical comparison. GPU is capable of processing thousands of geometry comparison in real time, thanks to it's special architecture. Since we are dealing with spatial objects (polygons, lines, points) and geometry comparison on them, it's promising to extend the MIGIS framework to harness the computational power of GPU for spatial processing on medical images.

6. RELATED WORK

Scientific databases often come with spatial aspects [3], for example, Large Synoptic Survey Telescope (LSST) generates huge amount of spatially oriented sky image data. Precursor of LSST, the Sloan Digital Sky Survey project (SDSS) 1 created a high resolution multi-wavelength map of the Northern Sky with 2.5 trillion pixels of imaging, and takes a large scale parallel database approach. SDSS provides a high precision GIS system for astronomy, implemented as a set of UDFs.

Pig/MapReduce based approach has been studied in [8] for structural queries for astronomy simulation analysis tasks and compared with IDL and DBMS approaches. In [5], an approach is proposed on bulk-construction of R-Trees and aerial image quality computation through MapReduce. In [18], a spatial join algorithm on MapReduce is proposed for skewed spatial data, without using spatial indexes.

7. CONCLUSION

Analytical pathology imaging provides high potential to support biomedical research and computer aided diagnosis, and high quality analytical results can be used as a new type of electronic health-care records. In my thesis, I work on building scalable and efficient query system for supporting large scale analytical queries on pathology images with declarative query languages, to provide feasible solutions for analytical imaging based biomedical research and clinical diagnosis. The goal of this proposal is to make fundamental contributions towards realizing an expressive, cost effective, and high performance medical imaging “GIS” for managing and querying analytical medical imaging records. The research objectives include: i) customizable efficient real-time spatial query engine to support medical imaging GIS query cases; ii) effective spatial data partitioning methods; iii) declarative spatial query and pattern discovery language; iv) spatial query parallelization and optimization by combining spatial indexing, data partitioning and system optimizations on top of commodity clusters and v) intelligent query translator which can automate translation of spatial queries into parallel executable programs. Our preliminary results demonstrate the feasibility our proposed system.

8. REFERENCES
