EMORY
UNIVERSITY

High Performance Spatial Query Processing for Large scale Scientific Data

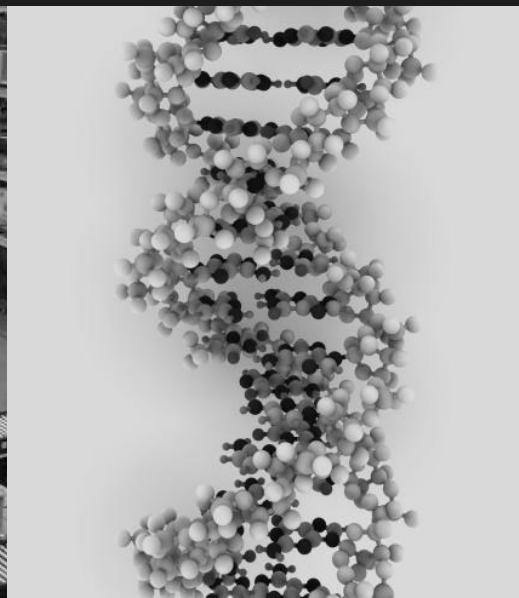
Ablimit Aji

Adviser: Fusheng Wang

Math & CS Seminar
Nov 16, 2012

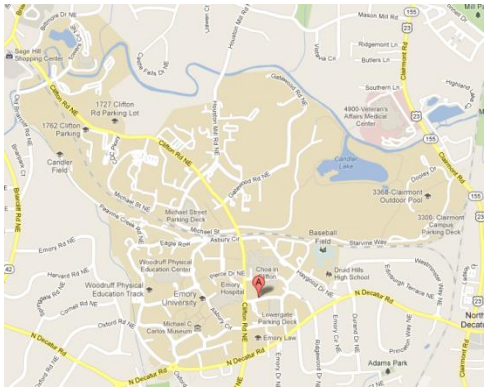
http://www

(c) 2006 Christopher Chan
flickr.com/photos/chanc/

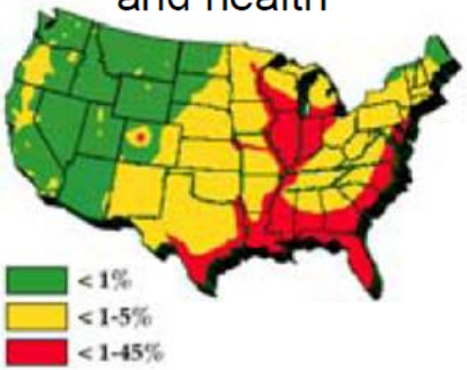


Spatial Applications

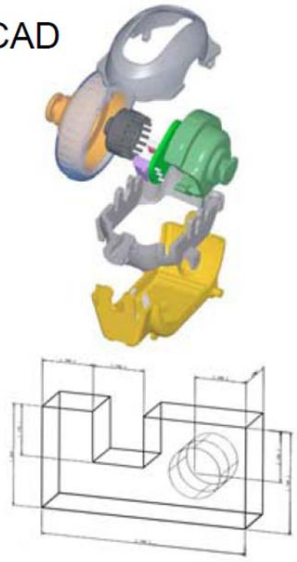
Map



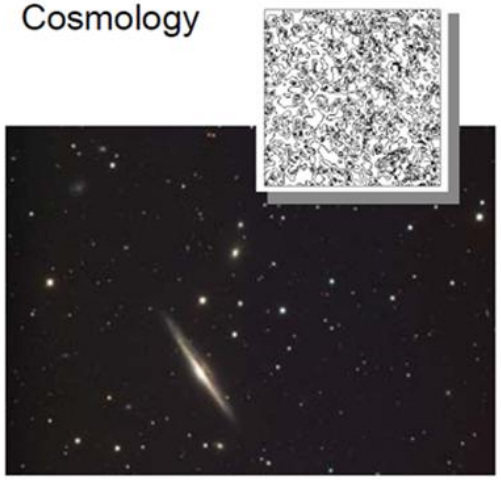
Epidemiology and health



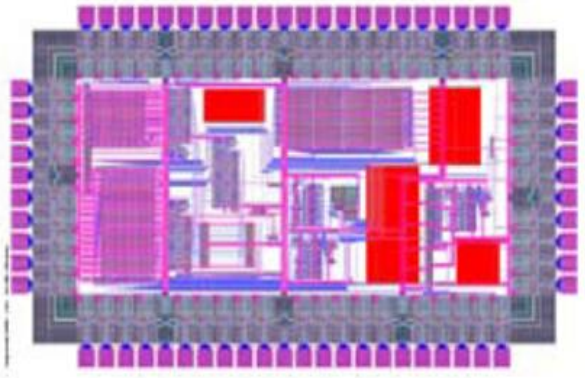
CAD



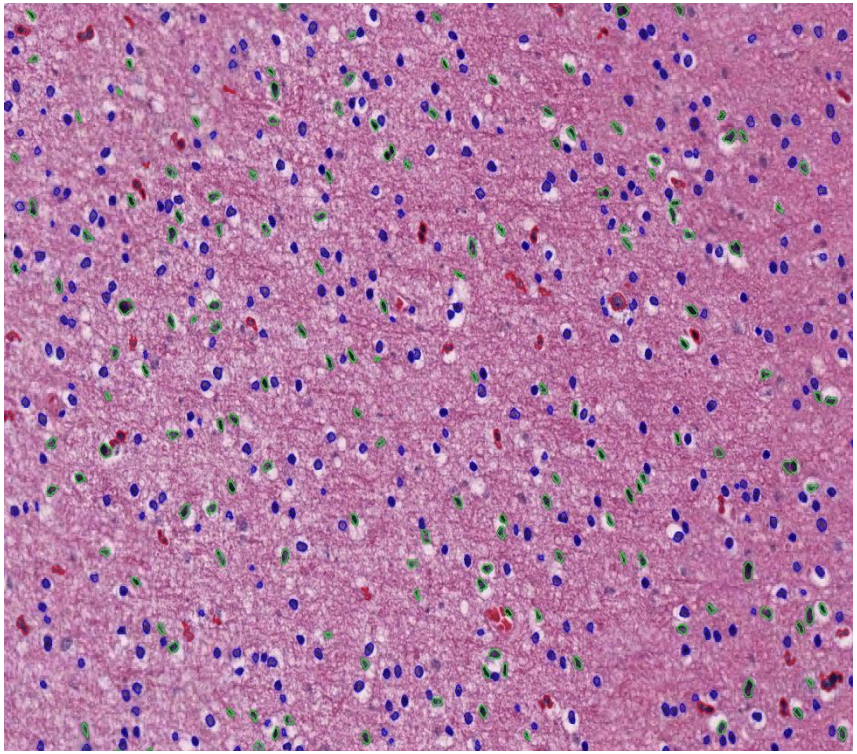
Cosmology



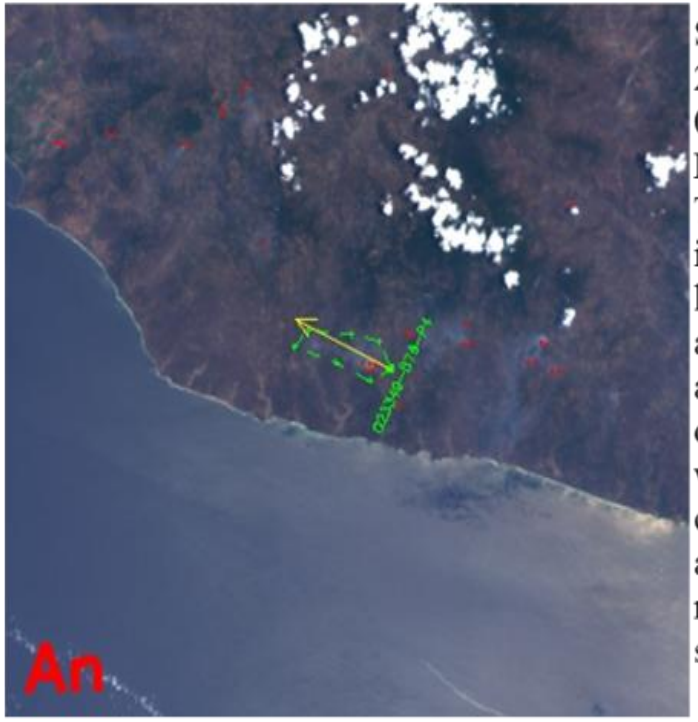
VLSI Design



Big Spatial Data



Pathology Analytical Imaging



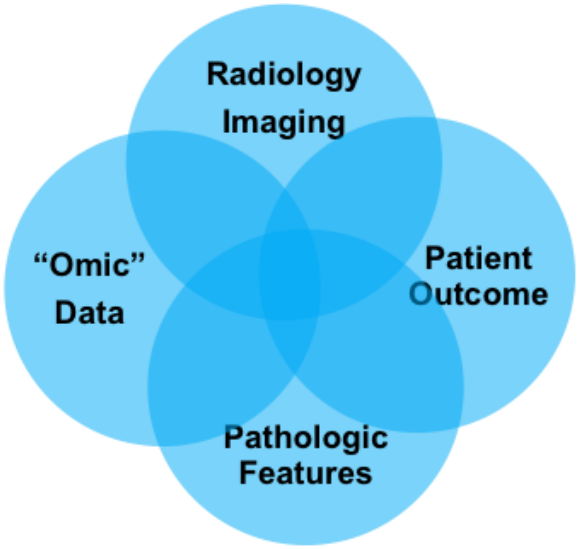
South America, 2004, orbit 022868, Block B61. The plume region is manually labeled with green arrows. Yellow arrow is wind direction. Bright white features are clouds. Red spots are MODIS retrieved fire spots.

Figure 1. Smoke Plume of Satellite Imagery

Satellite Imagery & Remote Sensing

Integrative Multi-Scale Biomedical Informatics

- Reproducible anatomic/functional characterization at gross level (Radiology) and fine level (Pathology).
- Integration with multiple types of "omic"s (genomics, proteomics, metabolomics) information.
- Create categories of jointly classified data to describe pathophysiology, predict prognosis and response to treatment.



Emory In Silico Brain Tumor Research Center

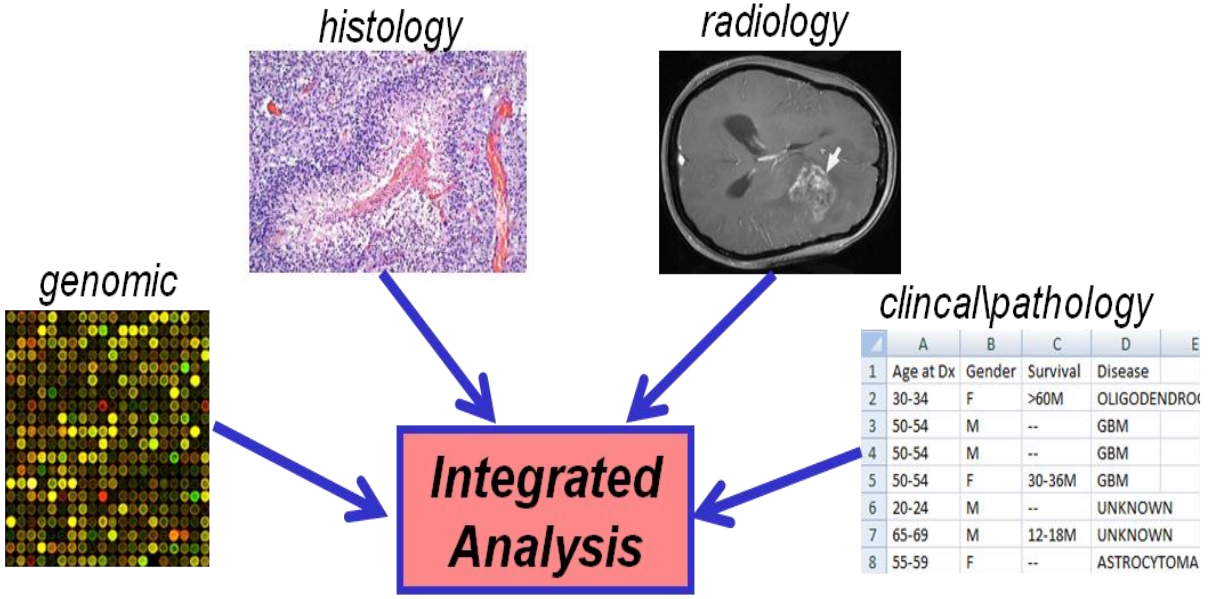
Integrative in-silico study of GBM brain tumors through digital pathology, radiology, "omics" data, and patient outcome.

THE CANCER GENOME ATLAS

REMBRANDT

caBIG™ cancer Biomedical Informatics Grid™

Emory In Silico Brain Tumor Research Center



	A	B	C	D	E
1	Age at Dx	Gender	Survival	Disease	
2	30-34	F	>60M	OLIGODENDRO	
3	50-54	M	--	GBM	
4	50-54	M	--	GBM	
5	50-54	F	30-36M	GBM	
6	20-24	M	--	UNKNOWN	
7	65-69	M	12-18M	UNKNOWN	
8	55-59	F	--	ASTROCYTOMA	

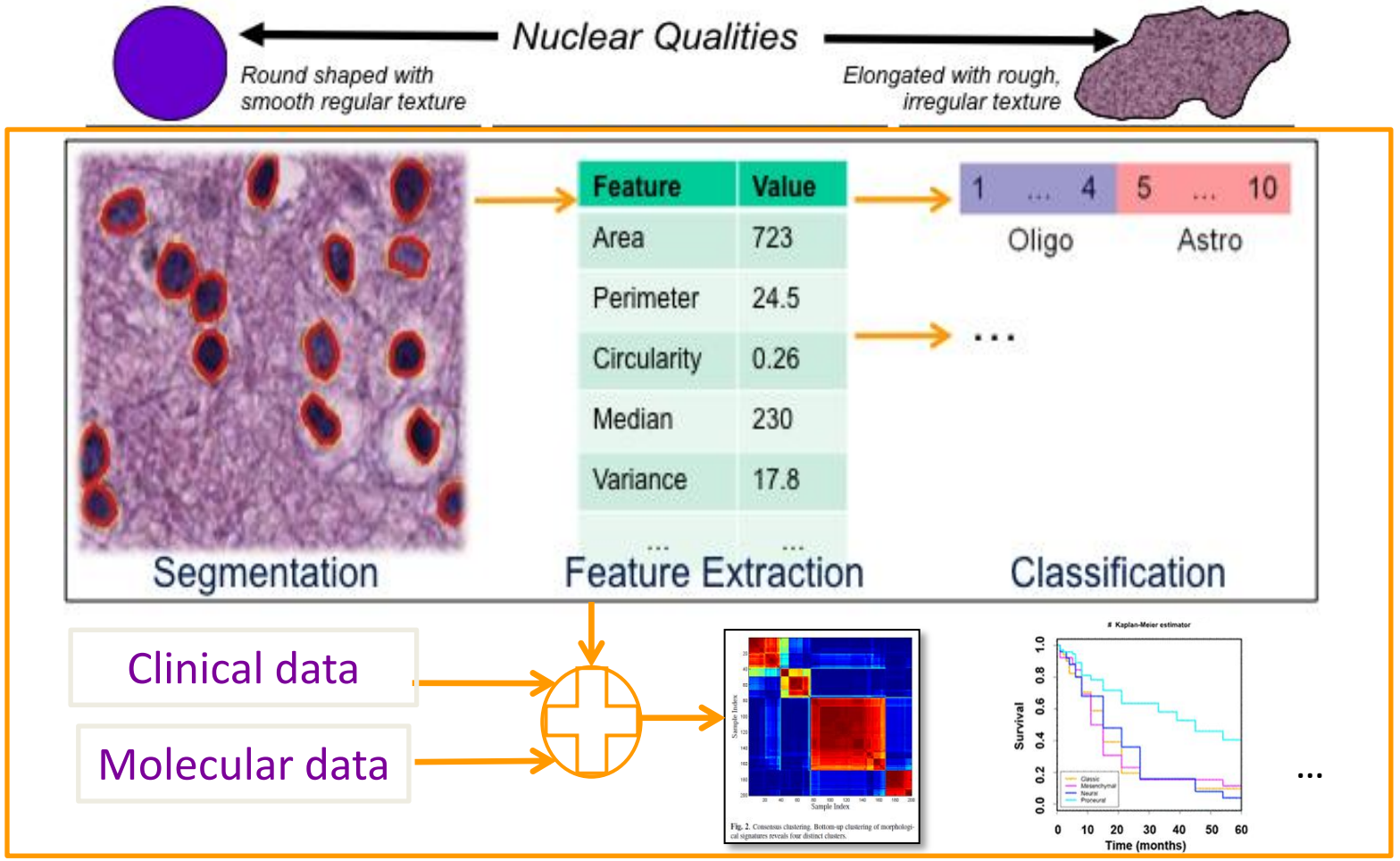
EMORY UNIVERSITY

Pathology Imaging

- High-resolution whole slide images provide rich information about morphological and functional characteristics of biological systems. (Ex1)

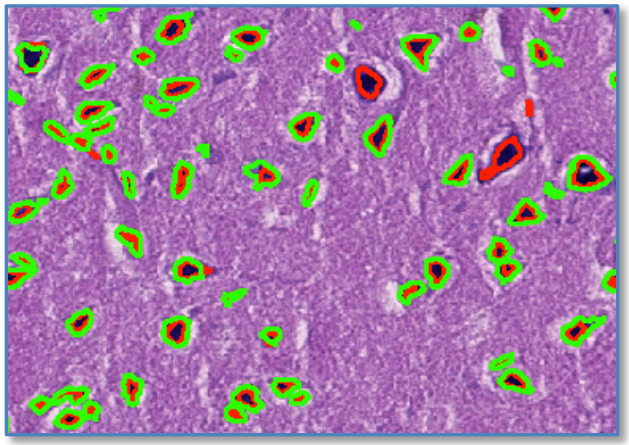
- Such information has tremendous potential for providing insights regarding the underlying mechanisms of disease onset and progression.

Distinguishing Characteristics in Pathology Images

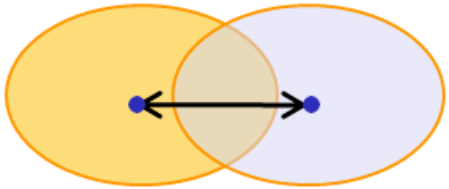


Systematic Image Algorithm Evaluation

- High quality image analysis algorithms are essential to support biomedical research and diagnosis
  - Validate algorithms with human annotations
  - Compare and consolidate different algorithm results
  - Sensitivity study on algorithms' parameters
- Example: What are the distances and overlap ratios between markup boundaries from two algorithms ?



Cross match / join two spatial data sets

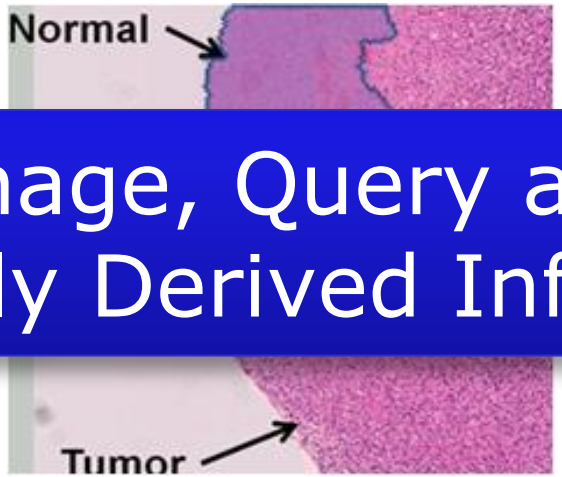


Algorithm one v.s. Algorithm two

Spatial Centric Queries

POINT

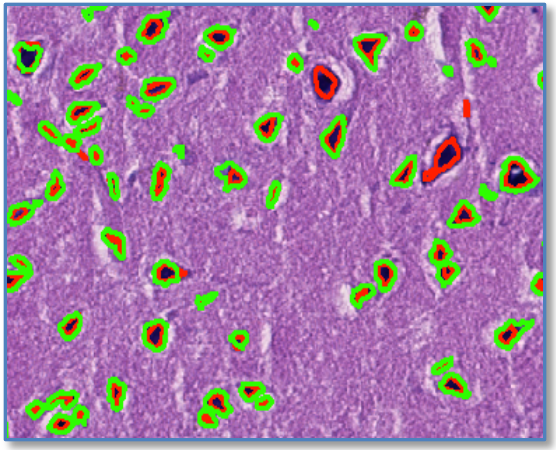
CONTAINMENT



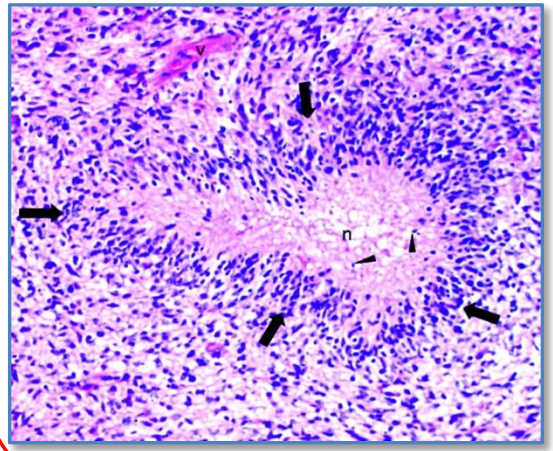
WINDOW

Need: Manage, Query and Compare Spatially Derived Information

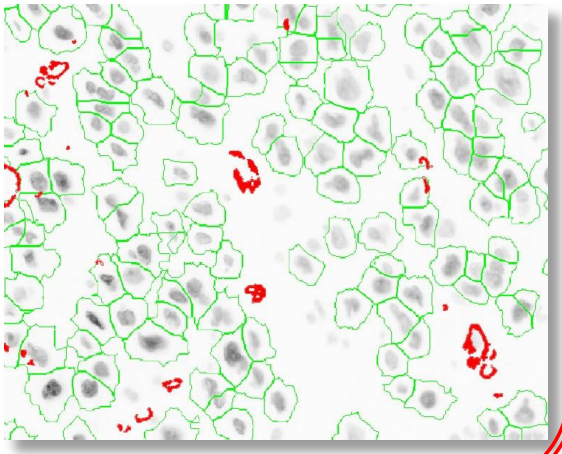
SPATIAL JOIN



DENSITY

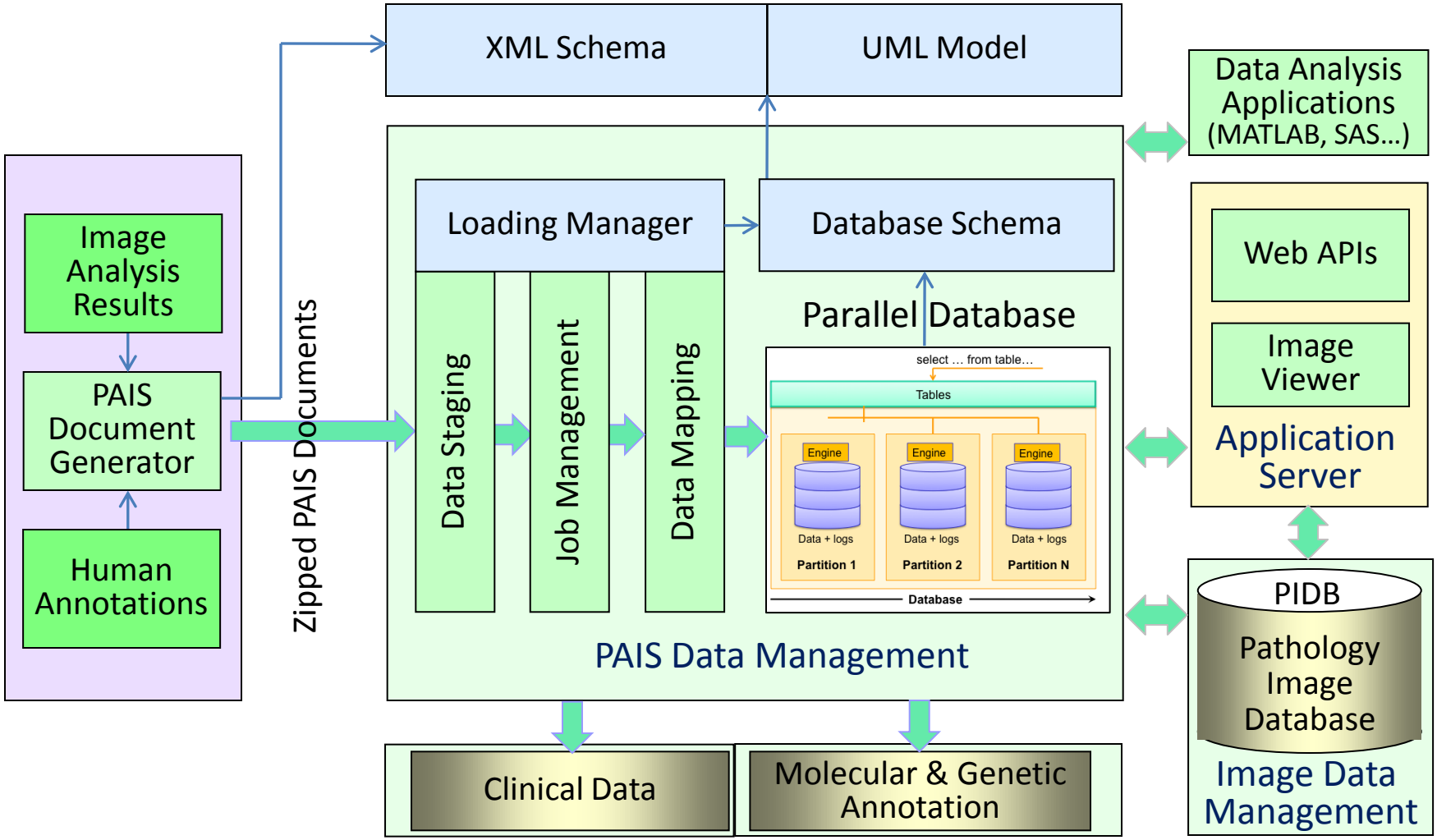


NEAREST NEIGHBOR



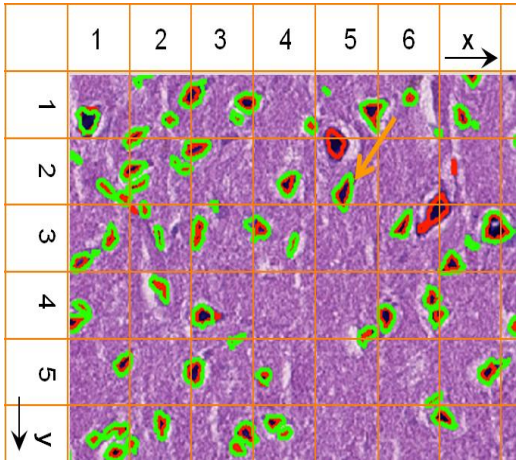
EMORY UNIVERSITY

PAIS Data Management Architecture



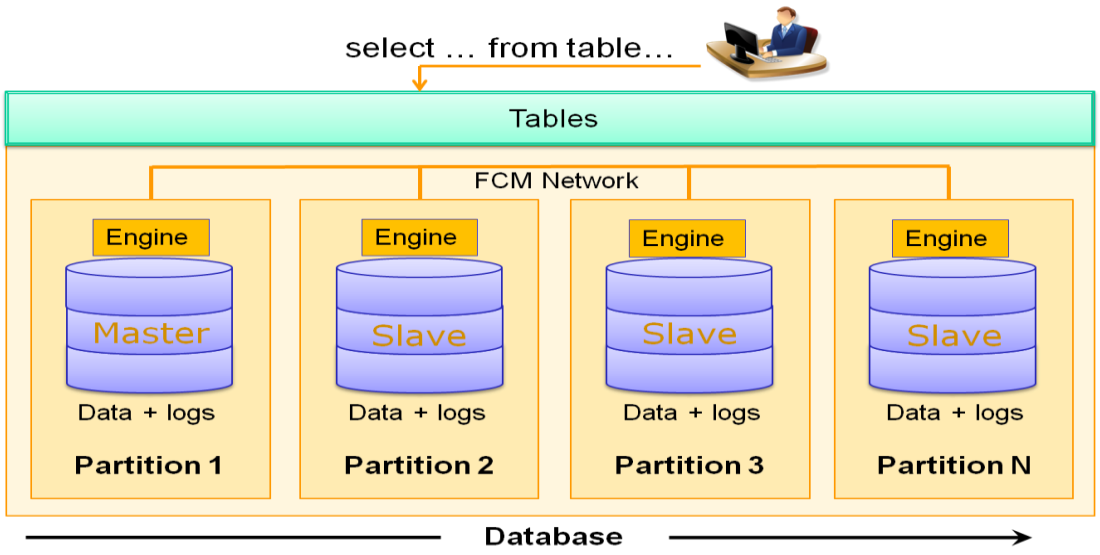
Spatial Database Engine

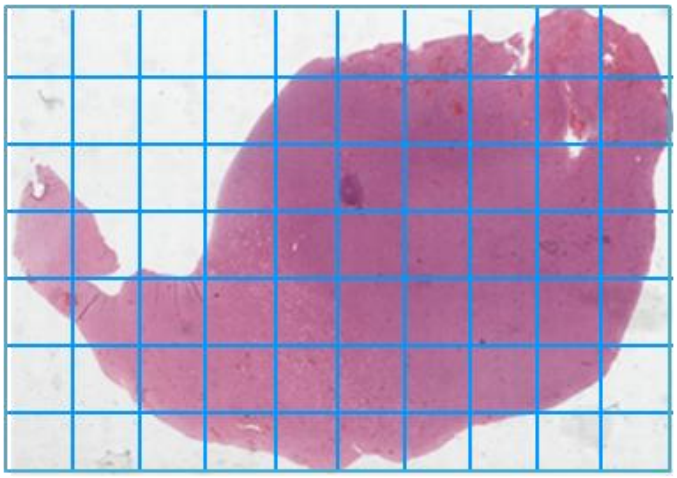
- Extended on RDBMS to support multi-dimensional spatial data types and queries
  - Extended spatial data types
  - Spatial functions and procedures
  - Spatial access methods
    - space oriented – e.g. Grid-File
    - data oriented – e.g. R*-Tree



Partitioned, Shared-Nothing Parallel DBMS

- Most often I/O is the main bottleneck
- Partition data for parallel data access
- Shared-nothing architecture is widely used in practice as a scalable database solution





Parallel Spatial DB Summary

- Comprehensive data model to represent data and provenance.
- Expressive query interface (Spatial SQL).
- Can be scaled with a partitioned parallel DBMS.
- Can be quickly setup, and support a set of queries for small/medium scale data.

Questions ?

Limitations of Parallel Database Approach

- Data loading takes very long time
- Limited scalability*: possible but with high cost*
- Expensive software/hardware license
- Limited query support
- Maintaining & tuning is complex
  - most often needs a DBA

F. Wang et al., High Performance Analytical Pathology Imaging Database for Algorithm Evaluation, In MICCAI-DCI 2011

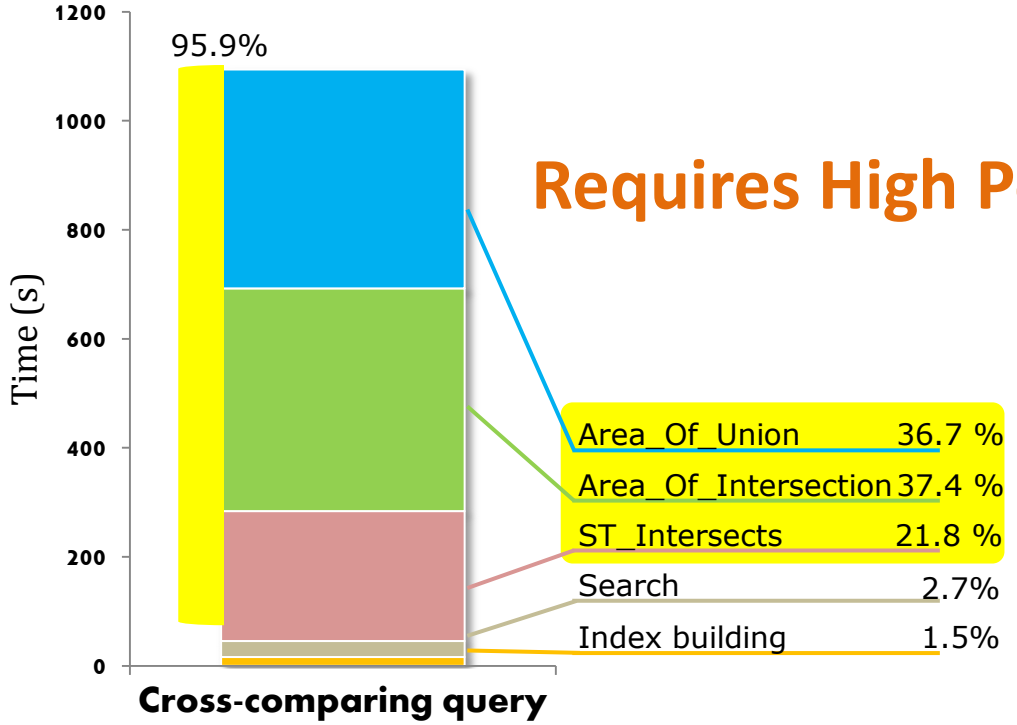
Large Volumes of Derived Data

- 10 billion pixels for each WSI: $10^5 x 10^5$
- 1M derived spatial objects, 100 M features per image
- Thousands of patients, tens of slides /patient
- Many algorithms with varying parameters
- Moderate size healthcare operations can generate Terabytes of data per day easily
- If full WSI would be adopted for clinical environment, petabytes of data per day is likely

Demands High-Throughput

High Complexity of Geometric Computation

```
SELECT  AVG(ratio) FROM  (
        SELECT
           ST_Area(ST_Intersection(p.the_geom, q.the_geom))/
         ST_Area(ST_Union(p.the_geom, q.the_geom))  AS  ratio
        FROM    oligoastroiii_1_1  AS  p,  oligoastroiii_1_2  AS  q
        WHERE  ST_Intersects(p.the_geom, q.the_geom)      );
```



Requires High Performance

EMORY UNIVERSITY

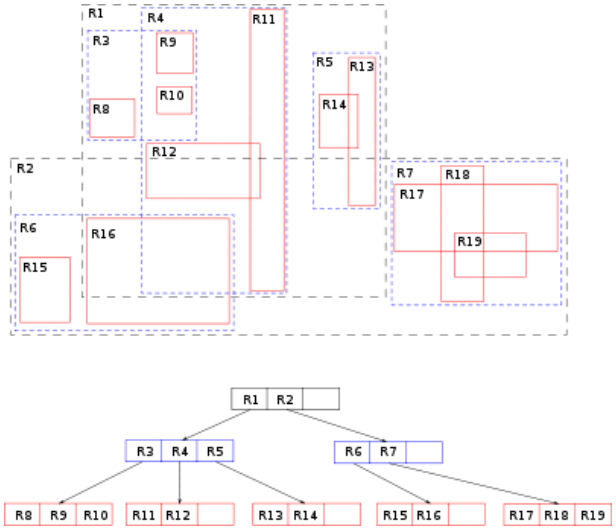
High Performance Queries with MapReduce

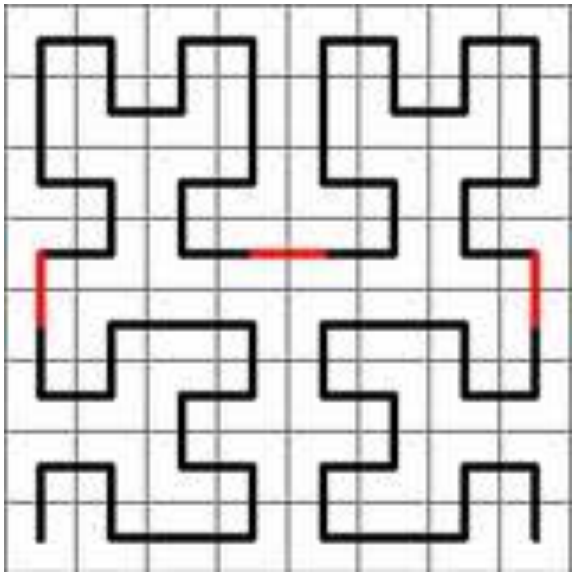
- MapReduce is a parallel computing framework widely used for large-scale data analysis and queries
  - Widely used in major internet applications

    open source version: Hadoop
  - Two simple UDFs for data processing : *map* & *reduce*
  - Very easy to setup develop scalable applications
  - Implicit parallelization by data partitioning
- Our approach:
  - Take advantage of MapReduce to run queries
  - Native spatial query support with a spatial engine

Major Gaps

- Spatial Data Partitioning and Index Support
  - Spatial data is multidimensional
  - Hard to preserve data locality
  - Supporting and managing multidimensional index on MapReduce is not easy

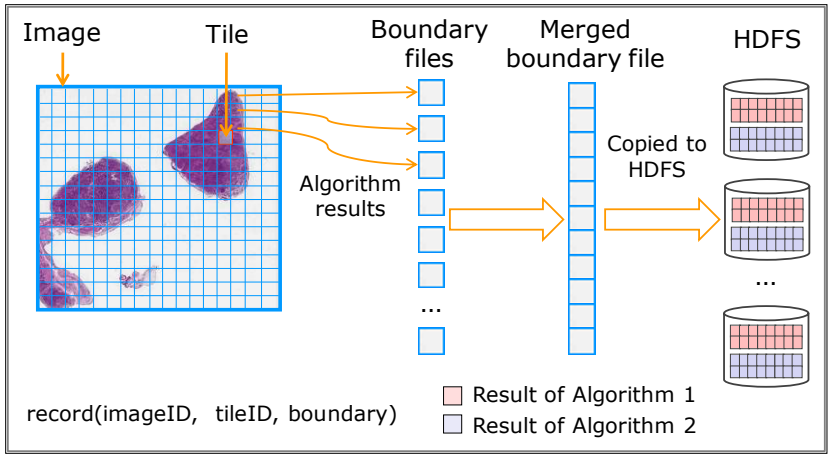


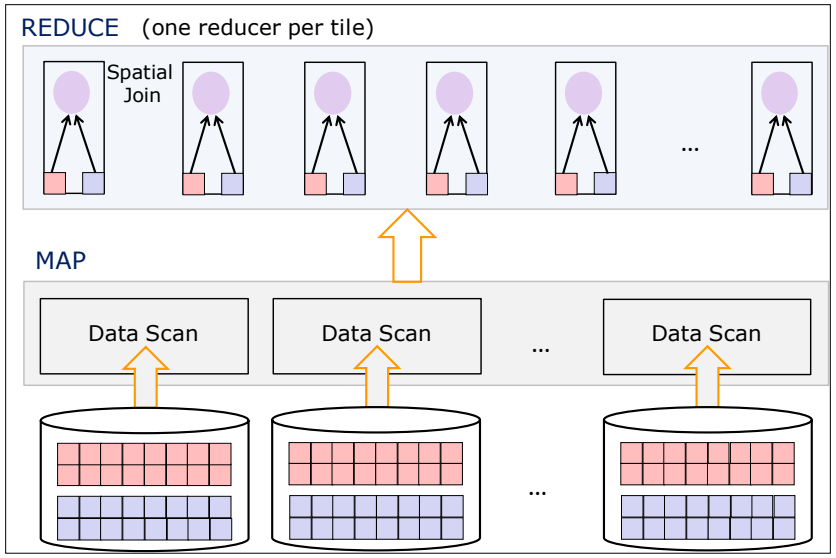


Major Gaps

- MapReduce only provides map() and reduce()
  - Need to map spatial query processing algorithms to map and reduce environment
  - Need to revise and rethink spatial query processing
- An easy to use query interface is required
  - MapReduce == assembly language for MPP
  - Industrial examples – Hive, Pig, Scope ..
  - e.g. in Facebook, 70% of MapReduce workload are submitted through Hive

Spatial Join Processing with MapReduce

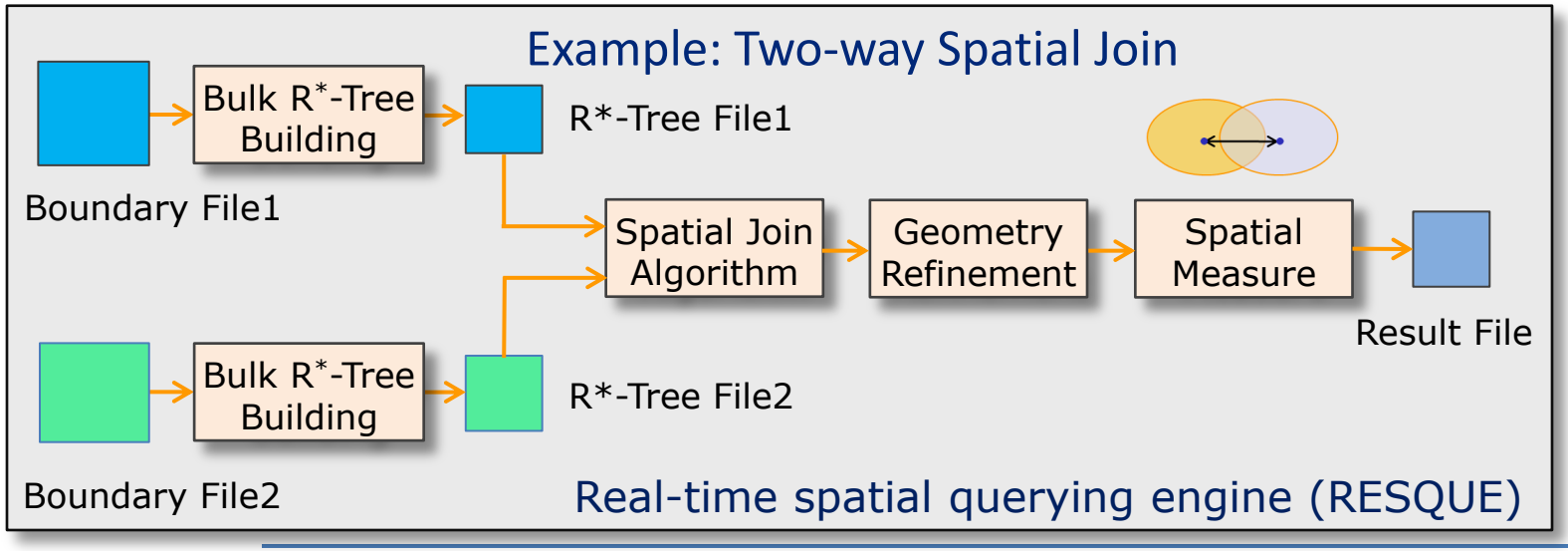




- Staging:
  - images are tiled into regular grids
  - tiles redistributed across HDFS as file blocks
  - small tiles are merged and metadata added into records
- Map:
  - identify records of same tiles to form tasks
- Reduce:
  - execute queries with the spatial query engine
  - aggregate query results

Real-Time Spatial Query Engine (RESQUE)

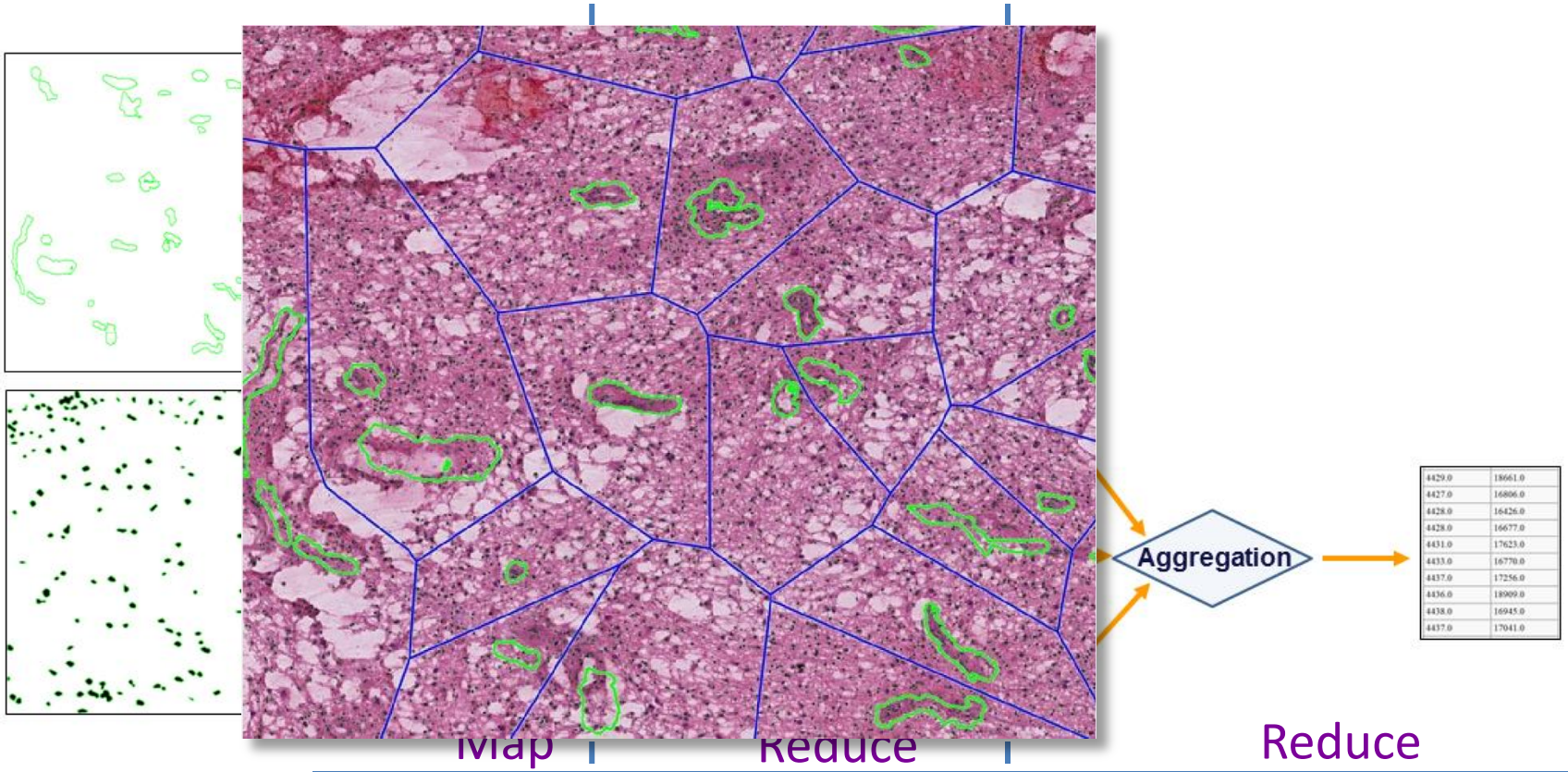
- Index building **on demand** (low overhead)
- Query pipelines to combine multiple steps of query processing
- Decoupled parallel spatial query processing
- Support of spatial access methods and algorithms
  - spatial selection, multi-way spatial join, nearest neighbor, and highest density queries, and extensible for new ones



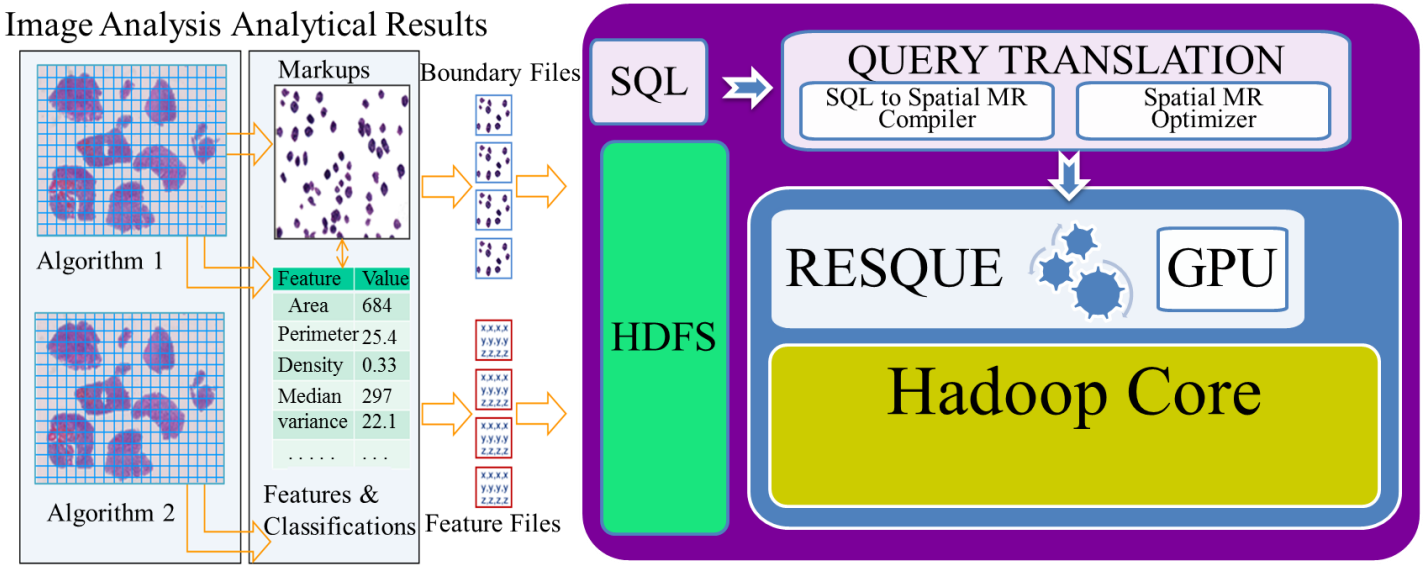
Nearest Neighbor Query Processing Workflow

Query e.g. : for each cell find the closest blood vessel and return distance to that blood vessel.

Access methods can vary (R*-Tree, Voronoi, etc..)



System Architecture — Hadoop-GIS



Goal: provide end-to-end support (from storage and indexing to query execution) for high performance spatial queries over big spatial scientific datasets

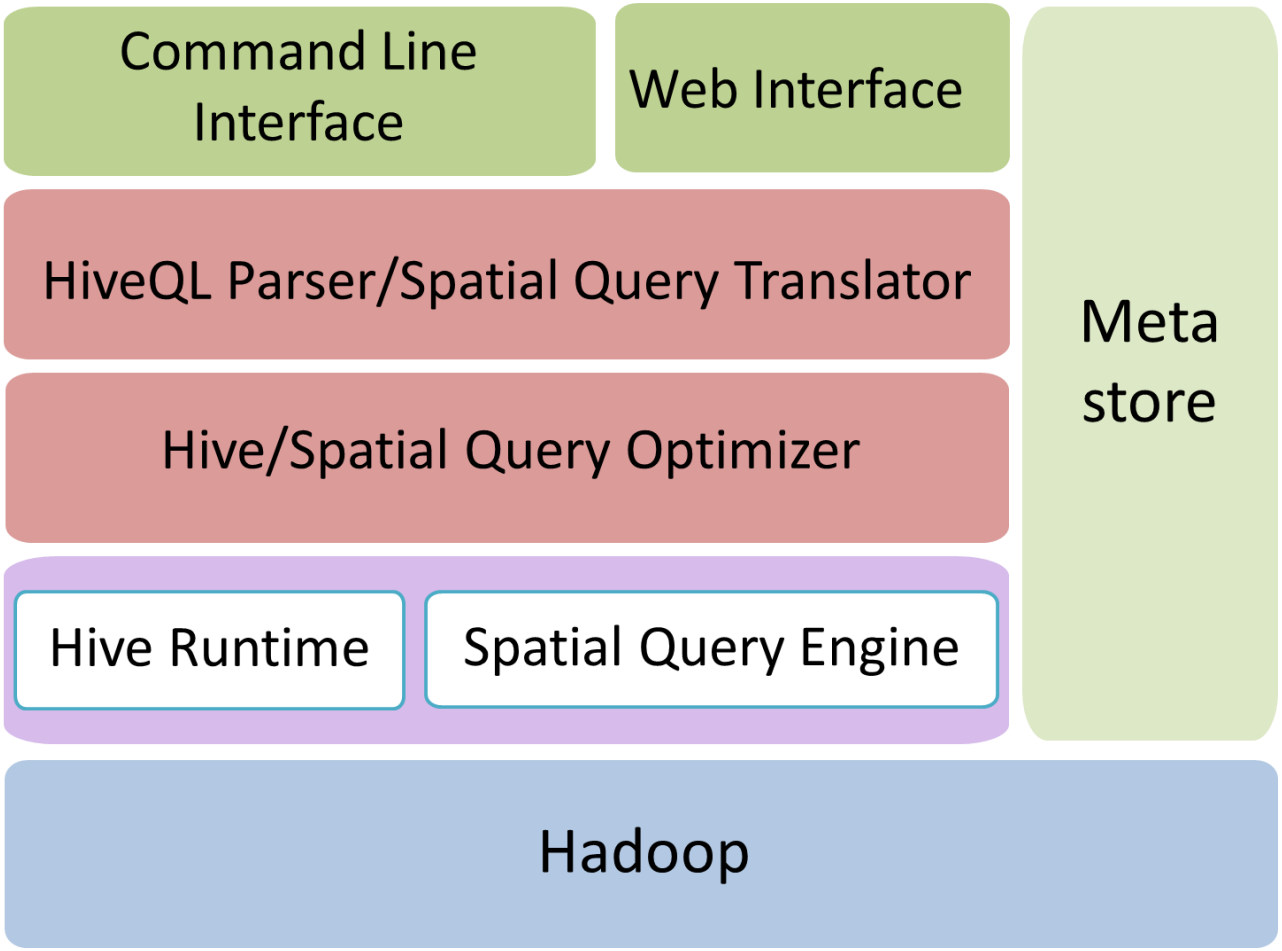
- SQL query interface for ease of use
- Query execution translated into MapReduce code
- Hadoop executes the translated MapReduce code
- Relies on the RESQUE for spatial query processing

Hadoop-GIS

- HiveQL –> Hive^{SP}
  - Pros: Widely used as EDW, Tightly integrated with Hadoop
  - Mature query processing pipeline is already in place
  - Cons: Married to Hive, less freedom for optimization

- YSmart –> YSmart-Spatial
  - Pros: Good query optimization framework
  - Better control of customized optimization (e.g. GPU)
  - Cons: Not widely used, lots of development

Hadoop-GIS: Hive^{SP}



Declarative Query Interface

Schema Creation:

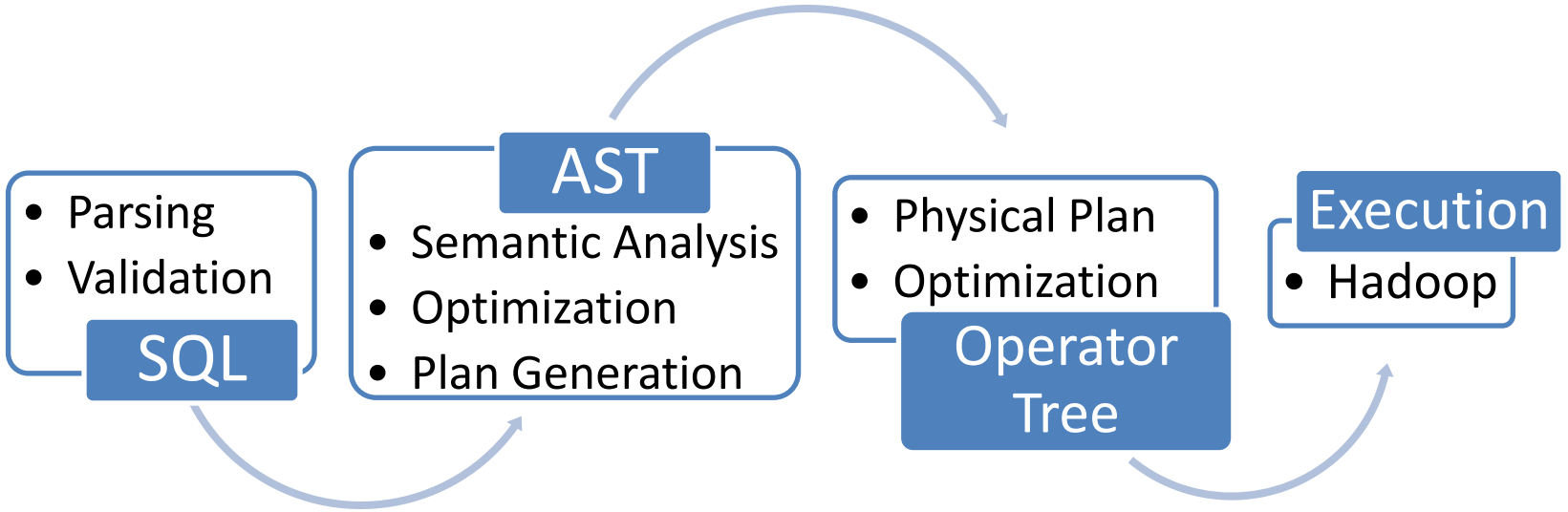
```
CREATE TABLE tcga_markups
( markup_id BIGINT, provenance STRING,  hand_marked
     BOOLEAN, center ST_POINT, polygon ST_POLYGON)
PARTITIONED BY TILE(polygon ST_POLYGON, 4096, 4096)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' STORED AS
TEXTFILE ;
```

Spatial Join Query:

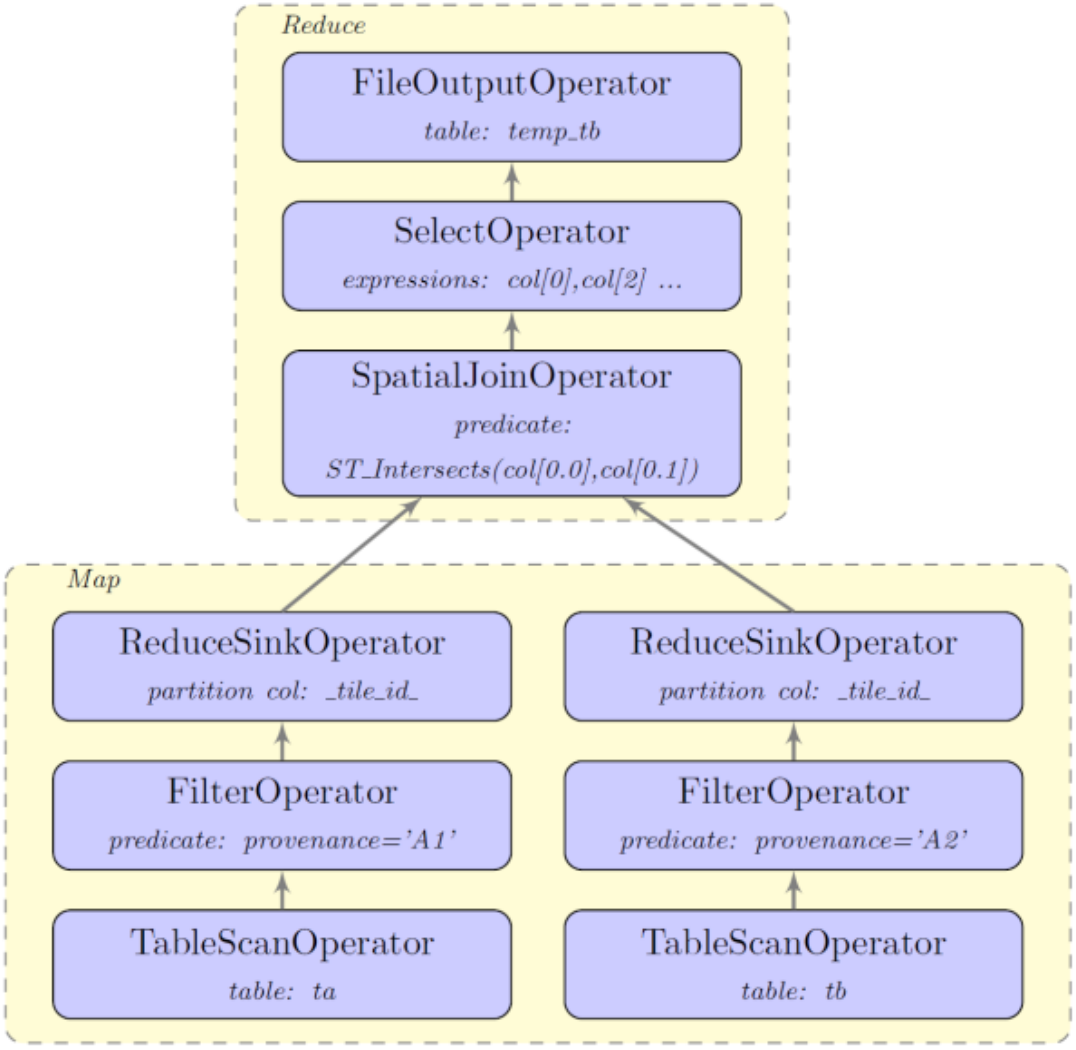
```
SELECT
      ST_AREA(ST_INTERSECTION(ta.polygon,tb.polygon))/
      ST_AREA(ST_UNION(ta.polygon,tb.polygon))  AS ratio,

ST_DISTANCE(ST_CENTROID(tb.polygon),ST_CENTROID(ta.polygon))
AS distance
FROM  tcga_markups ta JOIN  tcga_markups ON
     (ST_INTERSECTS(ta.polygon, tb.polygon) = TRUE)
WHERE ta.provenance='Algorithm1' AND
tb.provenance='Algorithm2' ;
```

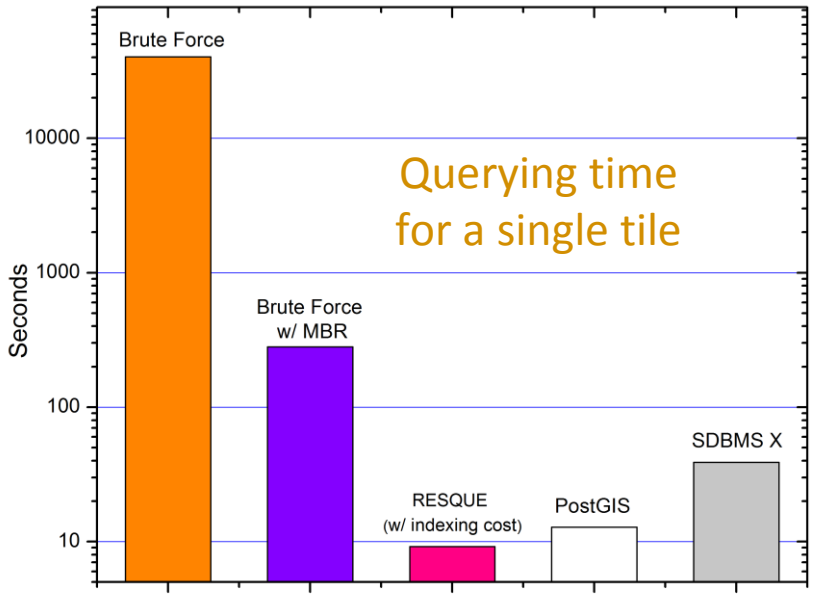
Query Processing Pipeline

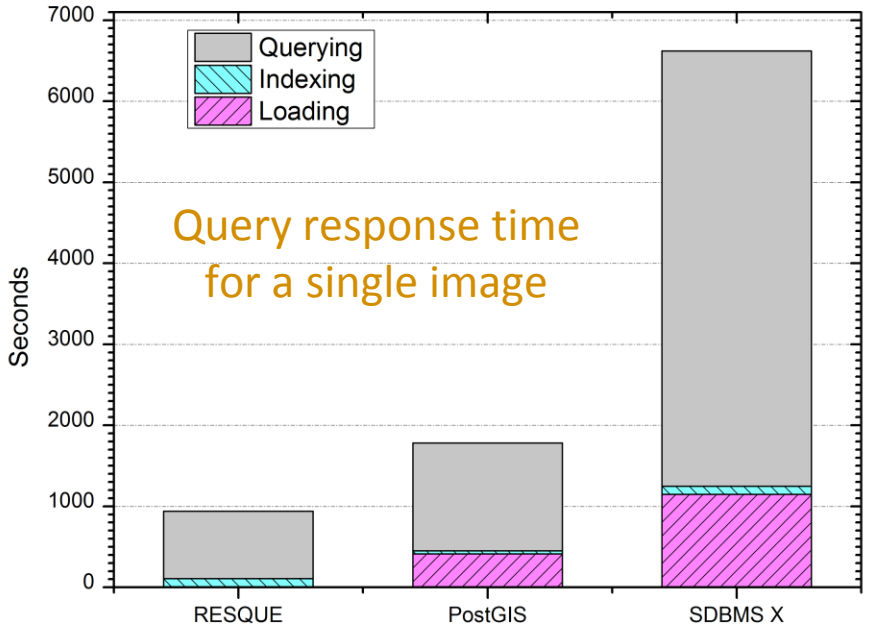


Two-way Join Query Plan



Spatial Query Engine Performance





- R*-Tree building ≈ 16% R*-Tree join ≈ 84%
- Indexing Scalability: IndexTime (Single Image) ≈ IndexTime(Tile) * tile#
- Storage: apply chain coding at R*-Tree leaves ≈ 42% storage reduction

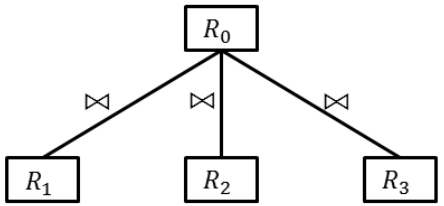
System Scalability Experiments

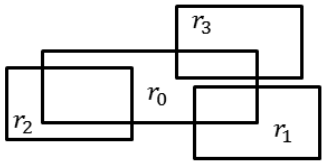
- Experimental Setup
  - in-house cluster with 10 physical nodes (192 cores)
  - join query: a set of 18 images (0.5 M nuclei/image)
    - 39GBs of data with 6 different results
  - nearest neighbor query: 50 images from TCGA (size)
    - 42 GBs
- Hadoop Setup
  - Cloudera Hadoop 0.20.2
  - Replication factor = 3
  - File split size = 128 MB
  - Individual task tracker memory = 512MB

Query Setup

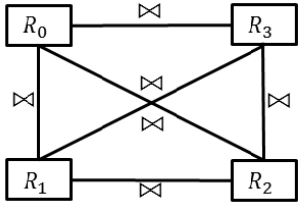
- Multi-way Spatial Join Query

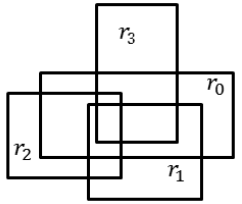
star join





clique join

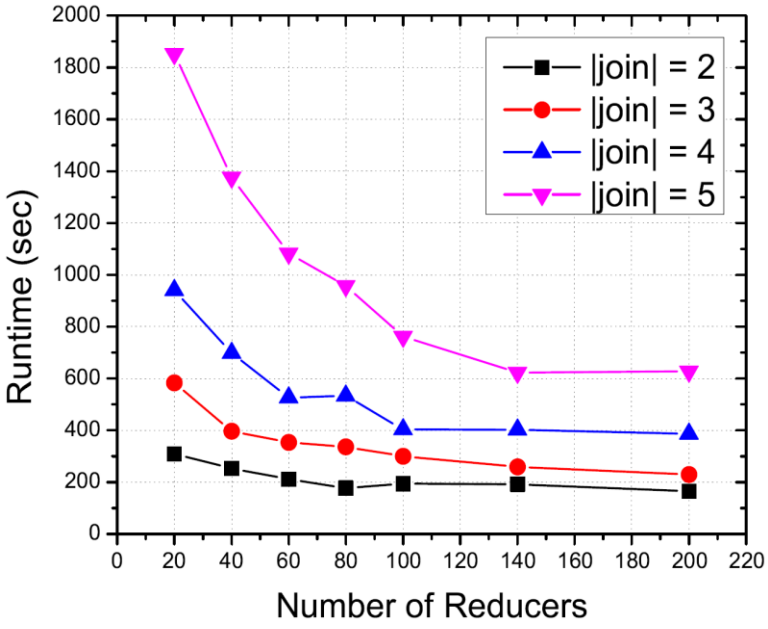




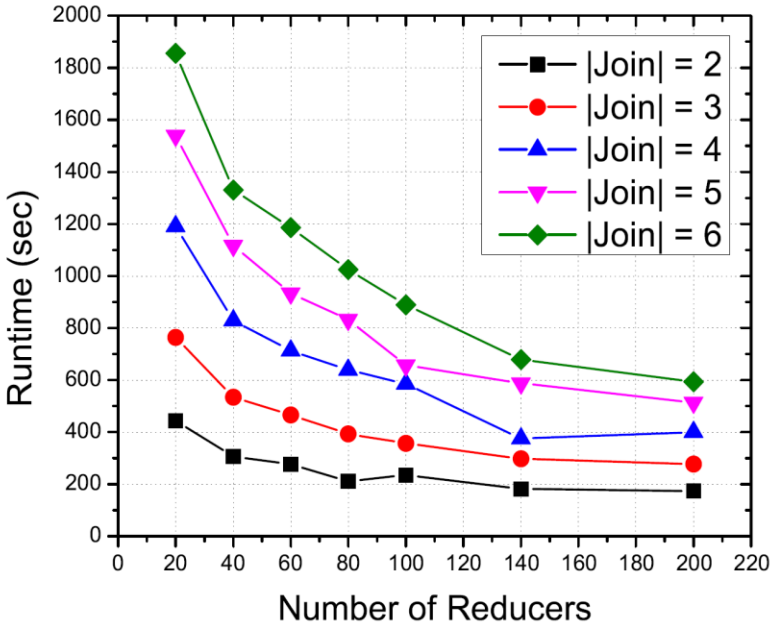
- Join Algorithms
  - R*-Tree join (Breadth First Tree Traversal)
  - Partition based spatial merge join (PBSM)
    1. refine with approximate processing
    2. filter candidates with precise algorithm

Multi-Way Spatial Join: Star Join

R*-Tree

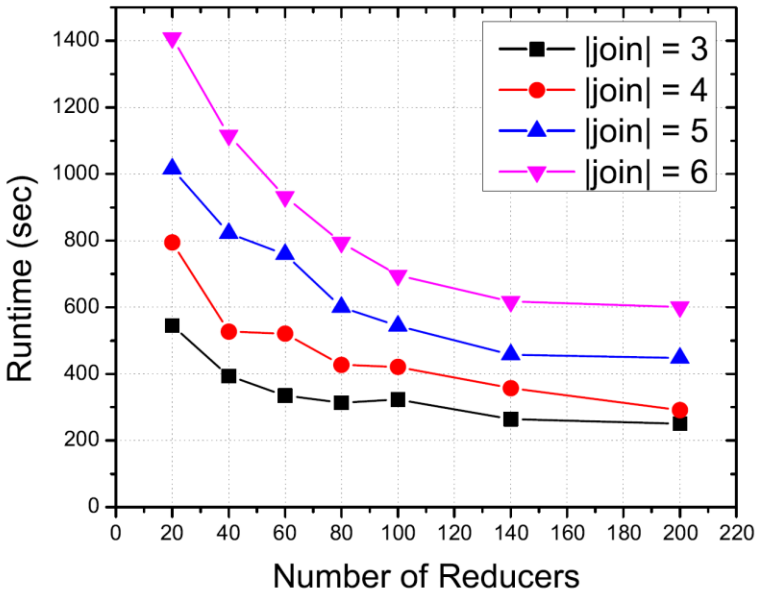


PBSM

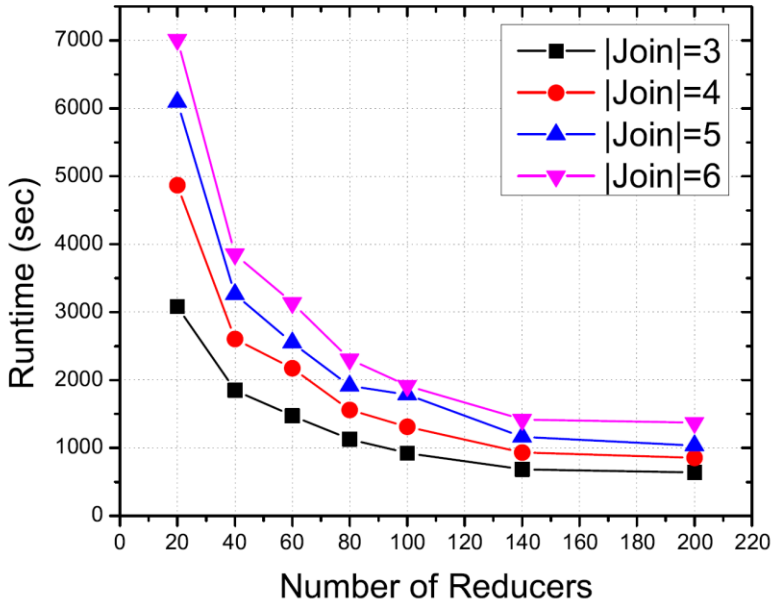


Multi-Way Spatial Join: Clique Join

R*-Tree

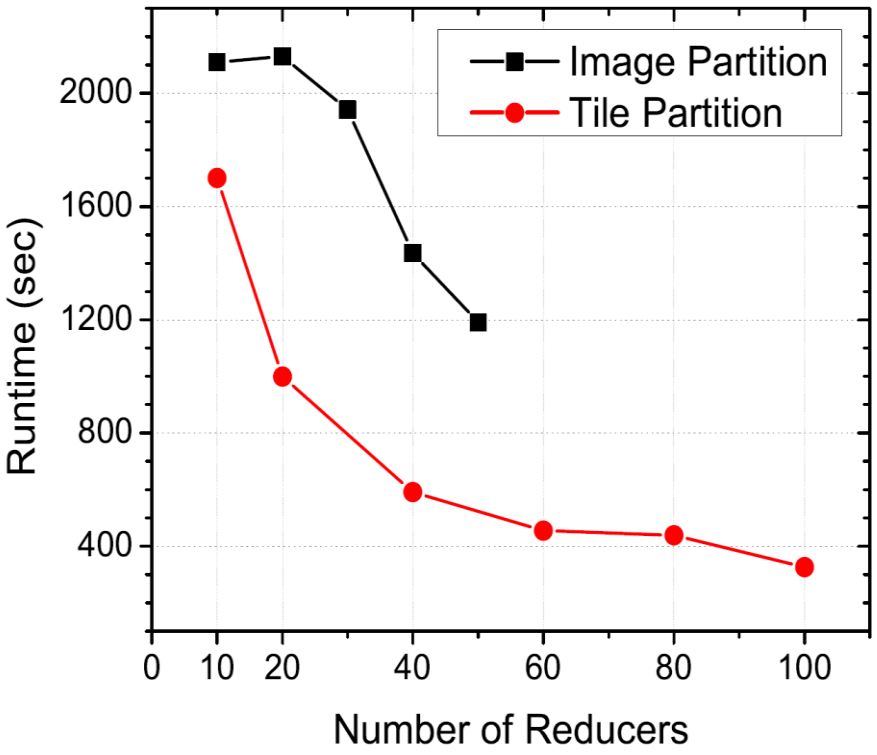


PBSM

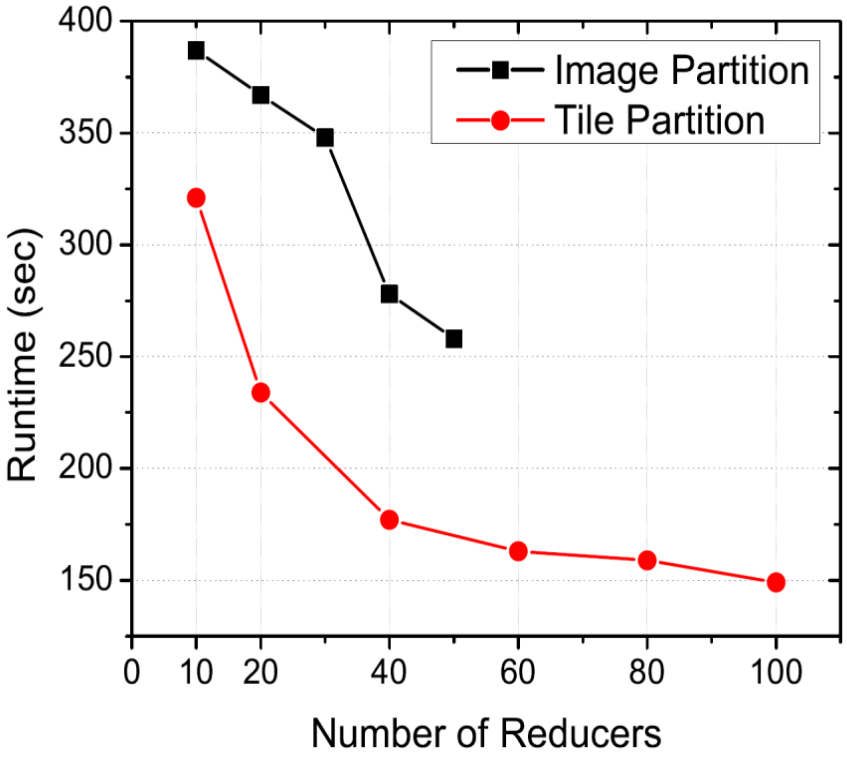


Nearest Neighbor Query

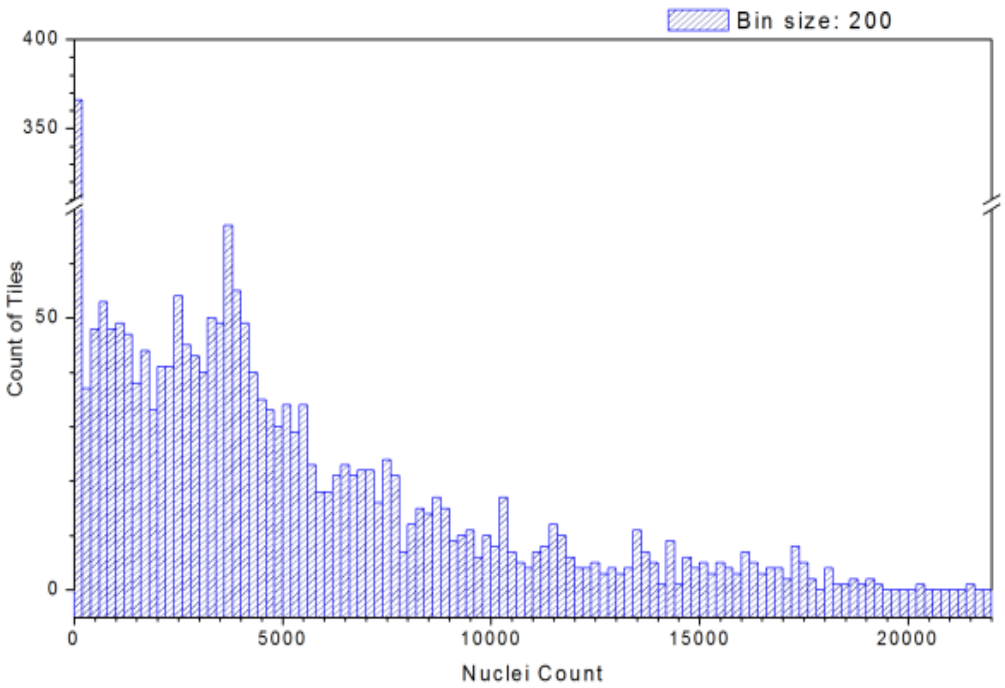
R*-Tree

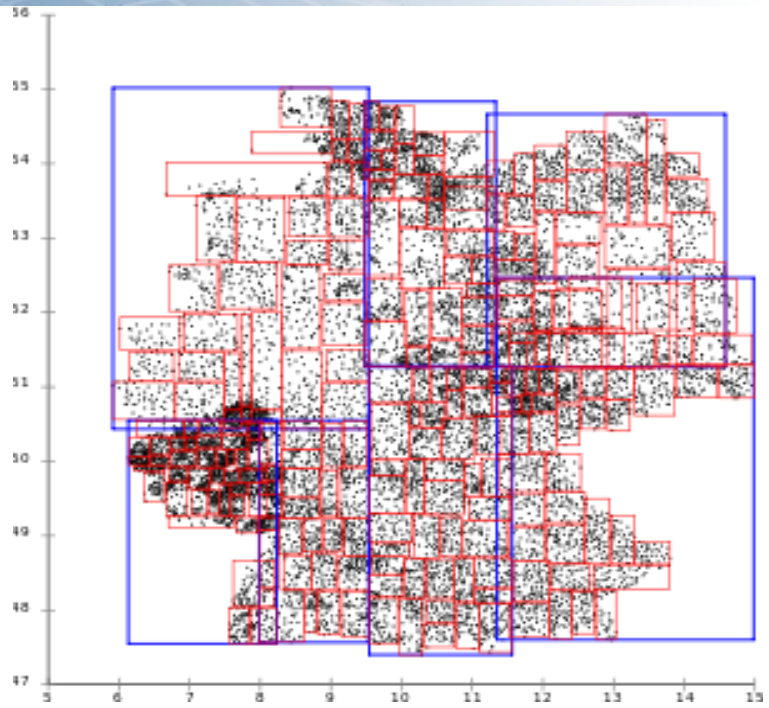


Voronoi Diagram



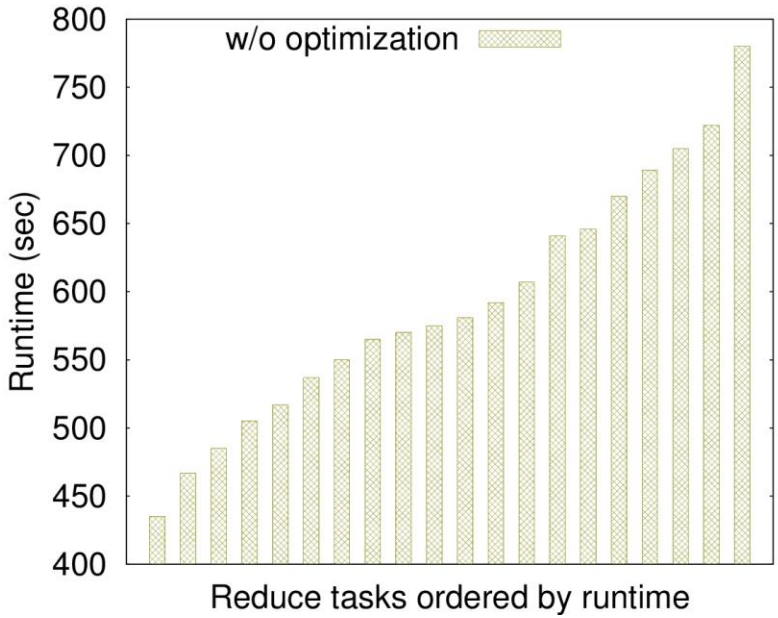
Data Skew in Spatial Query Processing

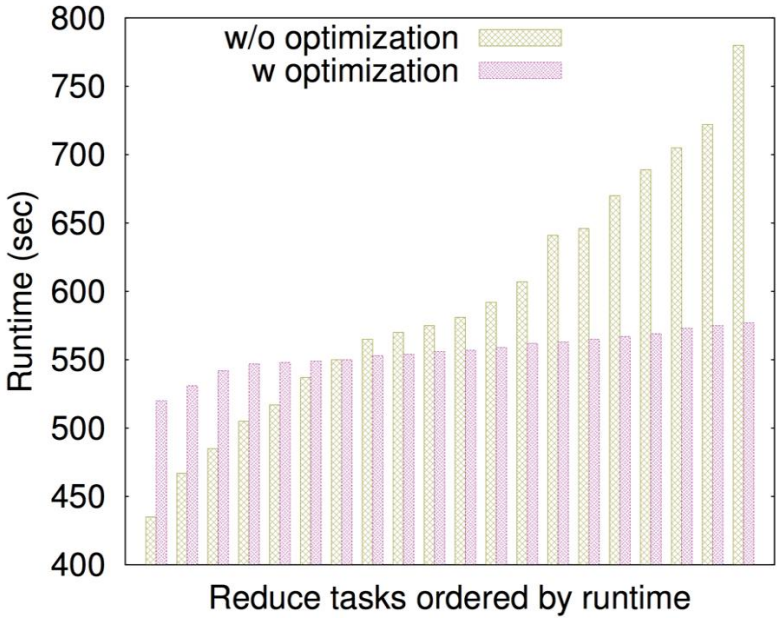




- Skew may arise for many reasons
- Longest running task becomes bottleneck

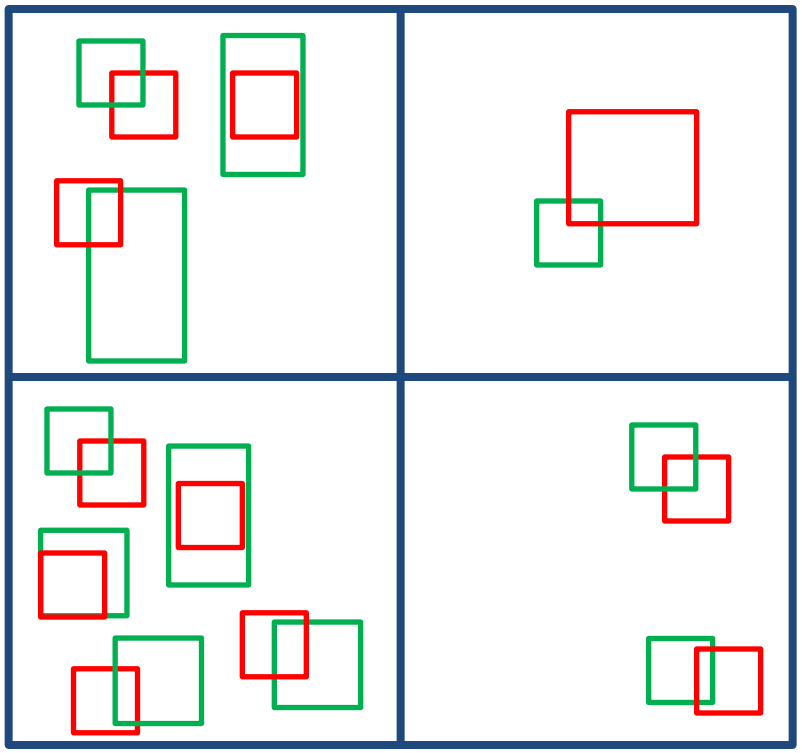
Stragglers


w/o optimization
800
750
700
650
600
550
500
450
400
Runtime (sec)
Reduce tasks ordered by runtime

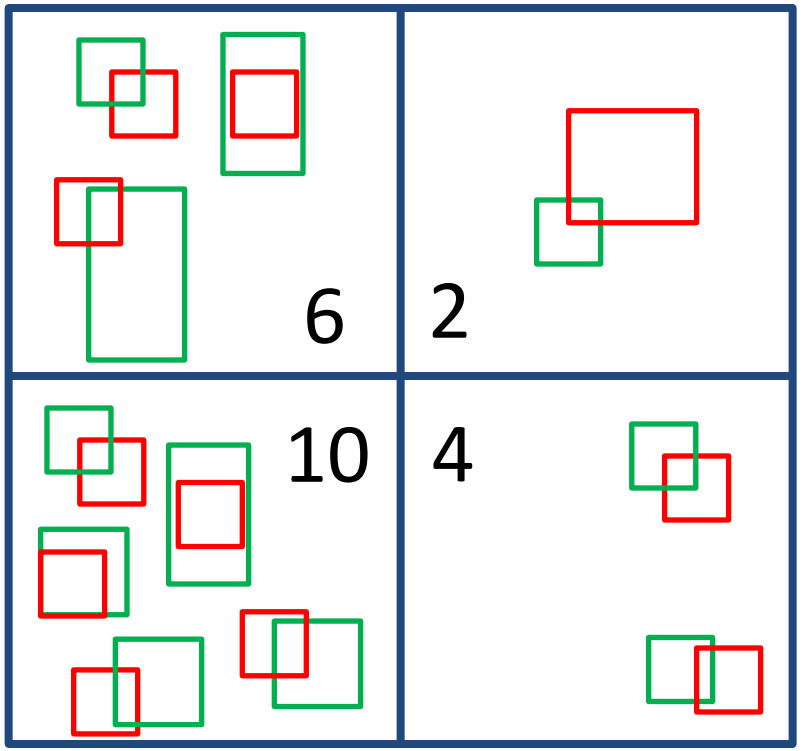

w/o optimization
w optimization
800
750
700
650
600
550
500
450
400
Runtime (sec)
Reduce tasks ordered by runtime

Skew Mitigation

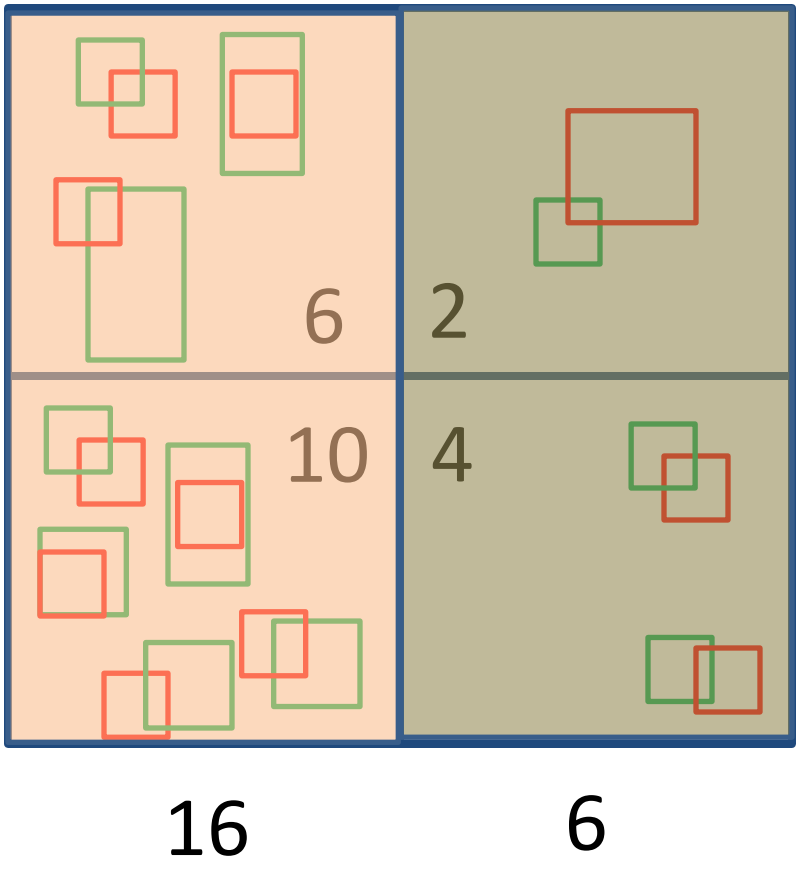
- Hadoop generates tasks using hash partitioning (default)
- Hash partitioning is not skew resistant
- We overwrite the default hashing partition function



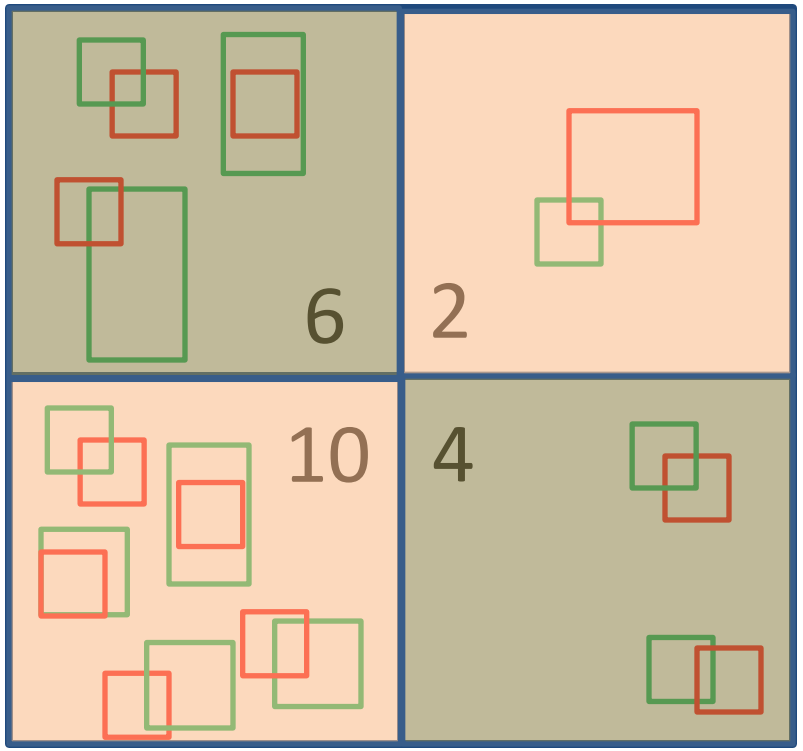
Cost Estimation



Repartition (hash)



Repartition (optimal)



12

10

Query Optimization

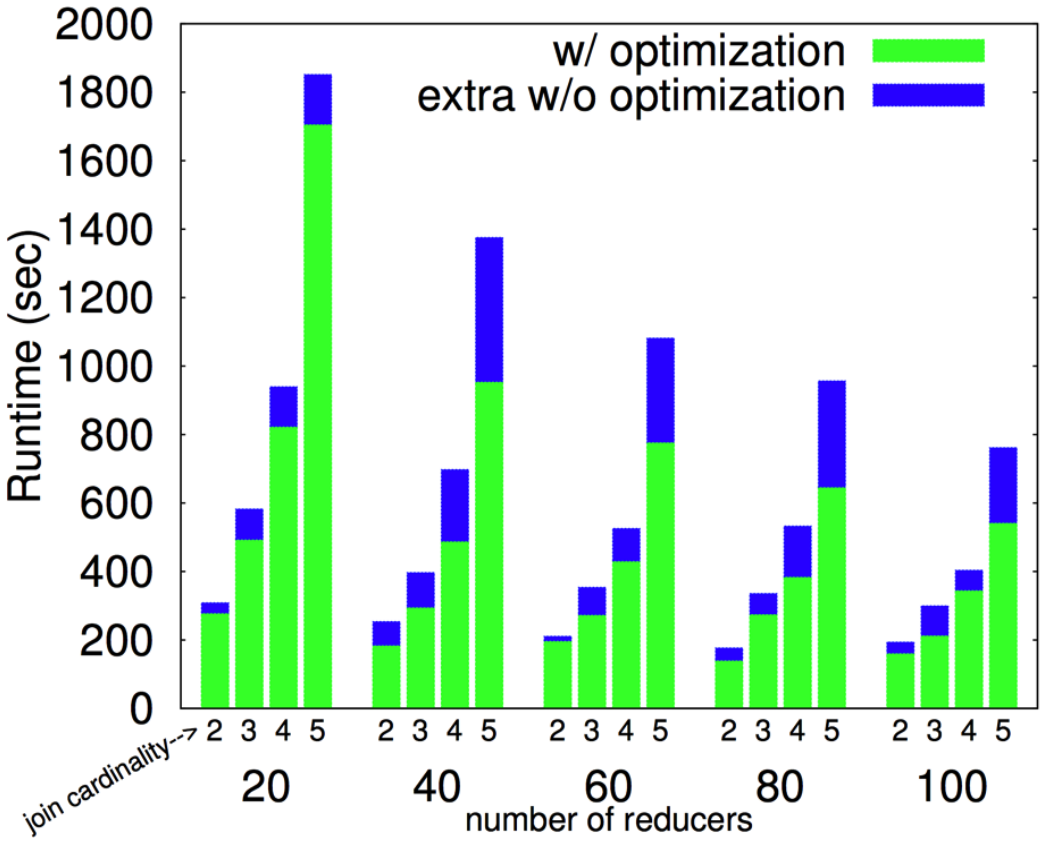
- Partition problem is NP-Complete
- Greedy Approach ≈ 4/3 OPT (not bad)

$$Q = R \overset{intersects}{\bowtie} S = \bigcup_{i=1}^{N} R_i \overset{intersects}{\bowtie} S_i$$

$$W_j = \sum_{i \in P_j} Cost(R_i \bowtie S_i)$$

$$Cost(R_i \bowtie S_i) = \alpha|R_i| + \beta|S_i| + \gamma$$

Query Optimization



Summary

- Effective management of **big spatial data** is one of the pressing challenges for next generation integrative biomedical research.
- We propose and provide a high performance **MapReduce** based **querying system** for large scale spatial data.
- We **empirically** test the concept with analytical medical imaging as an example application.
- The system is **efficient**, **cost effective**, **scalable**, and provides **expressive query language**.

Next Steps (I)

Spatial Partitioning and Optimization Methods for MapReduce

- Density aware partition methods
  - partition according to the local spatial density
  - dynamically adjust partition granularity
- Index and pipeline selection
  - one spatial index does not fit all (R*-Tree v.s Voronoi etc..)
  - select index based on query type data set properties
  - pipeline optimization

Next Steps (II)

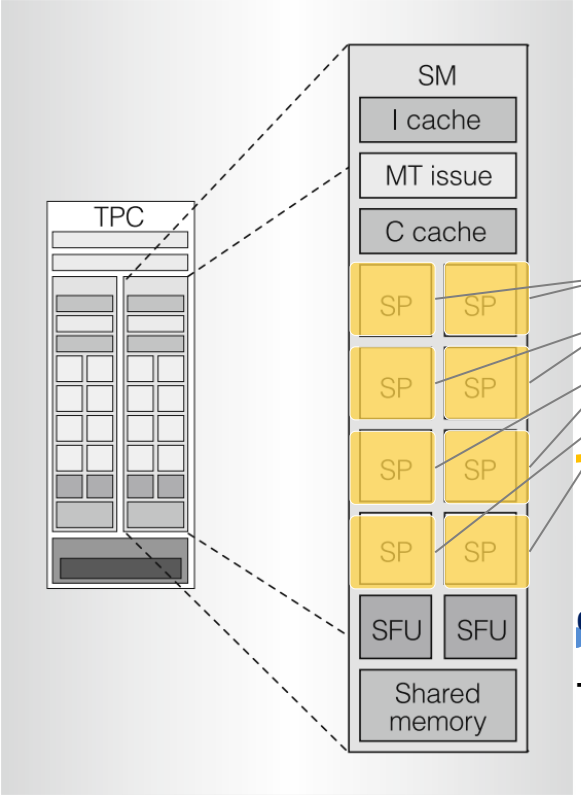
Spatial Query Optimization

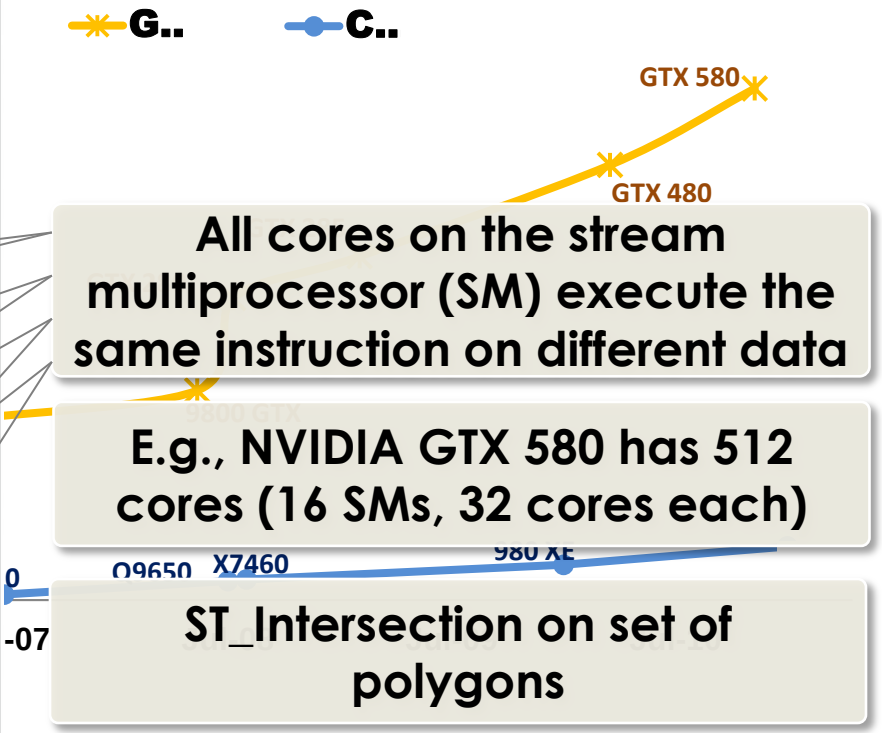
```
SELECT  AVG(ratio) FROM  (
        SELECT
           ST_Area(ST_Intersection(p.geom, q.geom))/
          ST_Area(ST_Union(p.geom, q.geom))  AS  ratio
        FROM    oligoastroiii_1_1  AS  p,  oligoastroiii_1_2  AS  q
        WHERE  ST_Intersects(p.geom, q.geom)      );
```

```
SELECT  AVG(ratio) FROM  (
      SELECT area_intersect/ (area_p + area_q -area_intersect) AS ratio
      FROM (
      SELECT ST_Area(ST_Intersection(p.geom, q.geom)) AS area_intersect,
             ST_Area(p.geom) AS area_p, ST_Area(q.geom) AS area_q
      FROM    oligoastroiii_1_1  AS  p,  oligoastroiii_1_2  AS  q
      WHERE  p.geom && q.geom) AS temp1 )
      WHERE area_intersect >0 ) AS temp2 ;
```

Next Steps (III)

Turbo Charge Hadoop-GIS with GPU



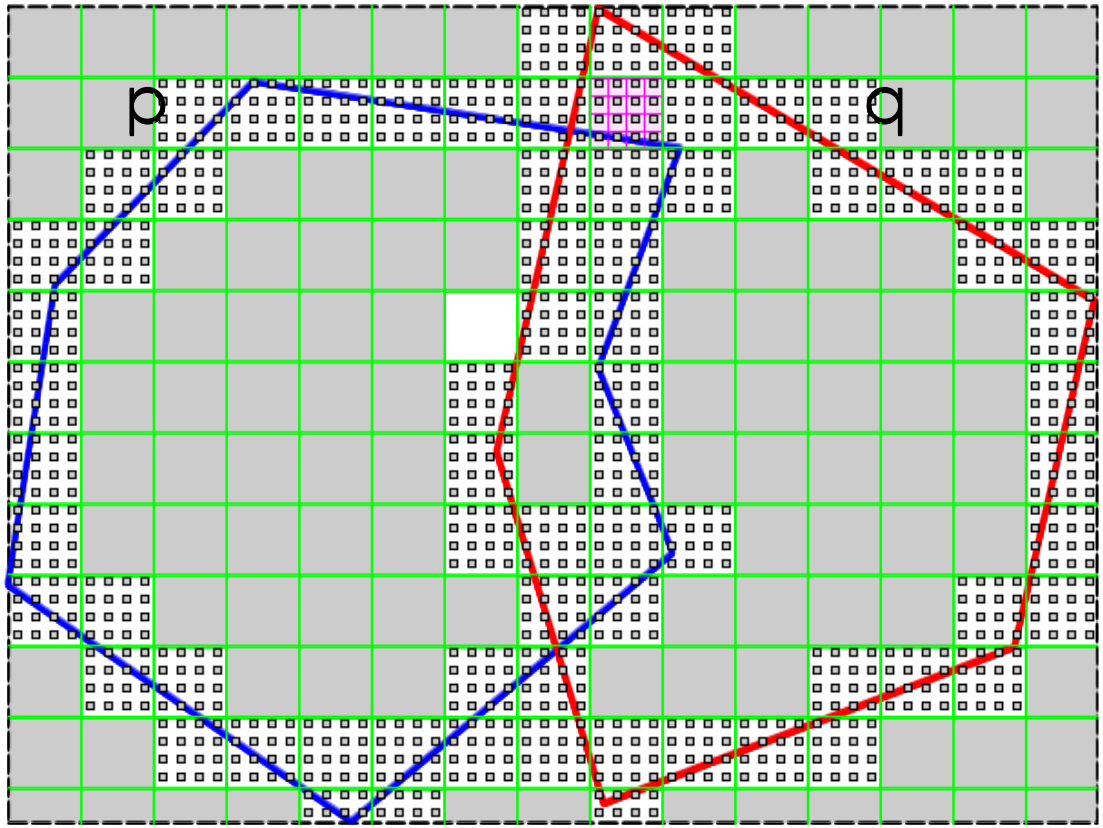


All cores on the stream multiprocessor (SM) execute the same instruction on different data

E.g., NVIDIA GTX 580 has 512 cores (16 SMs, 32 cores each)

ST_Intersection on set of polygons

Reduce Unnecessary Computation



K. Wang et al., Accelerating Pathology Image Data Cross-Comparison on CPU-GPU Hybrid Systems, In PVLDB 2012.

Next Steps (IV)

MapReduce + DB ≈ Hybrid System

- Database
  - A mature storage engine
  - Good single node performance (40 years of research)
  - But not scalable
- MapReduce
  - Highly scalable and fault tolerant
  - But low single node throughput
- Best of the two worlds ?
  - current efforts: HadoopDB, Secondo
  - how to merge the two for big spatial data ?

Questions?

Hadoop-GIS:

https://web.cci.emory.edu/confluence/display/HadoopGIS