第六章 空间查询处理与优化

陶煜波
计算机科学与技术学院

第六章 空间查询处理与优化

- 6.1 查询处理与优化
- 6.2 空间查询处理算法
- 6.3 查询优化
- 6.4 发展趋势 (自学)

References:

Spatial Databases: A Tour, Chapter 5

空间数据库管理系统概论，第七章

Analogy of Automatic Transmission in Cars

- Manual transmission : automatic :: Java : SQL
- Ex. List facilities within 10km of Minneapolies (44.978, -93.265)

Java JDK 1.8

```
Public class Facility {
    Protected String name;
    Protected Point location;
}
```

```
public class FacilityCollection {
    Protected Arraylist<Facility> facilityList;

    private double distance (Facility f, Point p) {
        return Math.sqrt((p.x - f.location.x)*(p.x - f.location.x)
                +(p.y - f.location.y)*(p.y - f.location.y));
    }

    public boolean withinDistance (Point p, double d) {
        for (int i = 0; i < facilityList.size(); i++) {
            if (distance(facilityList.get(i).location, p) < d)
                return true;
        }
        return false;
    }

    public static void main(String[] args) {
        FacilityCollection fCollection = new FacilityCollection();
        fCollection.withinDistance(new Point (44.978, -93.265), 10);
    }
}
```

Manual

Automatic

SQL (Oracle spatial)

```
Select f.name From Facility f
where SDO_WITHIN_DISTANCE (f.shape,
        SDO_GEOMETRY(2001,4326,SDO_POINT_TYPE(44.978,-93.265,NULL)
        ,NULL,NULL),distance=10') = 'TRUE'
```

SQL : Java :: Automatic Transmission : Manual

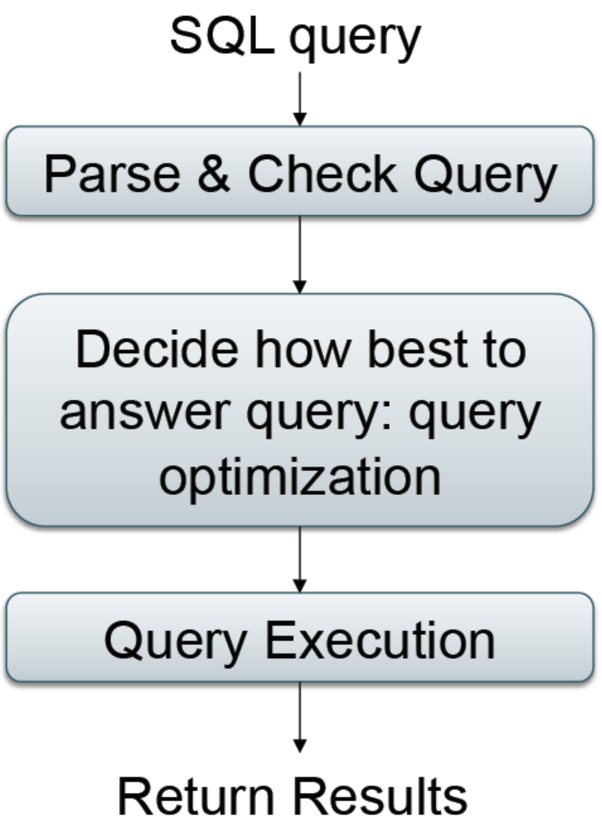
- SQL queries are declarative
  - Users do not specify algorithms and data-structures
  - Logical design and physical design are independent
  - No re-write needed for different users and data
  - DBMS needs to pick an algorithm to answer query
  - Analogy: automatic transmission choosing gear (1, 2, 3, …)

Query Processing and Optimization (QPO)

- Basic idea of QPO
  - In SQL, queries are expressed in high level declarative form (关系代数)
  - QPO translates a SQL query to an execution plan (执行规划)
    - Over physical data model
    - Using operations on file-structures, indices, etc.
  - Goal: reduce run-time of execution plan
    - Answer query in as little time as possible
  - Constraints: QPO overheads are small
    - Computation time for QPO steps << that for execution plan

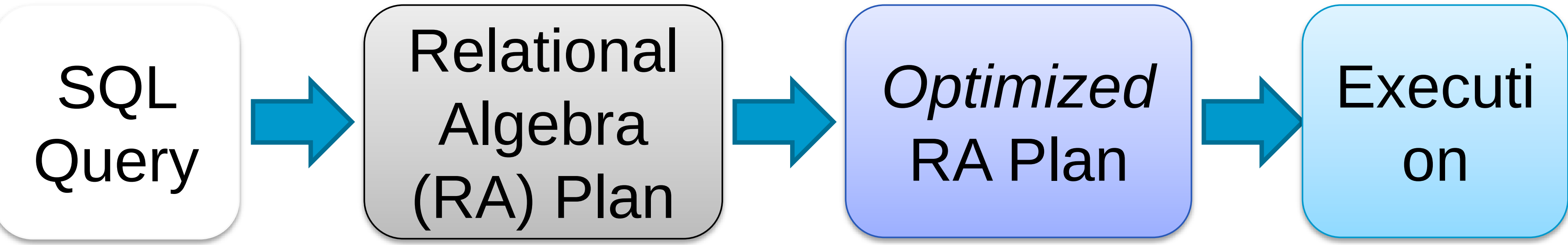
查询处理的步骤

- DBMS接收到SQL查询后，它的查询处理系统要将查询转换为操作代码，一般分为四个步骤
  - 查询分析
  - 查询检查
  - 查询优化
  - 查询执行



查询处理的步骤

- SDBMS architecture
  - How does a SQL engine work?
  - Relational Algebra allows us to translate declarative (SQL) queries into precise and optimizable expressions



Declarative query (from user)

Translate to relational algebra expression

Find logically equivalent- but more efficient- RA expression

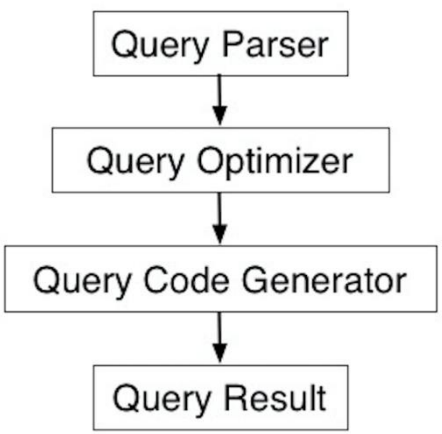
Execute each operator of the optimized plan

查询处理的步骤

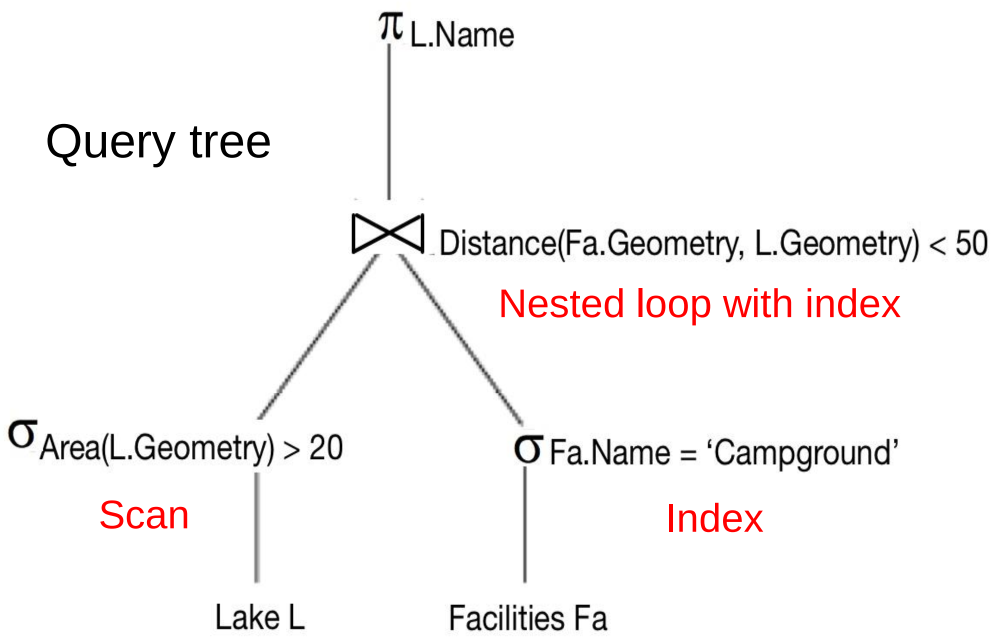
- Query processing and optimization (QPO)
  - Picks algorithms to process a SQL query
- QPO : Physical data model :: automatic transmission : engine

```sql

Select L.Name
From Lake L, Facilities Fa
Where Area (L.Geometry) > 20 And
      Fa.Name = 'campground' And
      Distance(Fa.Geometry, L.Geometry < 50)
```



Query tree



Why Learn About QPO?

- Why learn about automatic transmission in a car?
  - Identify cause of lack of power in a car
    - Is it the engine or the transmission?
  - Solve performance problem with manual override
    - Uphill, downhill driving => lower gears
- Why learn about QPO in a SDBMS?
  - Identify performance bottleneck for a query
    - Is it the physical data model or QPO?
  - How to help QPO speed up processing of a query?
    - Providing hints, rewriting query, etc.
  - How to enhance physical data model to speed up queries?
    - Add indices, change file structures, …

Three Key Concepts in QPO

- 1. Building blocks
  - Most cars have few motions, e.g. forward, reverse
  - Similar most DBMS have few building blocks:
    - Select (point query, range query), join, sort, ...
  - A SQL query is decomposed in building blocks

Point Query	Scan all
	Ordered file structure
	Spatial index
Range Query	Scan all
	Ordered file structure
	Spatial index
Join	Nested loop
	Sort merge
	Hybrid join
	Hash join
	Star join
Spatial Join	Nested loop
	Nested loop with index
	Spatial partitioning-based
	Tree matching

Three Key Concepts in QPO

- 1. Building blocks
- 2. Query processing strategies for building blocks
  - Cars have a few gears for forward motion: 1st, 2nd, 3rd, overdrive
  - DBMS keeps a few processing strategies for each building block
    - e.g. a point query can be answer via an index or via scanning data file
- 3. Query optimization

Three Key Concepts in QPO

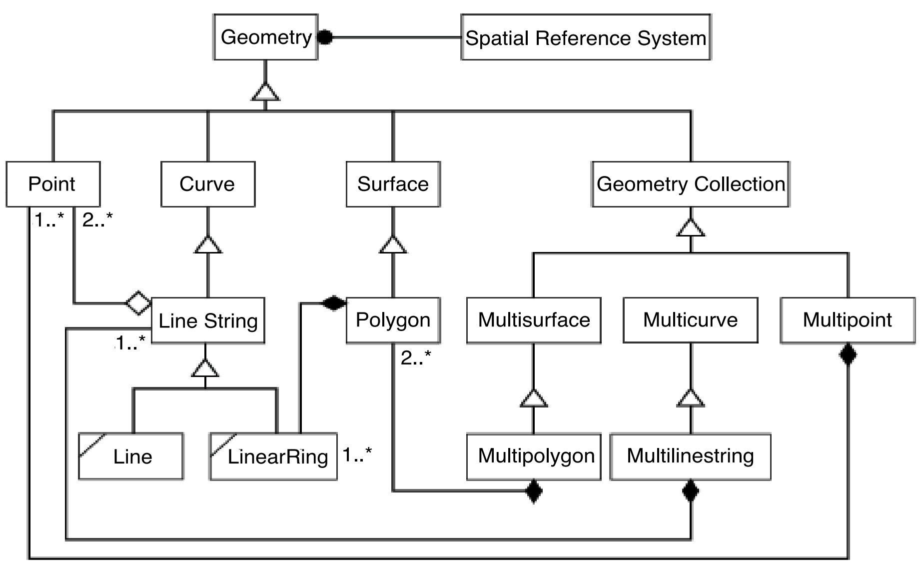
- 1. Building blocks
- 2. Query processing strategies for building blocks
- 3. Query optimization
  - Automatic transmission tries to picks best gear given motion parameters
  - For each building block of a given query, DBMS QPO tries to choose
    - "Most efficient" strategy given database parameters
    - Parameter examples: Table size, available indices, …

QPO Challenges

- Choice of building blocks
  - SQL Queries are based on relational algebra (RA)
  - Building blocks of RA are select, project, join
  - SQL3 adds new building blocks like transitive closure
- Choice of processing strategies for building blocks
  - Constraints: Too many strategies => higher complexity
  - Commercial DBMS have a total of 10 to 30 strategies
    - 2 to 4 strategies for each building block
- How to choose the "best" strategy from among the applicable ones?
  - May use a fixed priority scheme
  - May use a simple cost model based on DBMS parameters

QPO Challenges in SDBMS

- Building Blocks for spatial queries
  - Rich set of spatial data types, operations
  - A consensus on “building blocks” is lacking
  - Current choices include spatial select, spatial join, nearest neighbor



Spatial select:
Return the boundary of Minnesota

Spatial join:
List the countries sharing boundary with Germany

Nearest neighbor:
Find the nearest library from a given apartment

QPO Challenges in SDBMS

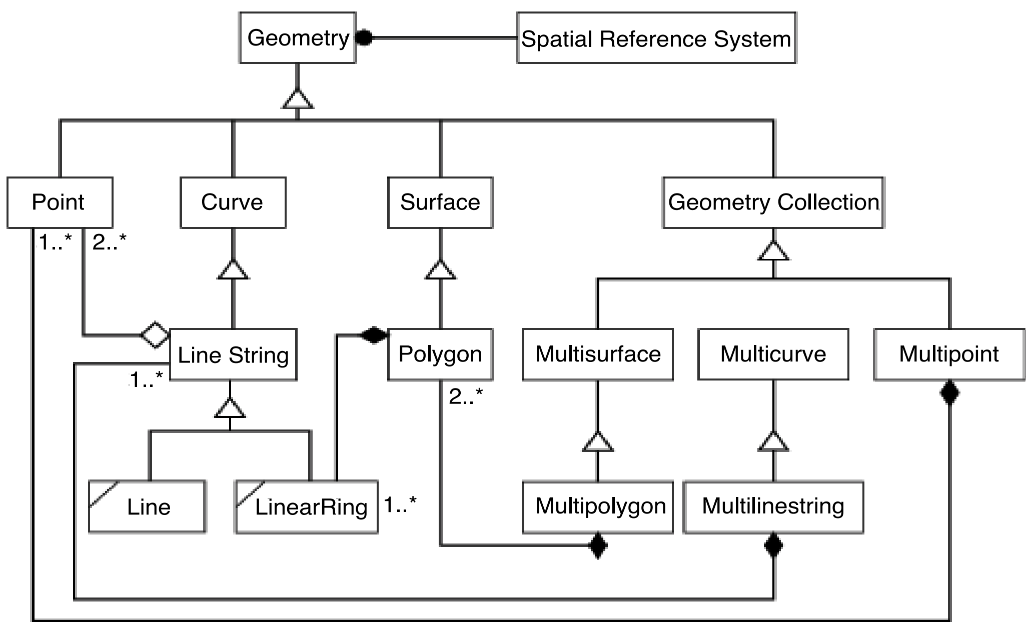
- Choice of strategies
  - Limited choice for some building blocks, e.g. nearest neighbor
- Choosing best strategies
  - Cost models are more complex since
    - Spatial Queries are both CPU and I/O intensive
    - While traditional queries are I/O intensive
  - Cost models of spatial strategies are in not mature

第六章 空间查询处理与优化

- 6.1 查询处理与优化
- 6.2 空间查询基本组件
- 6.3 查询优化
- 6.4 发展趋势 (自学)

Building Blocks for Spatial Queries

- Challenges in choosing building blocks
  - Rich set of data types - point, linestring, polygon, …
  - Rich set of operators - topological, euclidean, set-based, …
  - Large collection of computation geometric algorithms
    - Different spatial operations on different spatial data types
  - Desire to limit complexity of SDBMS



Basic Functions	SpatialReference ()
	Envelop ()
	Export ()
	IsEmpty ()
	IsSimple ()
	Boundary ()
Topological / Set Operators	Equal
	Disjoint
	Intersect
	Touch
	Cross
	Within
	Contains
	Overlap

Simplifying Choices for Building

- Reusing a Geographic Information System (GIS)
  - Which already implements spatial data types and operations
  - However may have difficulties processing large data set on disk
- SDBMS is used as a filter to reduce set of objects to a GIS
- This is filter and refinement approach

Simplifying Choices for Building

- Filter and refinement approach

Which countries are crossed by Nile River?

Brute-force approach:
Traverse all countries in the world and identify the crossed countries.

Filter-and-refine approach

Filter:
Rule out all the countries outside Africa

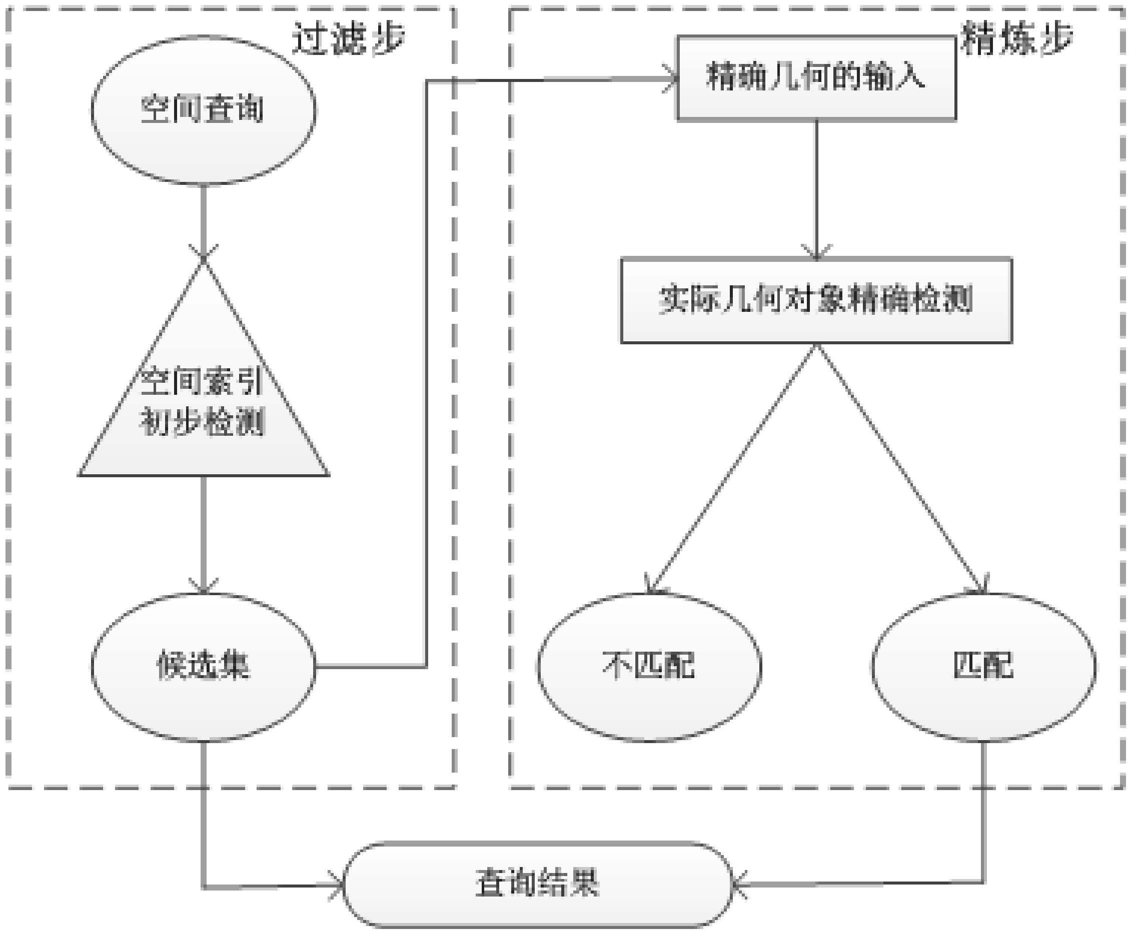
Refine:
Traverse all countries in Africa and identify the crossed countries

空间查询

- 空间数据库中空间查询操作一般分为过滤和精炼两步
  - 过滤步是利用空间对象索引信息以及空间对象的近似形状，检索出可能满足该空间查询条件的对象候选集
  - 精炼步是对候选集中的空间对象按查询要求进行精确的处理计算，以获得满足查询条件的最终结果
- 空间索引主要用于空间查询执行的过滤步

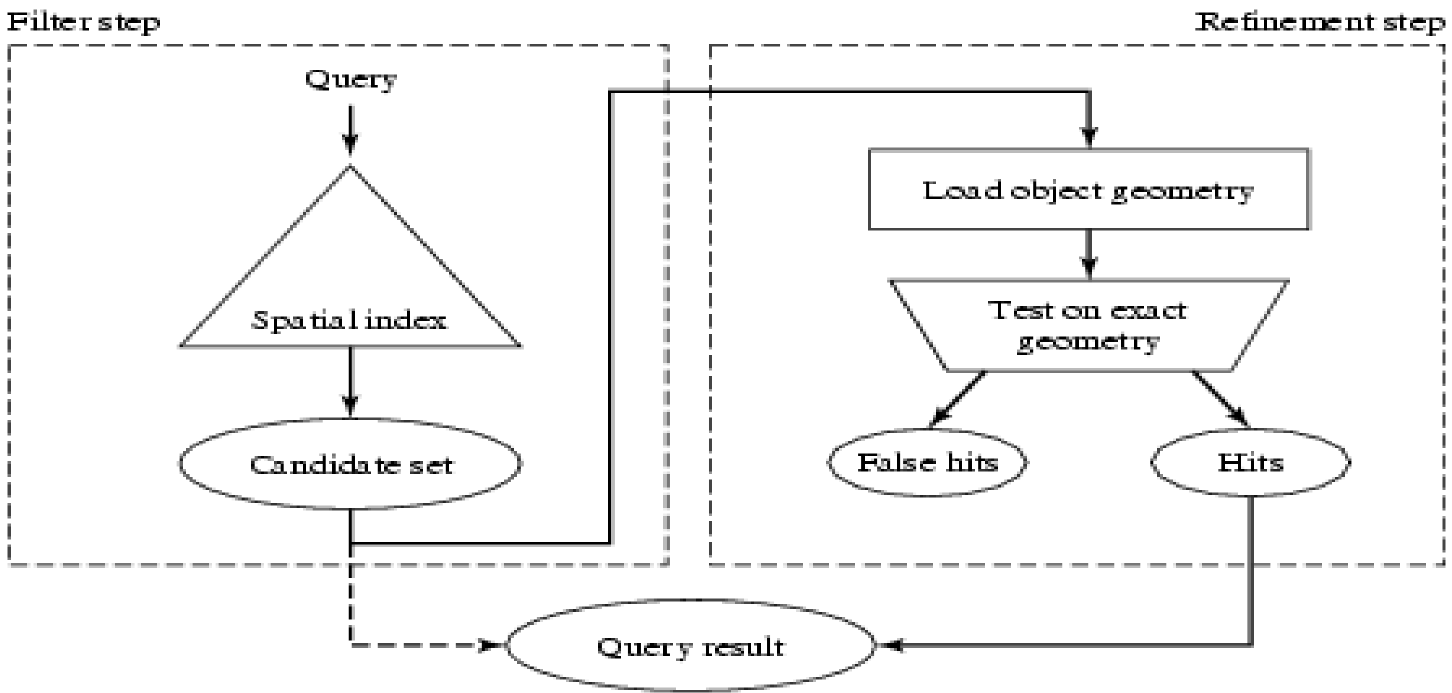
空间查询

- 空间查询一般分为过滤和精炼两步



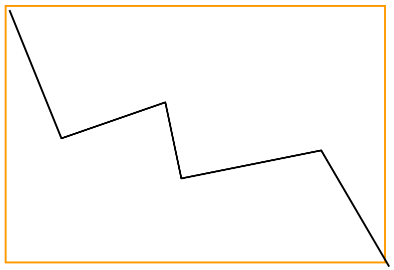
The Filter-Refine Paradigm

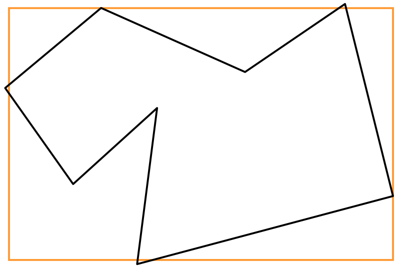
- Processing a spatial query Q
  - Filter step: find a superset S of object in answer to Q
    - Using approximate of spatial data type and operator
  - Refinement step: find exact answer to Q reusing a GIS to process S
    - Using exact spatial data type and operation



Approximate Spatial Data types

- Approximating spatial data types
  - Minimum orthogonal bounding rectangle (MOBR or MBR)
    - Approximates linestring, polygon, …
    - See Examples below (red rectangle are MBRs for black objects)
  - MBRs are used by spatial indexes, e.g. R-tree
  - Algorithms for spatial operations MBRs are simple
- Question: Which OGIS operation returns MBRs ?

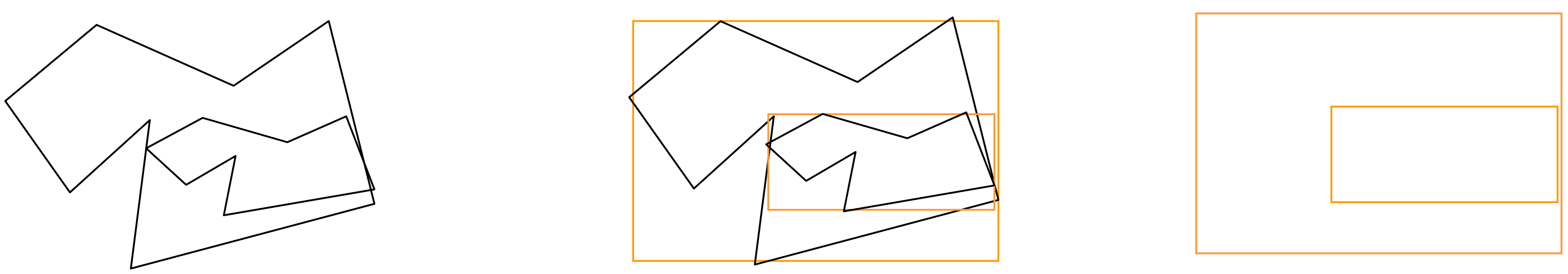




思考：Touches等拓扑操作的cost为100， MBRs获取的cost为1，
MBRs之间拓扑操作的cost？

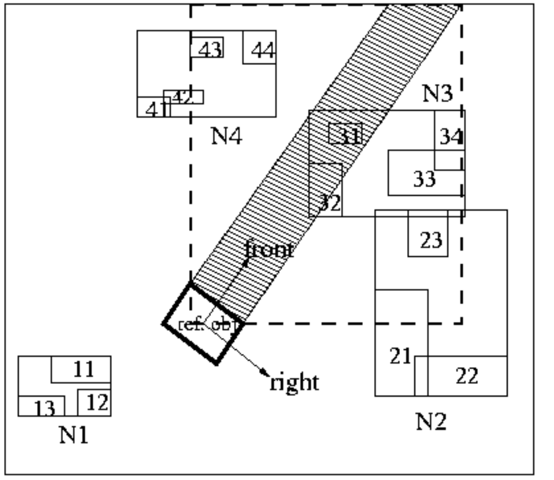
Approximate Spatial Operations

- Approximating spatial operations
  - SDBMS processes MBRs for refinement step
  - Overlap predicate used to approximate topological operations
  - Example: inside(A, B) replaced by
    - overlap(MBR(A), MBR(B)) in the filter step
    - Let A be outer polygon and B be the inner one
    - inside(A, B) is true only if overlap(MBR(A), MBR(B))
    - However overlap is only a filter for inside predicate needing refinement later

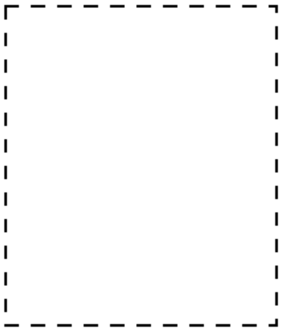


Filter Step Example 1

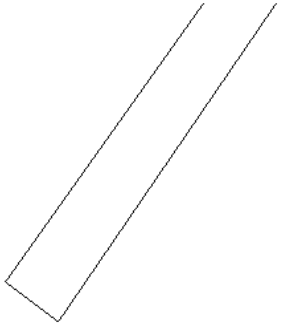
- Query:
  - List objects in front of a viewer V
- Equivalent overlap query
  - Direction region is a polygon
  - List objects overlapping with
    - polygon(front(V))
- Approximate query
  - List objects overlapping with
    - MBR(polygon (front (V)))



(a)



(b) Range Query

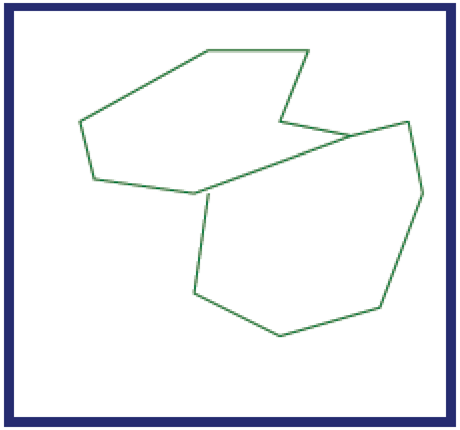


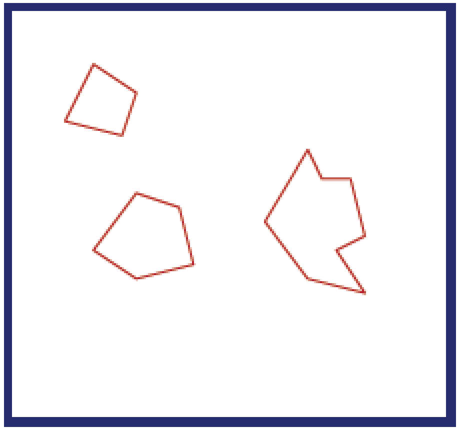
(c) Direction region

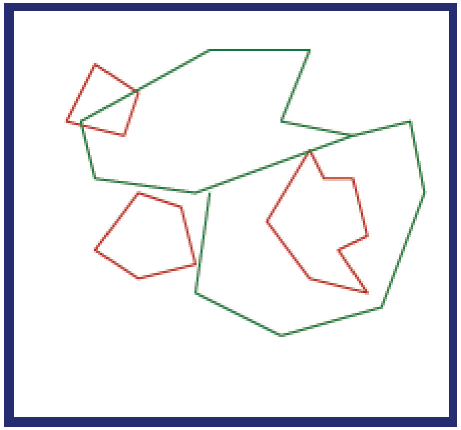
Filter Step Example 2

- Spatial joins: find (quickly) all

counties intersecting lakes

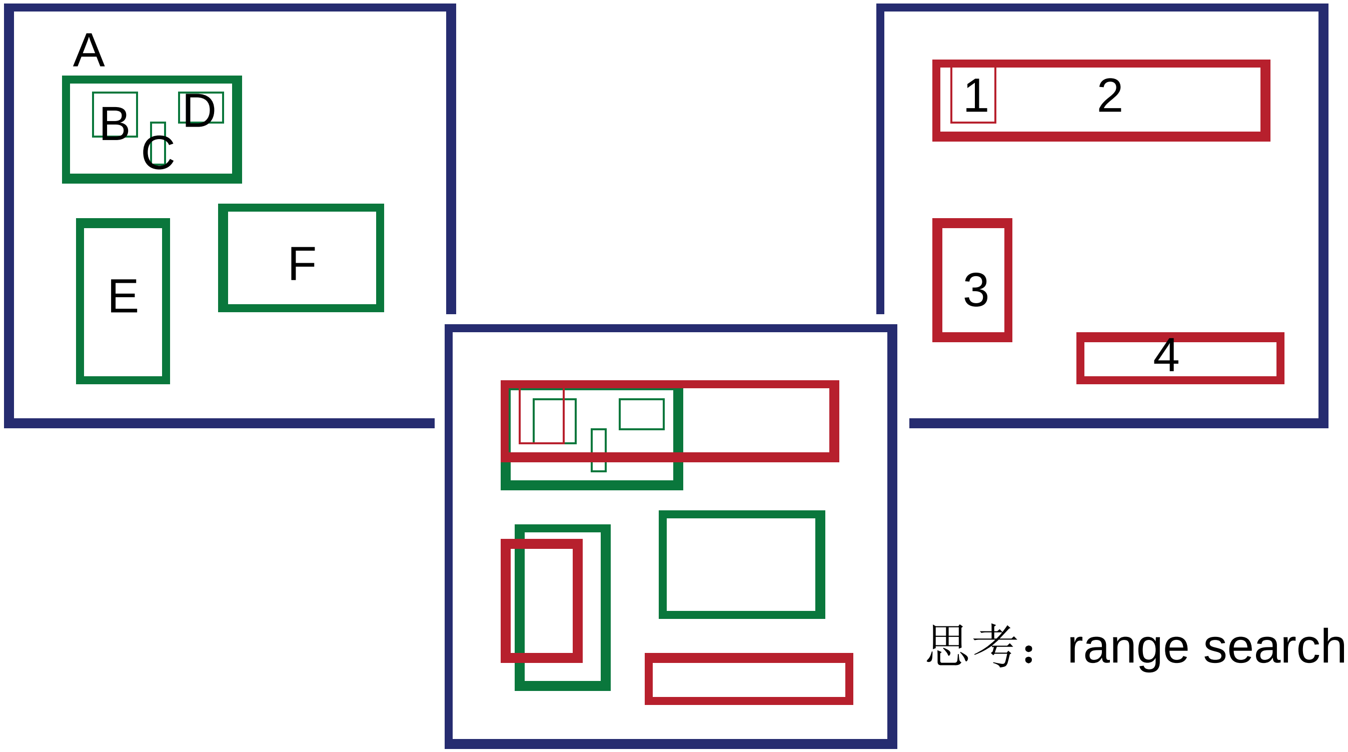






Filter Step Example 2

- Assume that they are both organized in R-trees:
  - 不利用R-trees：6*4 = 24次复杂几何的空间关系判断
  - 利用R-trees进行filter: (A, 1), (A, 2), (B, 1), (B, 2), (C, 2), (D, 2), (E, 3), (F, 4)



思考：range search的filter step

Choice of Building Blocks

- Choice of building blocks
  - Varies across software vendors and products
  - Representative building blocks are listed here
- List of building blocks
  - Point Query - Name a highlighted city on a digital map
    - Return one spatial object out of a table
  - Range Query - List all countries crossed by of the river Amazon
    - Returns several objects within a spatial region from a table
  - Nearest Neighbor - Find the city closest to Mount Everest
    - Return one spatial object from a collection
  - Spatial Join - List all pairs of overlapping rivers and countries
    - Return pairs from 2 tables satisfying a spatial predicate

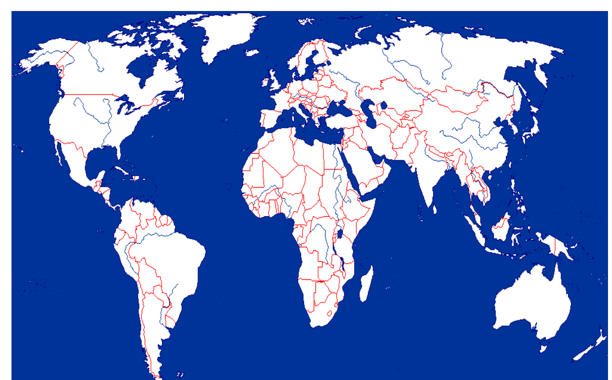
Choice of Building Blocks

Name the highlighted city

List countries crossed by Amazon River

Find the city closest to Chicago

List all pairs of overlapping rivers and countries

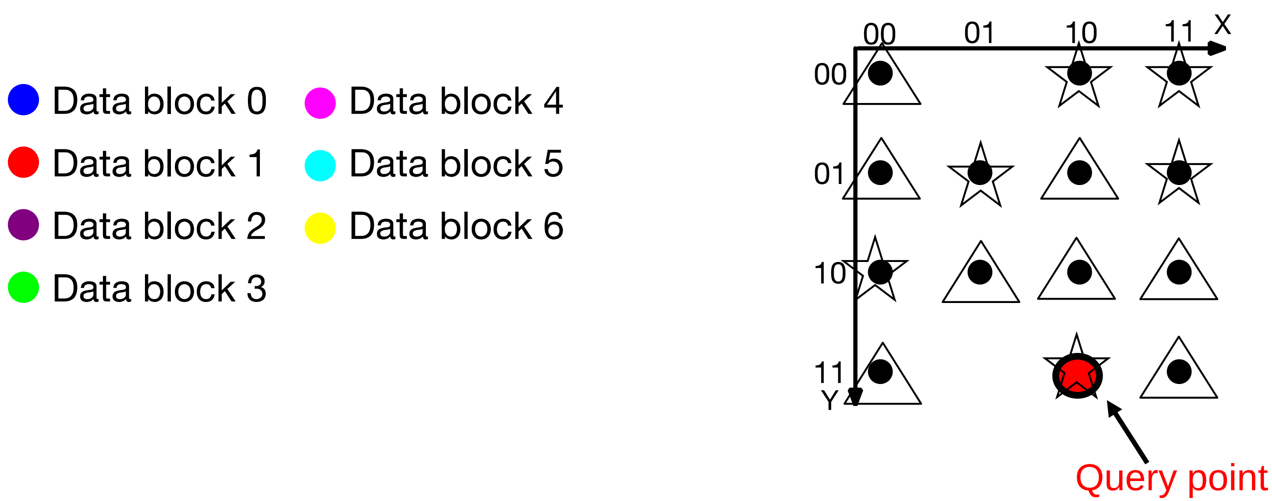


Strategies for Point Queries

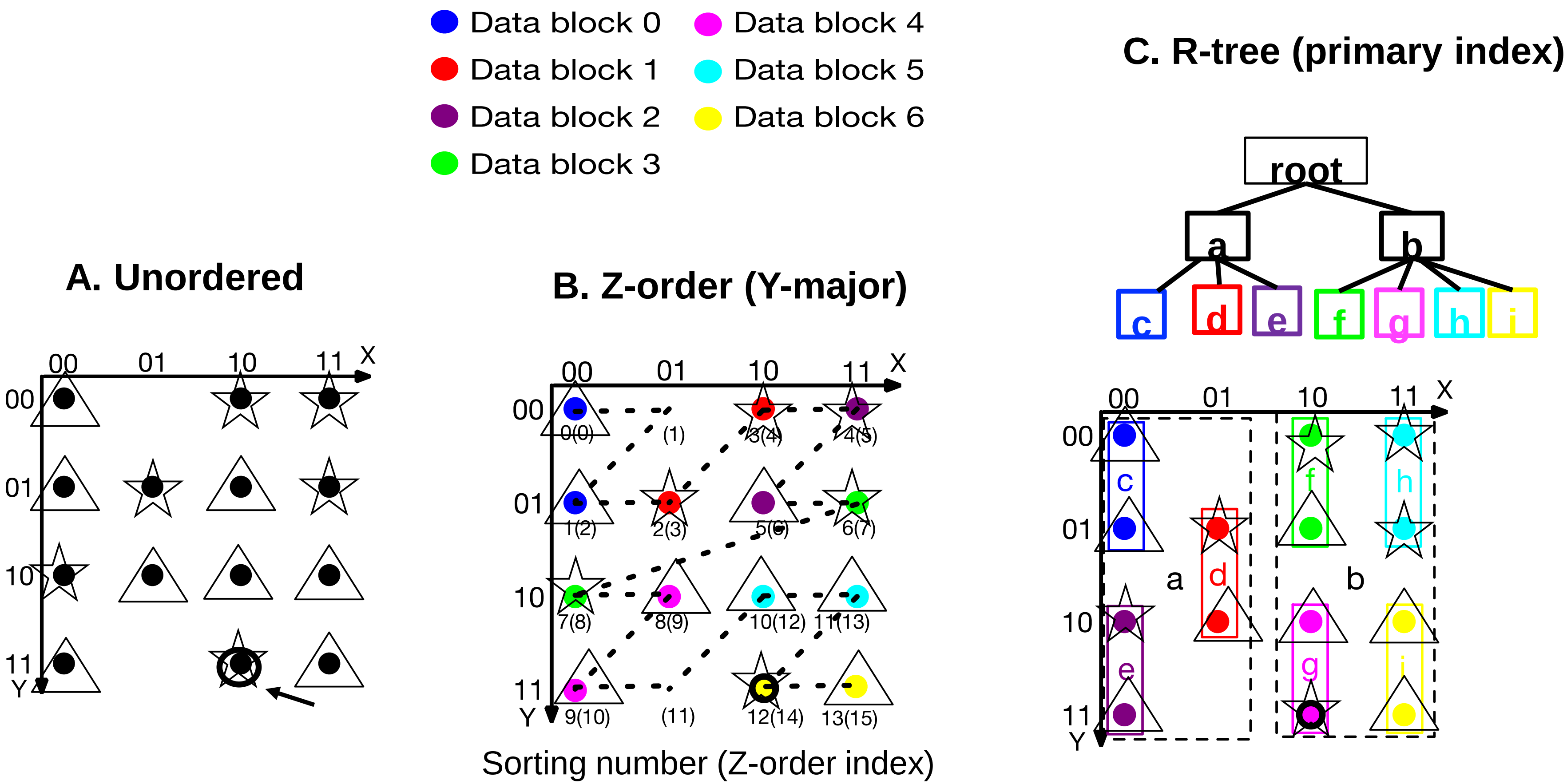
- Recall Point Query Example
  - Given a location
  - Return a property (e.g., place name) of the location
- List of strategies
  - Scan all B disk sectors of the data file
  - If records are ordered using space filling curve (say Z-Curve)
    - Then use binary search on the Z-Curve of search point to examine about log(B, base = 2) disk sectors
  - If an index is available on spatial location of data objects,
    - Then use find() operation on the index
    - Number of disk sector examined = depth of index (typically 4 to 5)

Point Query Example

- Data: 14 points, each with a type triangle or star
- Query: Return type at location (x, y) = (10, 11)
- Candidate Storage Methods
  - 7 data blocks, each with 2 points



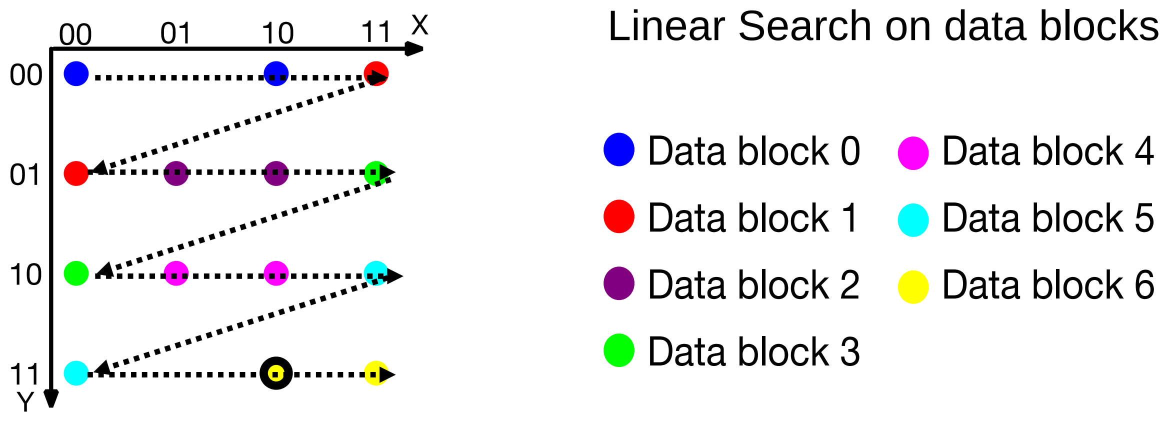
Candidate Storage & Indexing Methods



Linear Search for Point Query

- Data: 14 points, each with a type triangle or star
- (Marked) Query: Return the type of crime in the location (x, y) = (2, 3)
- Storage Methods: Unordered

Cost for linear search on this dataset: 7



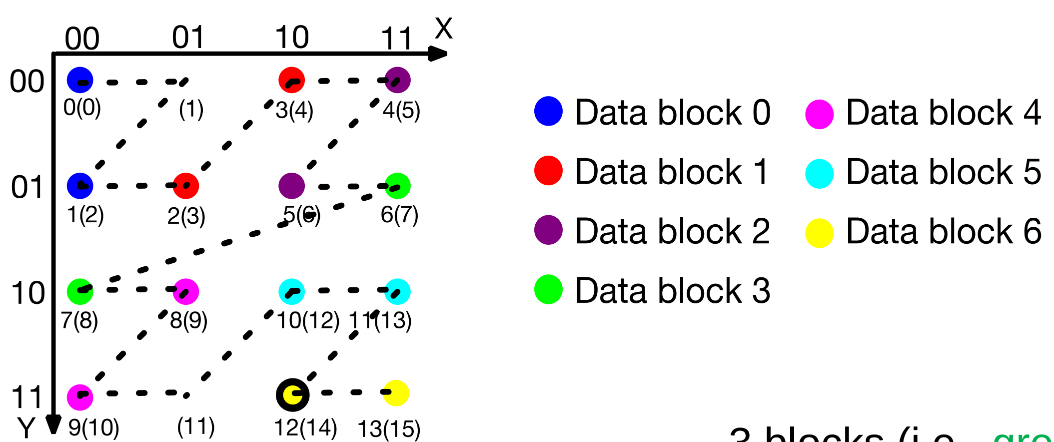
Binary Search for Point Query

- Data: 14 points, each with a type triangle or star
- (Marked) Query: Return the type of crime in the location (x, y) = (2, 3)
- Storage Methods: Z-order (Y-major)

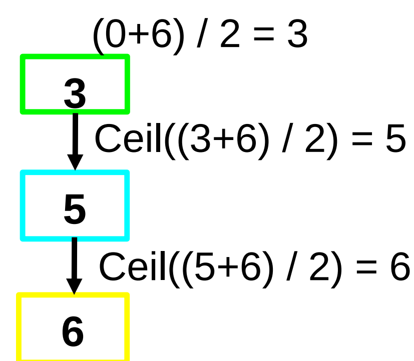
Block序号和Z-Value存在非线性关系，对Z-value构建索引，如B+树，需要访问哪些Blocks？

Cost for linear search on this dataset: 3

Y-major



Binary search on data blocks 0 .. 6

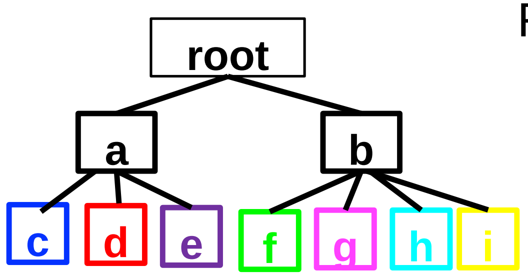


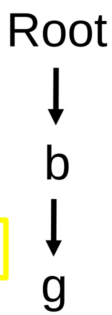
3 blocks (i.e., green, cyan, yellow) accessed

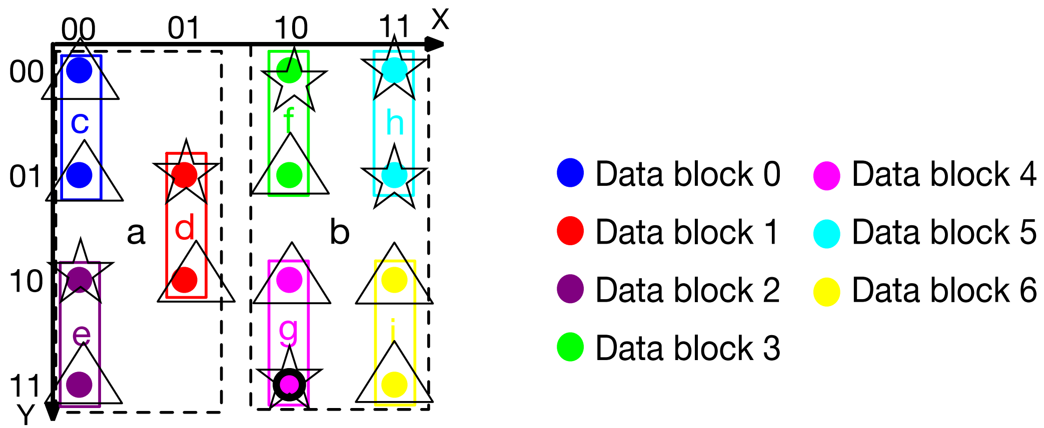
Search for Point Query Using R-Tree

- Data: 14 points, each with a type triangle or star
- (Marked) Query: Return the type of crime in the location (x, y) = (2, 3)
- Storage Methods: R-Tree (Primary Index), root cached in main meory

In this example	
Index block	1
Data block	1

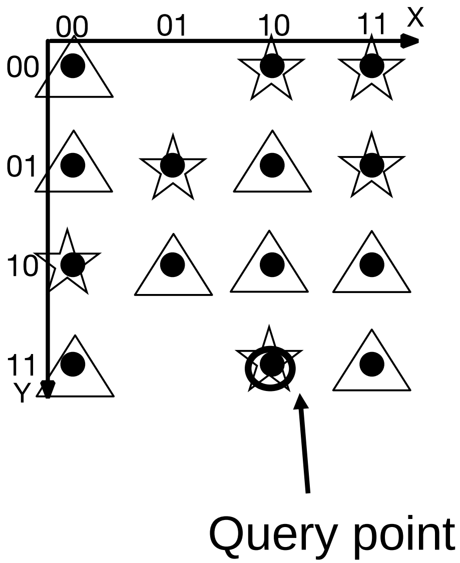






Comparing 3 Strategies for Point Query

- Data: 14 points, each with a type triangle or star
- (Marked) Query: Return the type of crime in the location (x, y) = (2, 3)



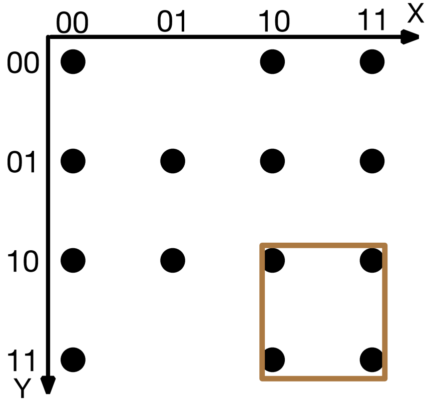
Storage Method	In this example	
Linear Search	7	
Binary Search	3	
Index Search	Index blocks	1
	Data Blocks	1

Strategies for Range Queries

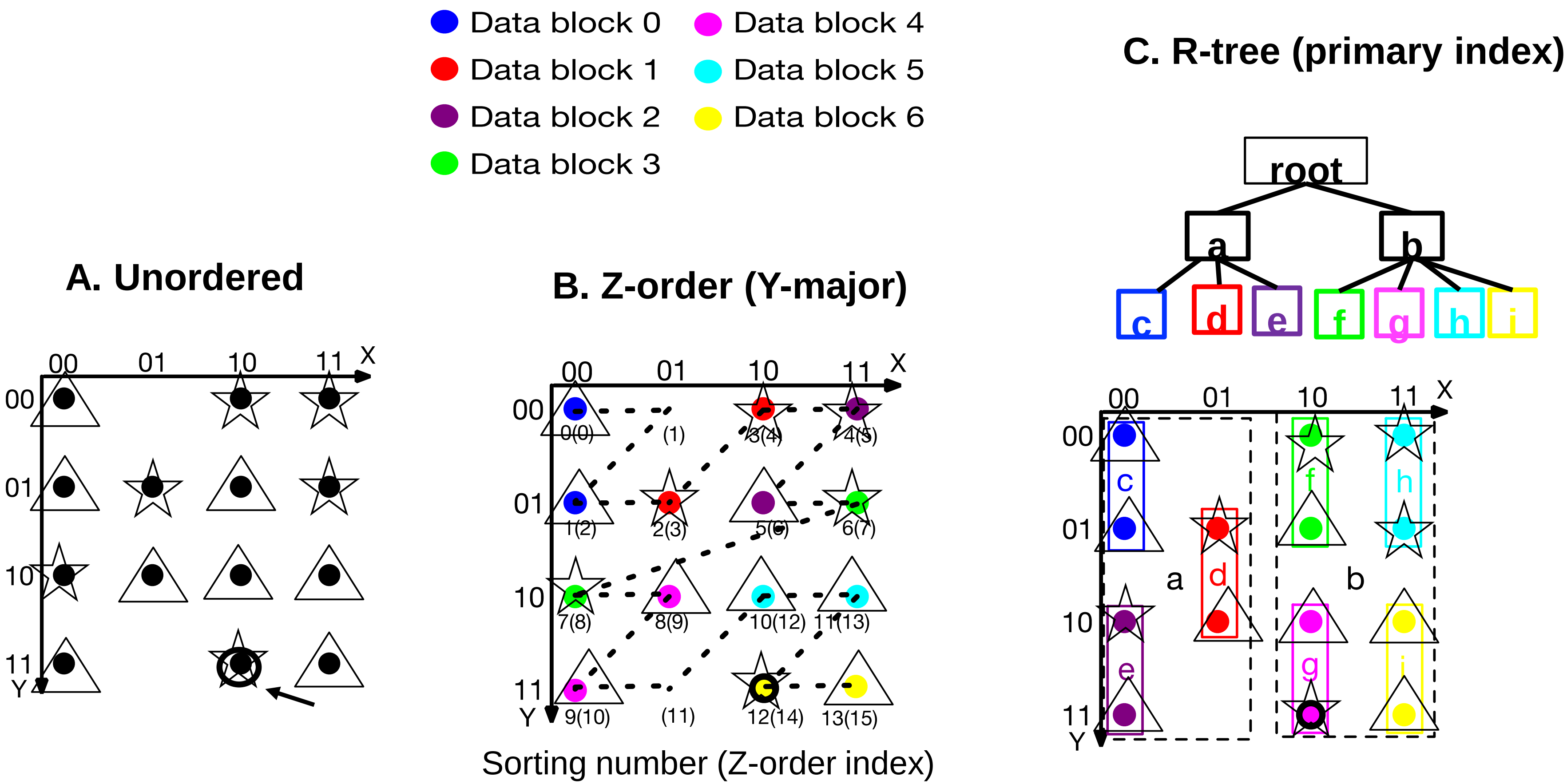
- Recall Range Query Example
  - List all countries crossed by of the river Amazon
    - Returns several objects within a spatial region from a table
- List of strategies
  - Scan all B disk sectors of the data file
  - If records are ordered using space filling curve (say Z-order)
    - Then determine range of Z-order values satisfying range query
    - Use binary search to get lowest Z-order within query answer
    - Scan forward in the data file till the highest z-order satisfying query
  - If an index is available on spatial location of data objects
    - then use range-query operation on the index

Range Query Example

- Data: 14 points, each with a type triangle or star
- (Brown Box) Query: (2 <= x <= 3) and (2 <= y <= 3)
- Storage Methods:
  - 7 data block with 2 points each
  - Unordered, Z-ordered, R-tree



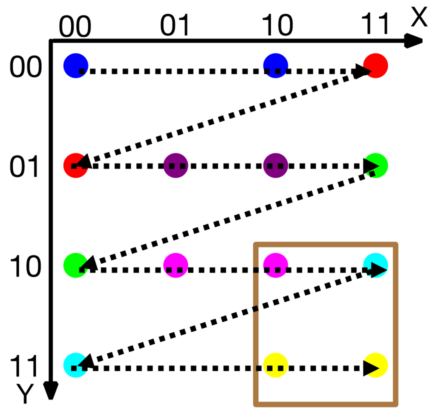
Candidate Storage & Indexing Methods



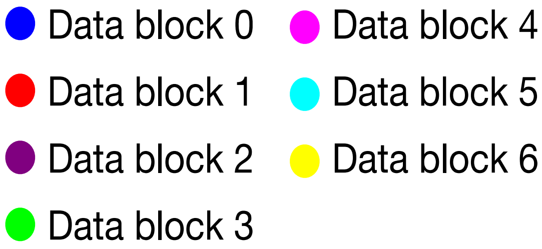
Linear Search for Range Query

- Data: 14 points, each with a type triangle or star
- (Brown Box) Query: (2 <= x <= 3) and (2 <= y <= 3)
- Storage Method: Unordered

Cost for linear search on this dataset: 7



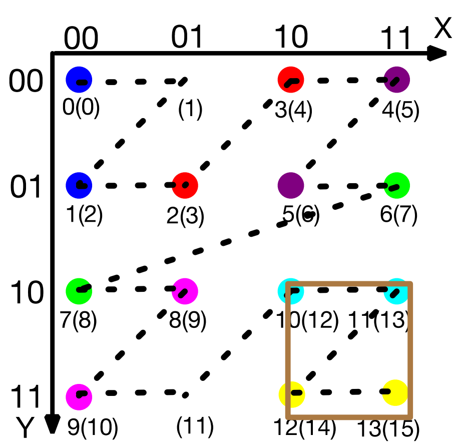
Linear search on data blocks 0 ..6

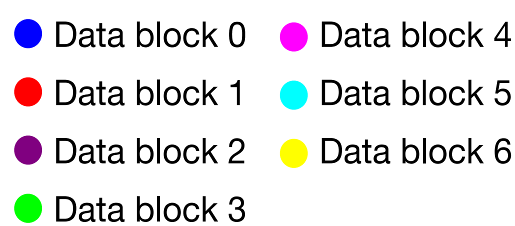


Binary Search for Range Query

- Data: 14 points, each with a type triangle or star
- (Brown Box) Query: (2 <= x <= 3) and (2 <= y <= 3)
- Storage Method: Z-order
  - One Z-interval 12..15 ➔ search for 12 then scan forward

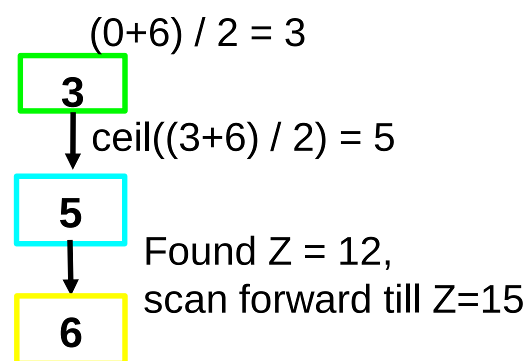
3 blocks (i.e., green, cyan, yellow) accessed





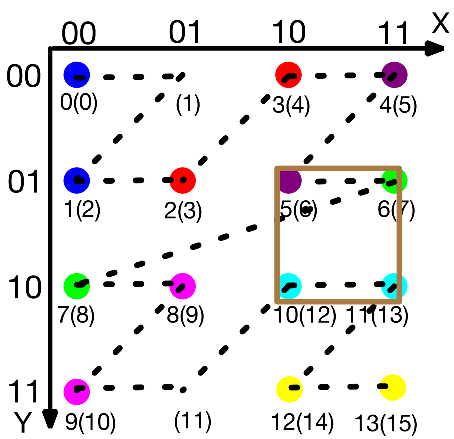
3和5是查找时获取的Block，5和6是读取12..15时获取的Block

Binary search
on data blocks 0..6



Range Query with Two Z-intervals

- Data: 14 points, each with a type triangle or star
- (Brown Box) Query: (2 <= x <= 3) and (1 <= y <= 2)
  - Two Z-intervals: [5..6] and [12..13]
  - One binary search (followed by scan) for each Z-interval



- Data block 0
- Data block 1
- Data block 2
- Data block 3
- Data block 4
- Data block 5
- Data block 6

3 blocks
(i.e., green, purple, cyan)
accessed

Binary search to find [(6), (7)]

3 (0+6) / 2 = 3

2

Found Z=5
Scan till Z= 6

Binary search to find [(12), (13)]

(0+6) / 2 = 3

3

ceil((3+6) / 2) = 5

5

Found Z=10
Scan till Z= 11

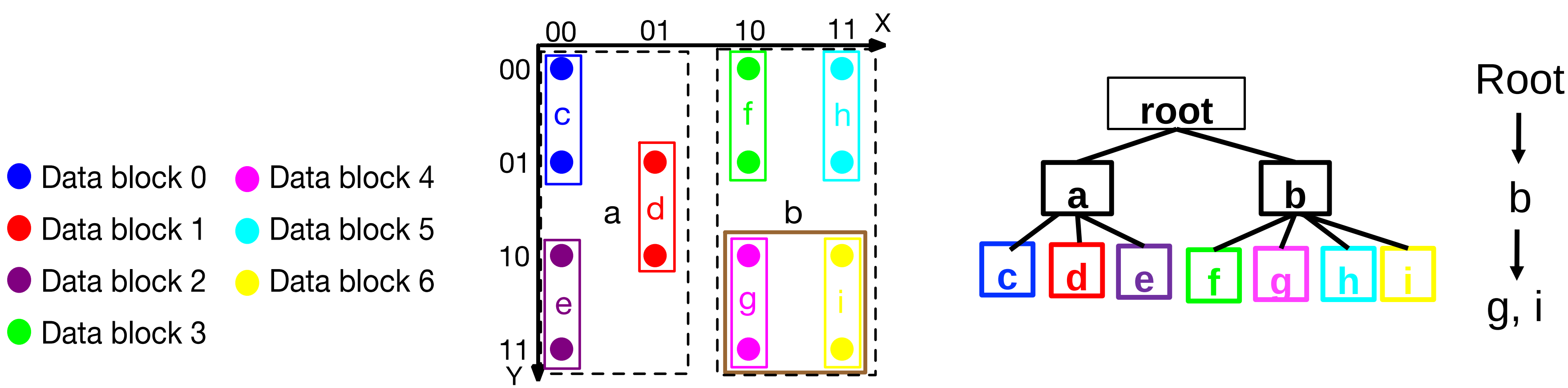
这里假设内存或Cache比较大，如果内存较小，Block 3可能需要重复读取

是否需要读Block 6？

Search for Range Query Using R-Tree

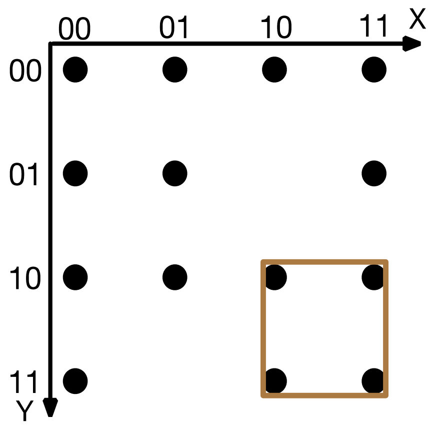
- Data: 14 points, each with a type triangle or star
- (Brown Box) Query: (2 <= x <= 3) and (2 <= y <= 3)
- Storage Method: R-Tree (Primary Index)

Cost in this example	
Index block	1
Data block	2



Comparing Algorithms for Range Query

- Data: 14 points, each with a type triangle or star
- (Brown Box) Query: (2 <= x <= 3) and (2 <= y <= 3)



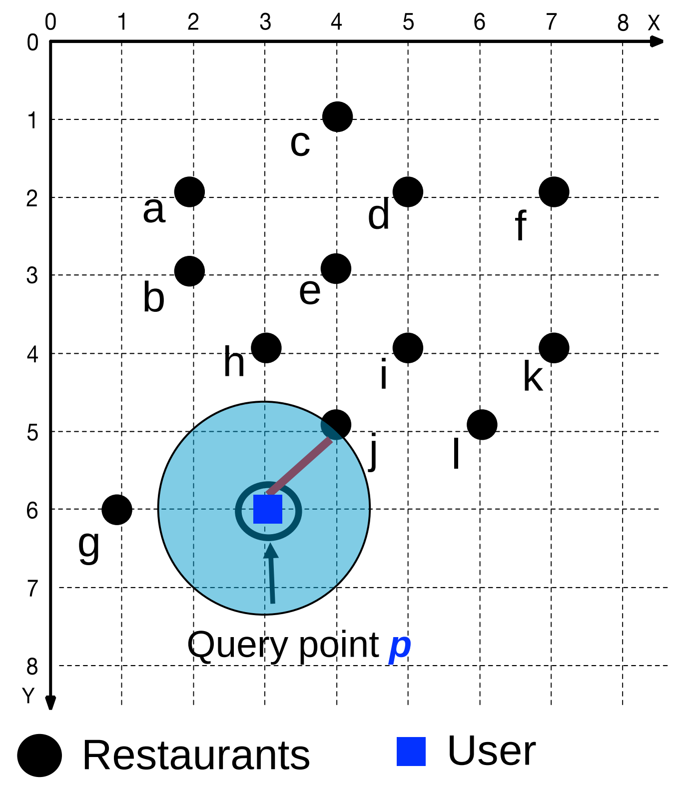
Storage Method	In this example	
Linear Search	7	
Binary Searches	3	
Index Search	Index blocks	1
	Data Blocks	2

Strategies for Nearest Neighbor Queries

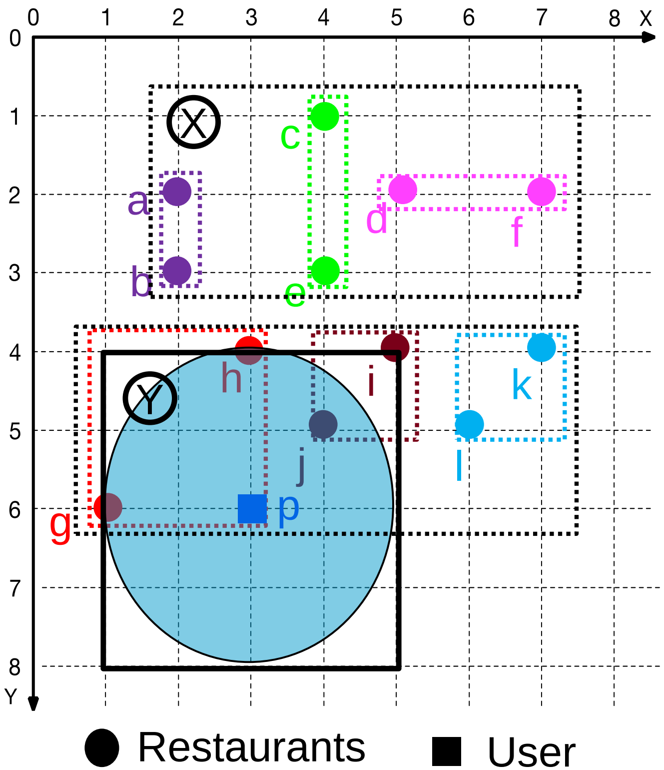
- Recall Nearest Neighbor Example
  - Find the city closest to Mount Everest
    - Return one spatial object from city data file C
- List of strategies
  - Two phase approach
    - Fetch C’s disk sector(s) containing the location of Mt. Everest
    - M = minimum distance(Mt. Everest, cities in fetched sectors)
    - Test all cities within distance M of Mt. Everest (Range Query)
  - Single phase approach
    - Recursive algorithm for R-tree
    - Eliminate children dominated by some other children
    - Check the remaining data blocks for nearest neighbor

Nearest Neighbor Example

- Each point represents location of a restaurant
- Query: Given the location of a user p, find the nearest restaurant (if more than one nearest neighbors, return all results)
- Result
  - Nearest neighbor of p is j



Two Phase Strategy (with a R-Tree)



Find the index leaf containing the query point p: block red

In red leaf, Point g, h are the closest points to p, $d_B = 2$

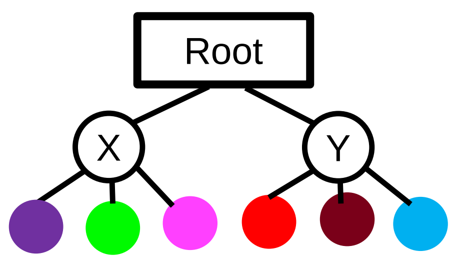
Create a circle Circle_p whose center is p, and radius = d_B

Create the MOBR of Circle_p : M_p

Range query: M_p, and test all points in M_p

Root -> Y -> Block brown

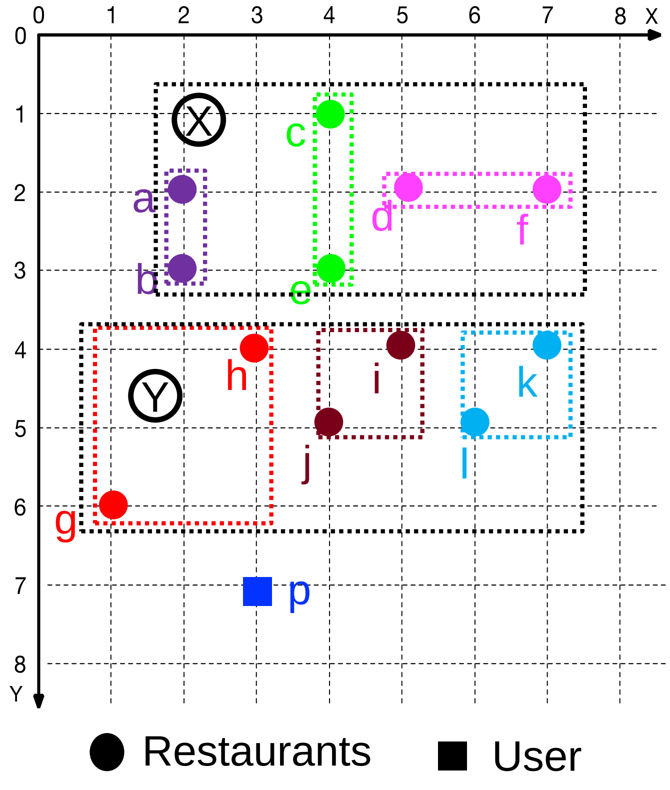
Since dist(p, j) = 1.41 < d_B, point j is nearest neighbor of p

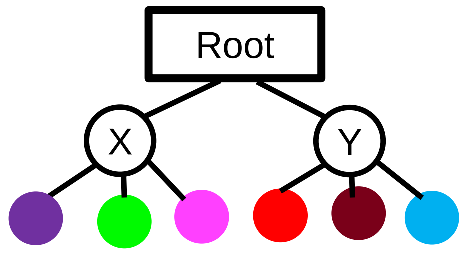


Cost:	Index blocks	Data blocks
Phase 1	1	1
Phase 2	1	1

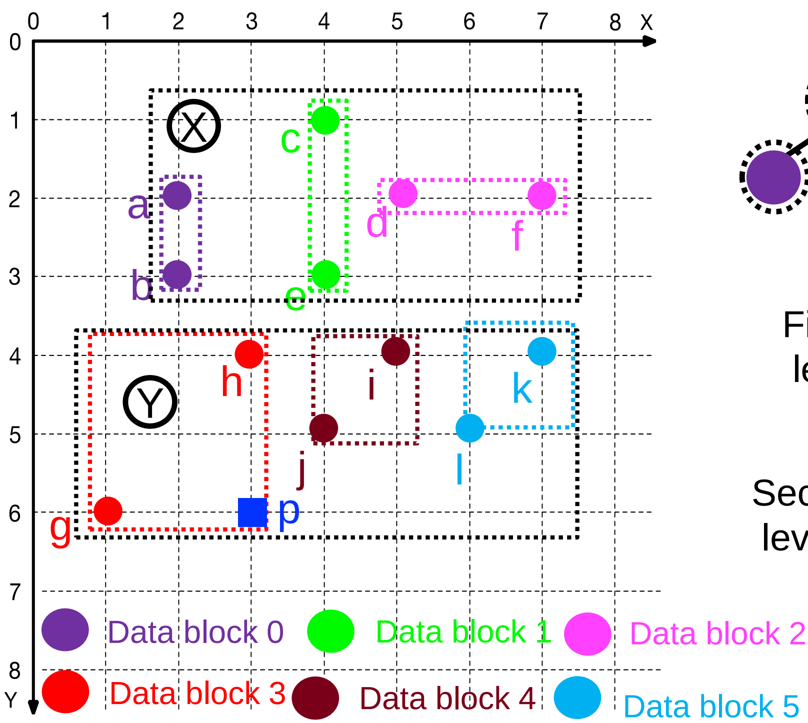
Two Phase Strategy (with a R-Tree)

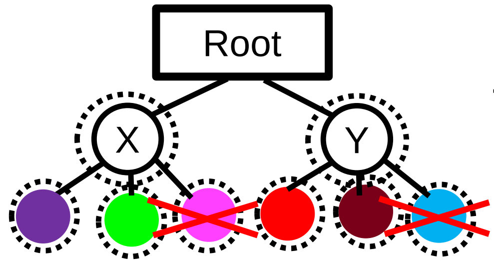
- Exercise: Generalize the algorithm to the case when query point is outside bounding box of root of the R-Tree?





One Phase Strategy (with a R-Tree)





Finally, check blocks 0, 1, 3, 4 for nearest neighbors

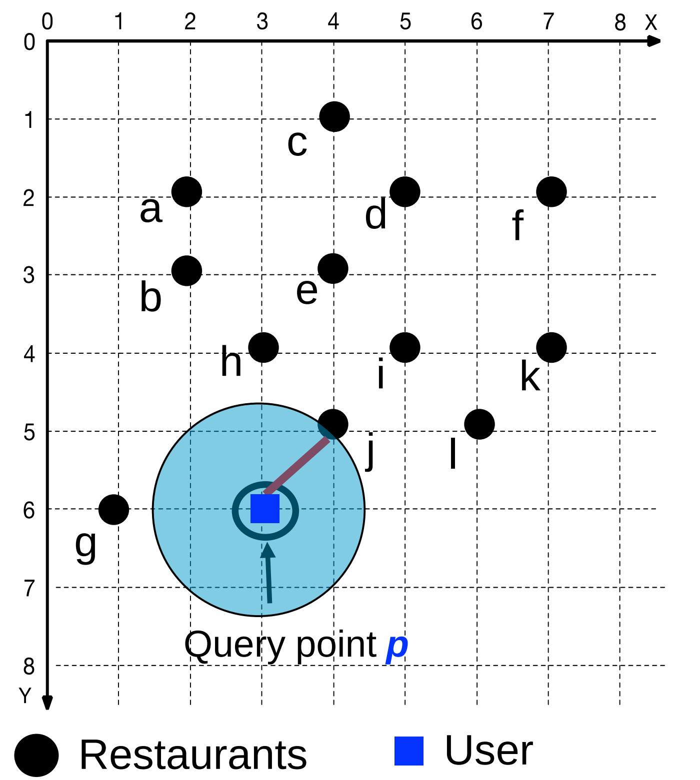
Index blocks	Data blocks
2	4

	Node	MinDist	MaxDist	
First level:	X	3	7.47	
	Y	0	4.47	Nothing eliminated
Second level:	0	3.16	4.12	
	1	3.16	5.10	
	2	4.47		Node 2 eliminated
	3	0	2.83	
	4	1.41	2.83	
	5	3.16		Node 5 eliminated

Comparing Algorithms for Nearest Neighbor Queries

- Each point represents location of a restaurant
- Query: Given the location of a user p, find the nearest restaurant (if more than one nearest neighbors, return all results)
- Result
  - Nearest neighbor of p is j

Storage Method	In this example	
Two phase approach	Index blocks	2
	Data Blocks	2
One phase approach	Index blocks	2
	Data Blocks	4



Strategies for Spatial Joins

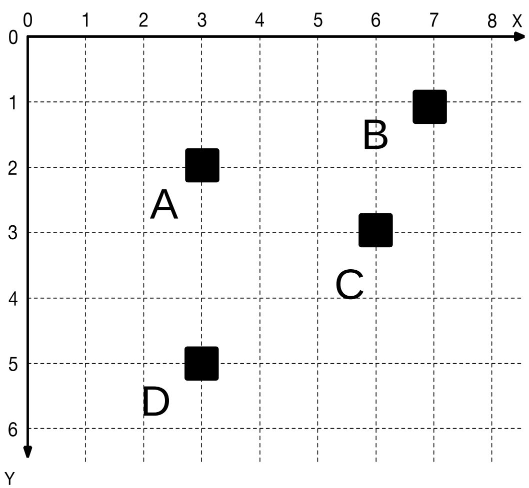
- Recall Spatial Join Example
  - List all pairs of overlapping rivers and countries
    - Return pairs from 2 tables satisfying a spatial predicate
- List of strategies
  - Nested loop
    - Test all possible pairs for spatial predicate
    - All rivers are paired with all countries
  - Space Partitioning
    - Test pairs of objects from common spatial regions only
    - Rivers in Africa are tested with countries in Africa only
  - Tree Matching
    - Hierarchical pairing of object groups from each table
  - Other, e.g. spatial-join-index based, external plane-sweep, …

Sptial Join Example

- Query: For each fire station, find all the houses within a distance <= 1

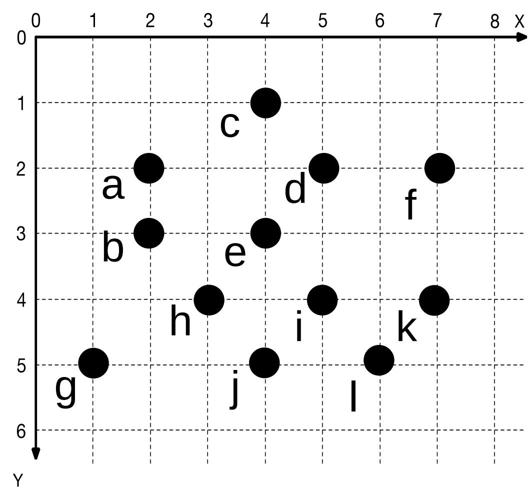
Fire-station	House
A	a
B	f
D	h
D	j

Fire station map



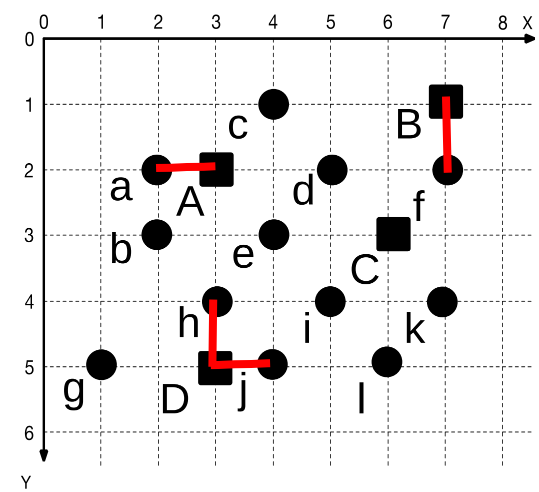
Fire-stations

House map



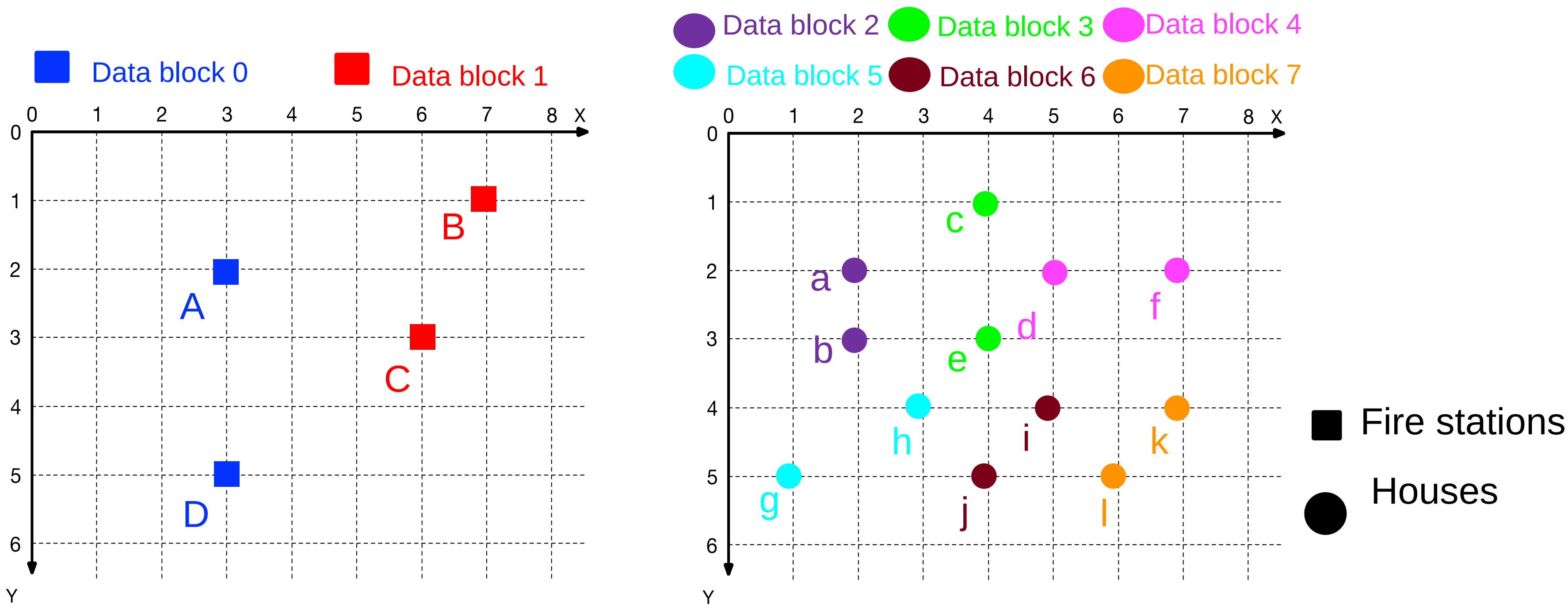
Houses

Overlay



Storage Structure

- 2 blocks for fire stations
- 6 blocks for houses



Nested Loop

- Nested loop
  - Test all possible pairs for spatial predicate
  - Outer loop: bring data blocks of first table in memory
  - Inner loop: scan the second table

Data block 0 Data block 1

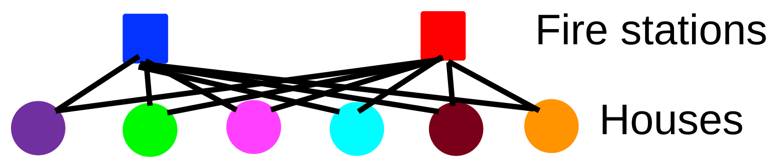
Data block 2 Data block 3 Data block 4

Data block 5 Data block 6 Data block 7

Algorithm: For each block B_{fs} of fire stations

For each block B_h of houses

Scan all pairs of fire stations in B_{fs} and houses in B_h



Cost: For Blue block, inner loop fetches all 6 (circle) blocks

For Red block, inner loop fetches all 6 (circle) blocks

blocks for fire stations * # blocks for houses = 2*6 = 12

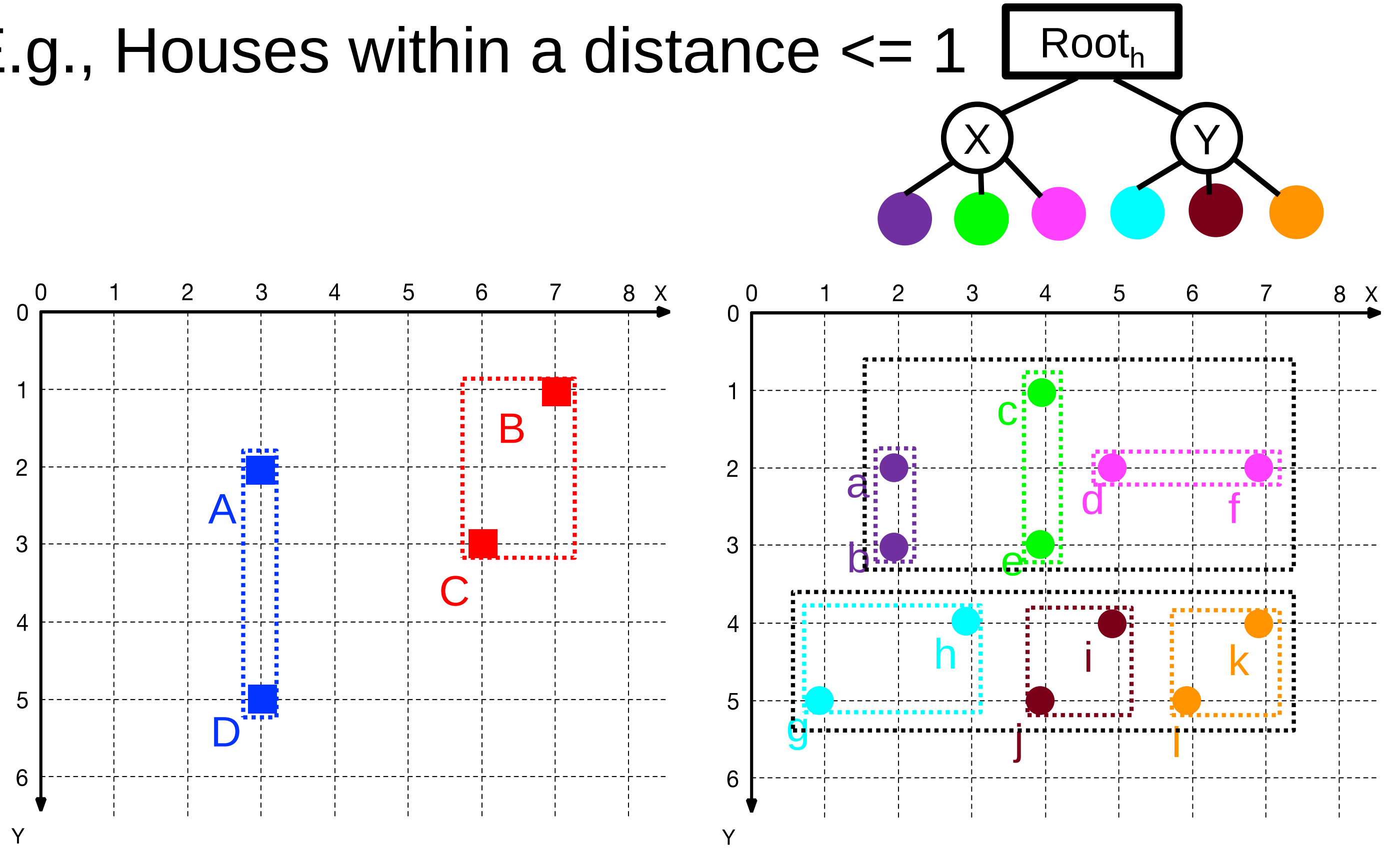
总的Block = 2 + 12
Outer loop数据有关

Assume: 3 memory buffers

(i.e., 1 for fire-stations, 1 for houses, 1 for results)

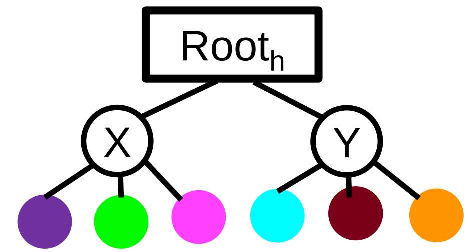
Nested Loop with Spatial Index

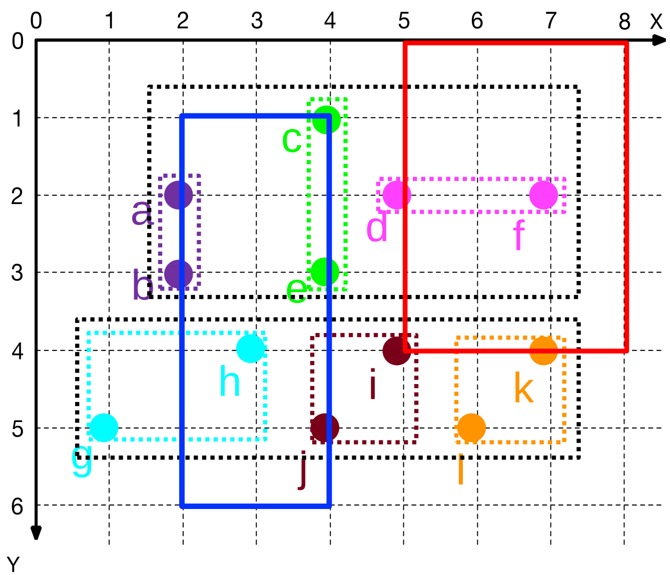
- Outer loop: For each data blocks D of first table
- Inner loop: Range Query second table for overlapping block
  - E.g., Houses within a distance <= 1

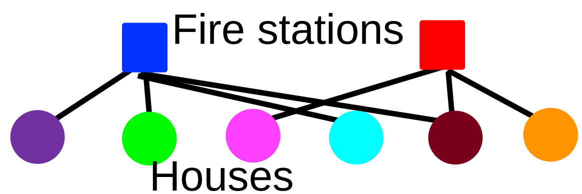


Nested Loop with Spatial Index

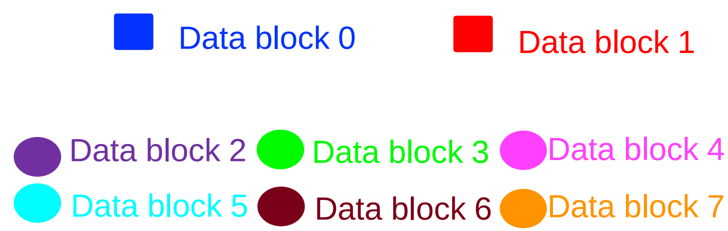
- Outer loop: For each data blocks D of first table
- Inner loop: Range Query second table for overlapping block
  - E.g., Houses within a distance <= 1



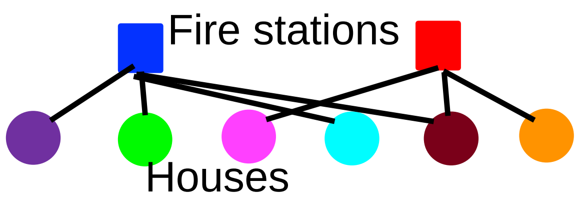


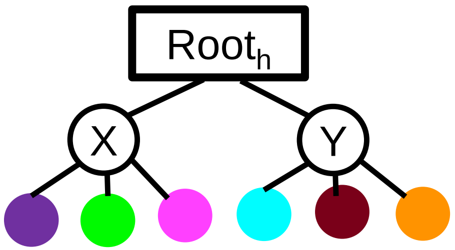


Outer	Inner blocks
0	2, 3, 5, 6
1	4, 6, 7



Nested Loop with Spatial Index





Block 0: Root -> X -> 2, 3
-> Y -> 5, 6

Block 1: Root -> X -> 4
-> Y -> 6, 7

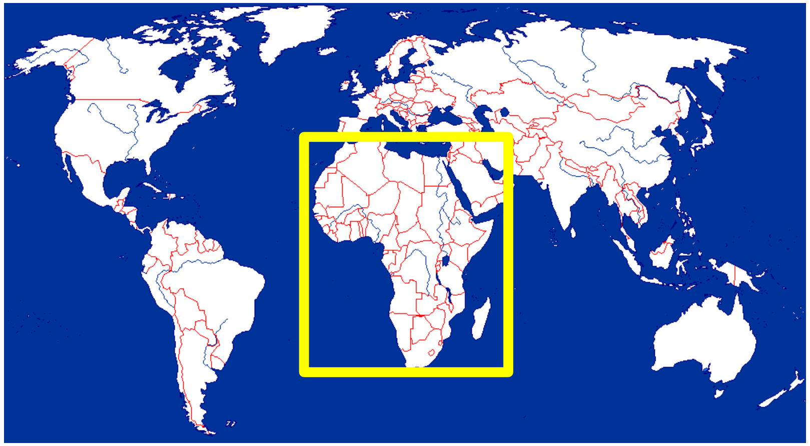
Index blocks	Data blocks	
2 + 2 = 4	FS	2
	House	4+3=7
	Total	9

Data block 0 Data block 1

Data block 2 Data block 3 Data block 4
Data block 5 Data block 6 Data block 7

Space Partitioning Join

- Example Query: Pair rivers with countries they pass through.
  - Do we need to test Nile river with countries outside Africa?
- Space Partitioning Idea
  - Rivers in Africa are tested with countries in Africa only
  - Test pairs of objects within common spatial regions

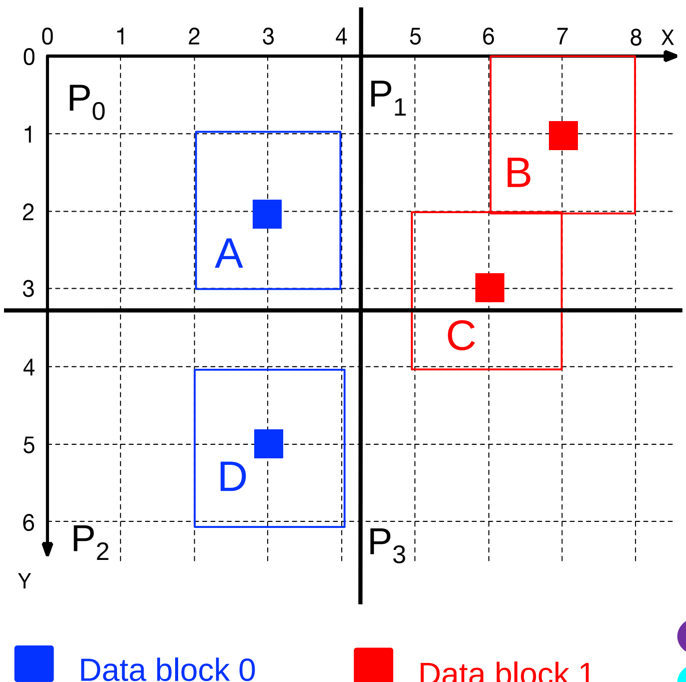


Common Space Partitioning

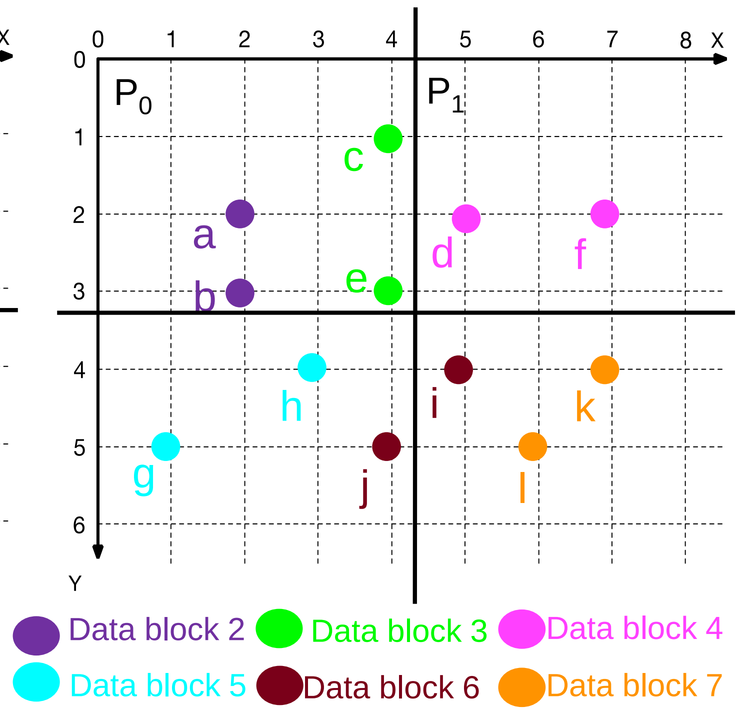
- Query: For each fire station, find houses within distance <= 1

Four Partition: P_0, P_1, P_2, P_3

For each fire station, create MOBR with length of 1



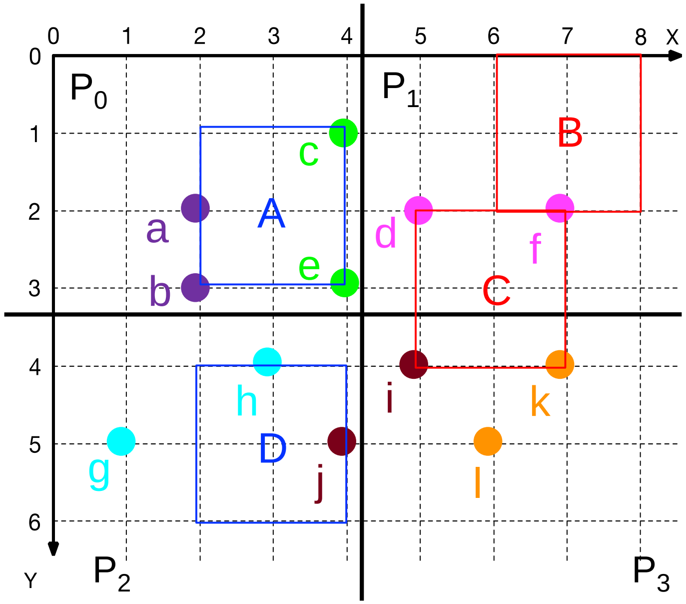
Q? Why C in two partitions?



P_0	A	a, b, c, e
P_1	B, C	d, f
P_2	D	g, h, j
P_3	C	i, k, l

Space Partition Join Algorithm

- Filter: For each partition Pi
  - Bring Partition in main memory
  - Test all pairs of MOBR Mfs of fire-station in Pi and all houses in Pi
- Refine: Test remaining pair with exact geometry, e.g., distance <= 1



Partitions

P_0	A	a, b, c, e
P_1	B, C	d, f
P_2	D	g, h, j
P_3	C	i, k, l

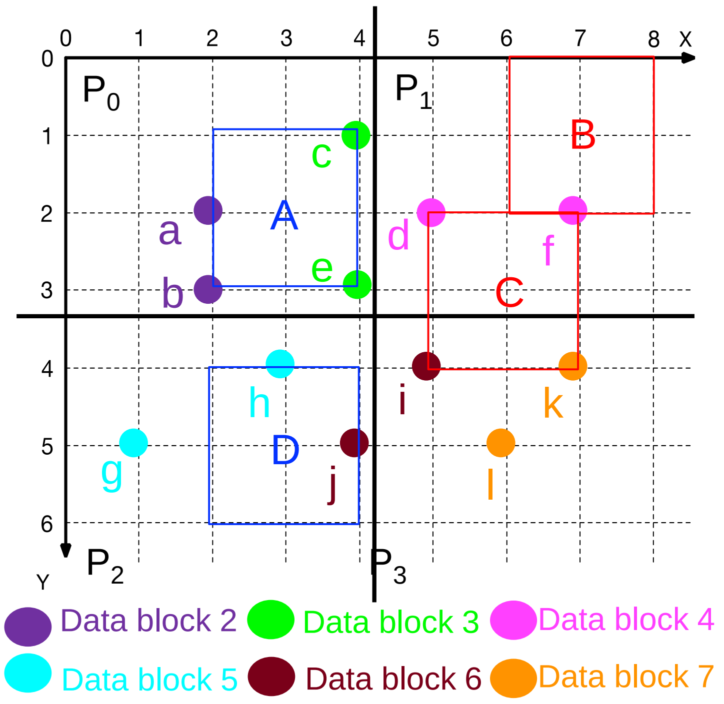
Result after Filter Phase

Result	MOBR	House
P_0	A	a, b, c, e
P_1	B	f
	C	d, f
P_2	D	h, j
P_3	C	i, k

Space Partition Join Algorithm

Total cost = 8+8+(3+2+3+3) = 27

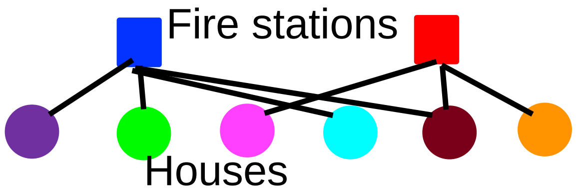
Read all data blocks	Write partitioning back	Compute for each partition	
8	8	P_0	3
		P_1	2
		P_2	3
		P_3	3



About 3 "scans" of each table
If replication of objects across partitions is rare.

Tree Matching

- Nested Loop with an Index
  - Inner loop range queries
  - Eliminated pairs of data-blocks if disjoint MOBRs

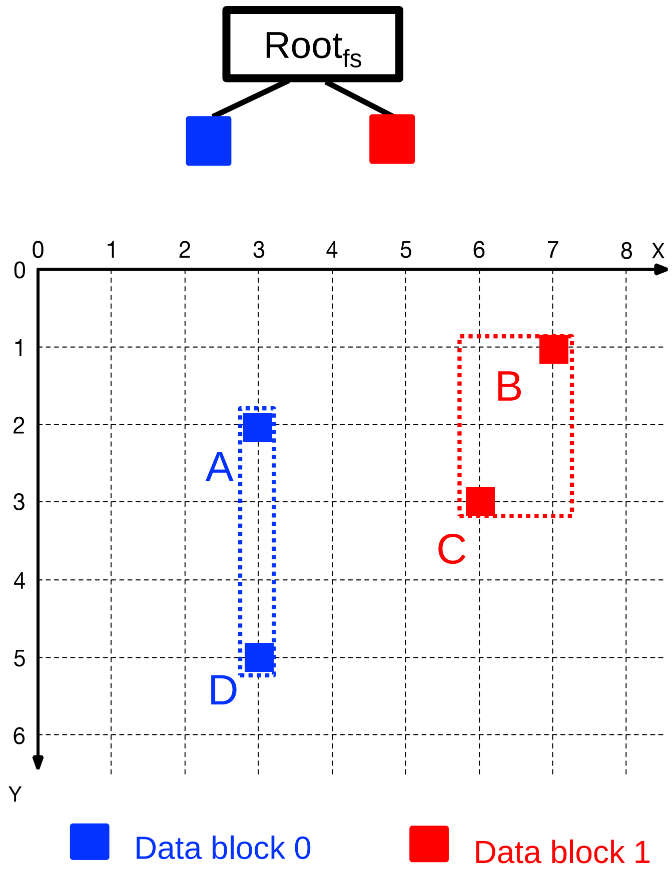


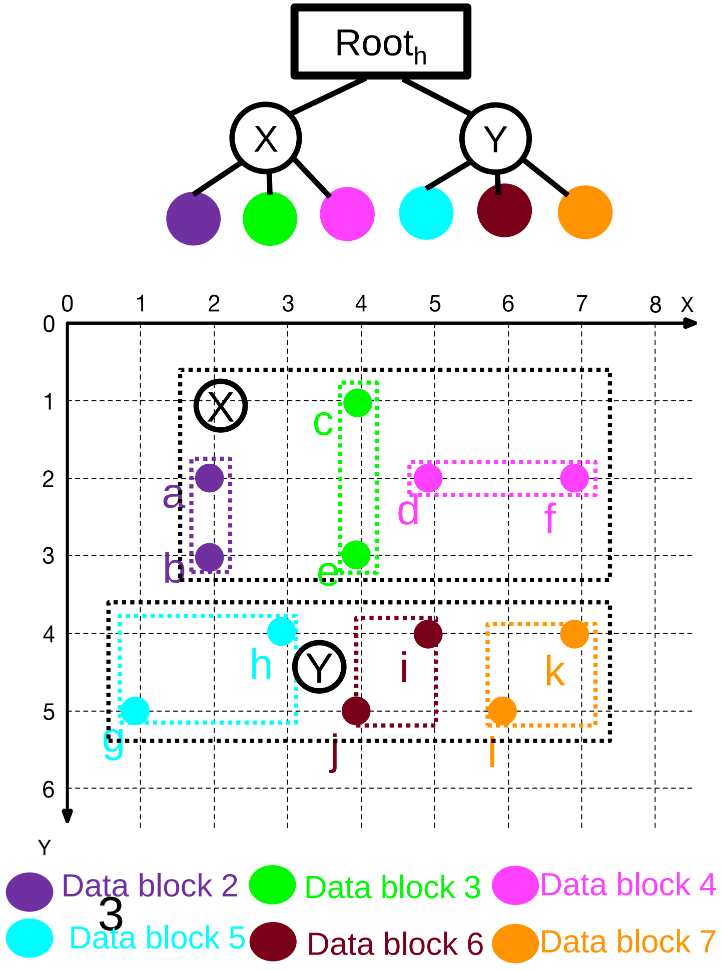
- Space-partitioning join
  - Eliminated partition-pairs ((P0, P1), …) since disjoint MOBRs

P_0	A	a, b, c, e
P_1	B, C	d, f
P_2	D	g, h, j
P_3	C	i, k, l

- Tree Matching, if both tables are indexed:
  - Eliminate pairs of index/data-blocks if disjoint MOBRs
  - Start at Root level – Eliminate child-pair if irrelevant
  - Recursion on remaining pairs

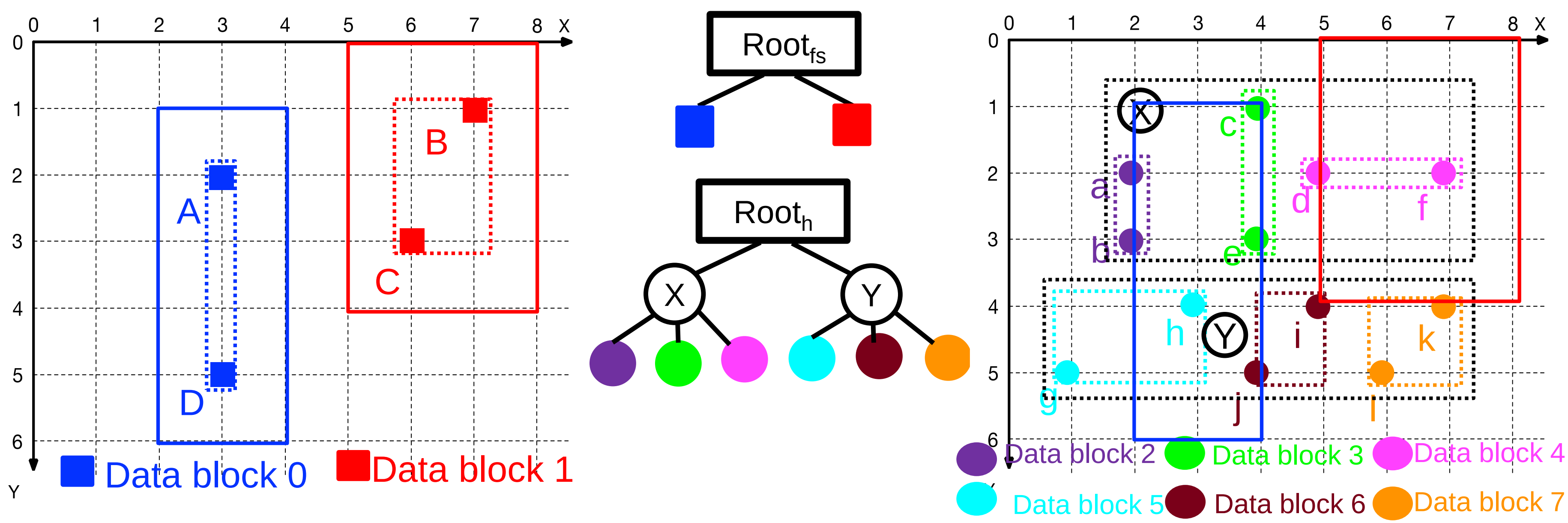
Tree Matching





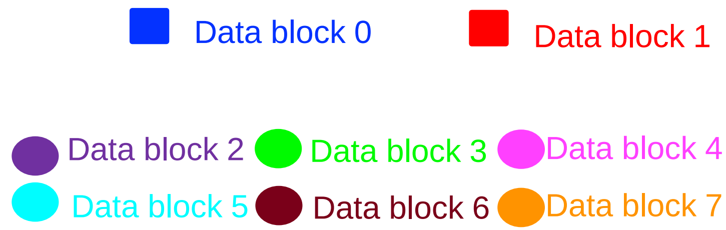
Tree Matching

- For each fire station, find houses within distance <= 1
- MOBR buffer of size 1 to mimic spatial join predicate, i.e. distance <= 1
- Root level – no child-pair is eliminated
- Recursion on remaining pairs, i.e., (X, 0), (Y, 0), (X, 1), (Y, 1)

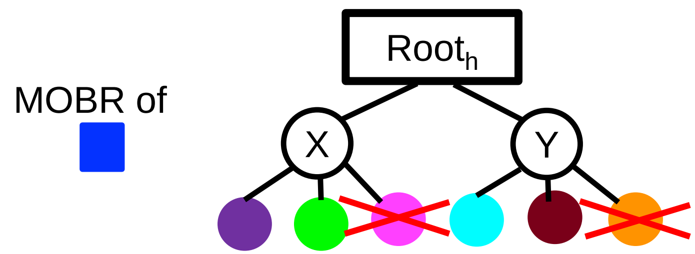


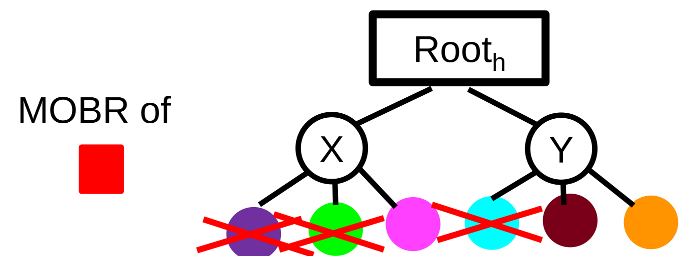
Tree Matching

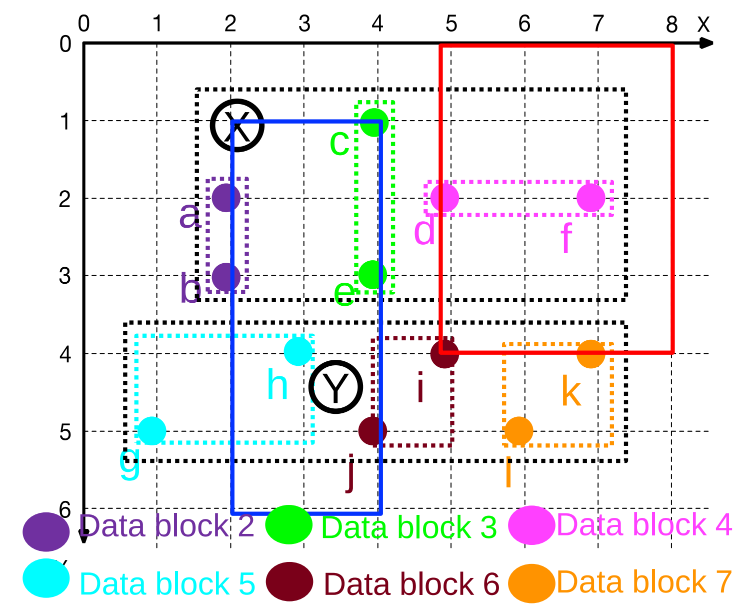
- Next Iteration



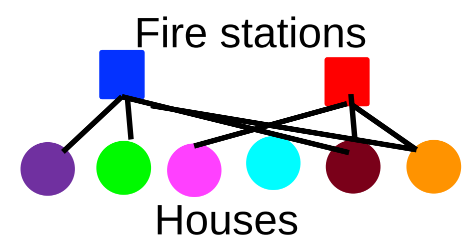
- Recursion on (X, 0) => remaining pairs: (2, 0), (3, 0),
- Recursion on (Y, 0) => remaining pairs: (5, 0), (6, 0)
- Recursion on (X, 1) => remaining pairs: (4, 1),
- Recursion on (Y, 1) => remaining pairs: (6, 1), (7, 1)







Index blocks	Data blocks	
2 + 2 = 4	FS	2
	House	4+3=7
	Total	9



Tree Matching

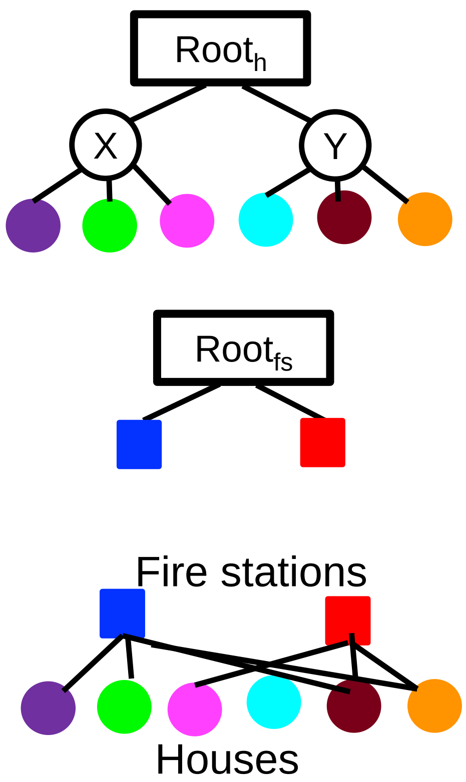
- Cost

Data block 0 Data block 1

Data block 2 Data block 3 Data block 4

Data block 5 Data block 6 Data block 7

- Pairs examined:
  - (X, 0), (Y, 0), (X, 1), (Y, 1)
  - (2, 0), (3, 0), (5, 0), (6, 0), (4, 1), (6, 1), (7, 1)
- Blocks accessed
  - Index blocks besides roots: X, Y
  - Data blocks: all with 6 accessed twice



Index blocks	Data blocks	
2 + 2 = 4	FS	2
	House	4+3=7
	Total	9

Comparing Algorithms for Spatial Join

- Default choice is Nested loop

- Neither table has spatial index
  - Space partitioning if spatial-join predicate is selective

- One table has a spatial index
  - Nested loop with index

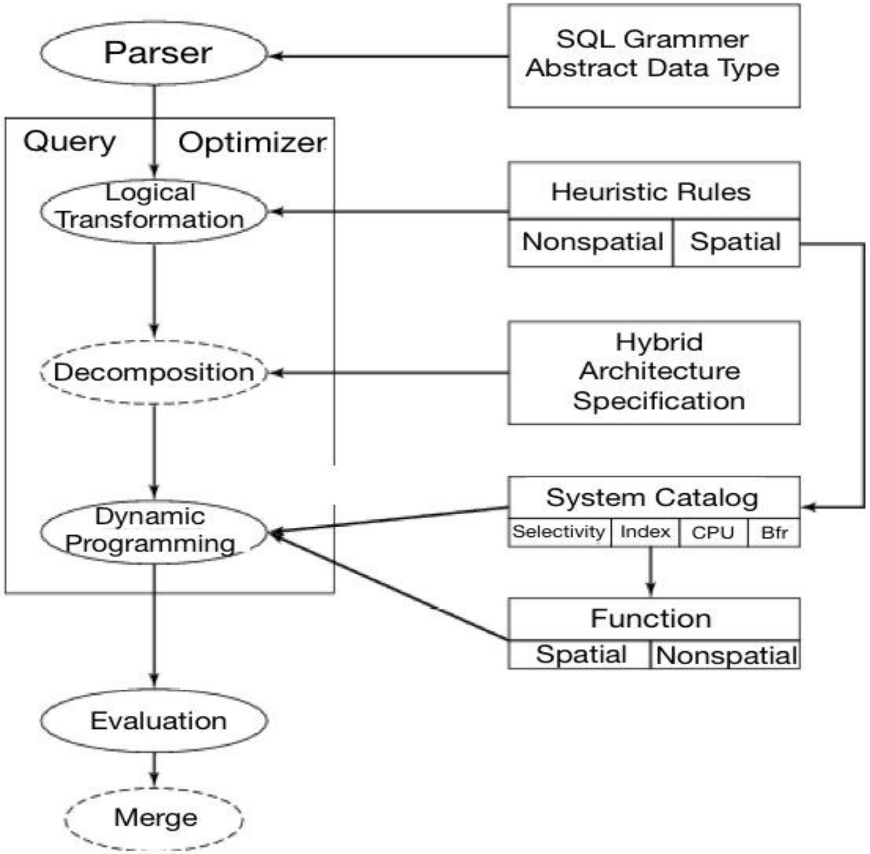
- Both table have spatial tree indexes & selective spatial join predicate
  - Tree matching

第六章 空间查询处理与优化

- 6.1 查询处理与优化
- 6.2 空间查询处理算法
- 6.3 查询优化
- 6.4 发展趋势 (自学)

Query Processing and Optimizer Process

- A site-seeing trip
  - Start : A SQL Query
  - End: An execution plan
  - Intermediate Stopovers
    - Query trees
    - Logical tree transforms
    - Strategy selection

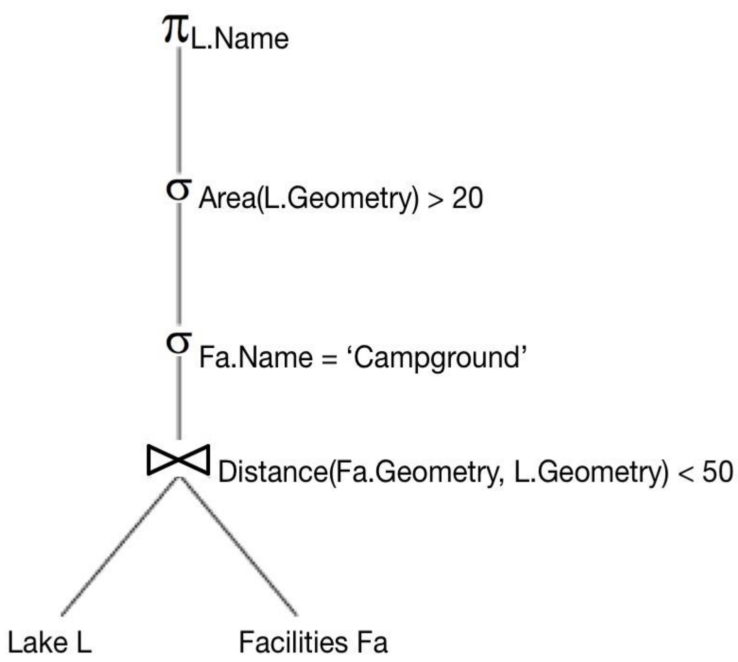


- What happens after the journey?
  - Execution plan is executed
  - Query answer returned

Query Trees

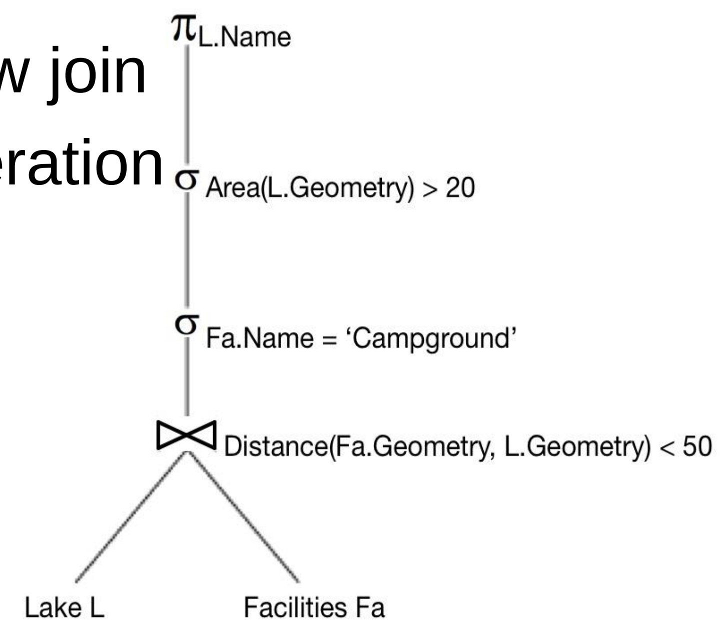
- Nodes = Building blocks of (spatial) queries
- Children = Inputs to a building block
- Leafs = Tables
- Example SQL query and its query tree follows:

```
Select L.Name
From Lake L, Facilities Fa
Where ST_Area(L.Geometry) > 20 and
         Fa.Name = 'compground' and
         ST_Distance(Fa.Geometry, L.Geometr
```



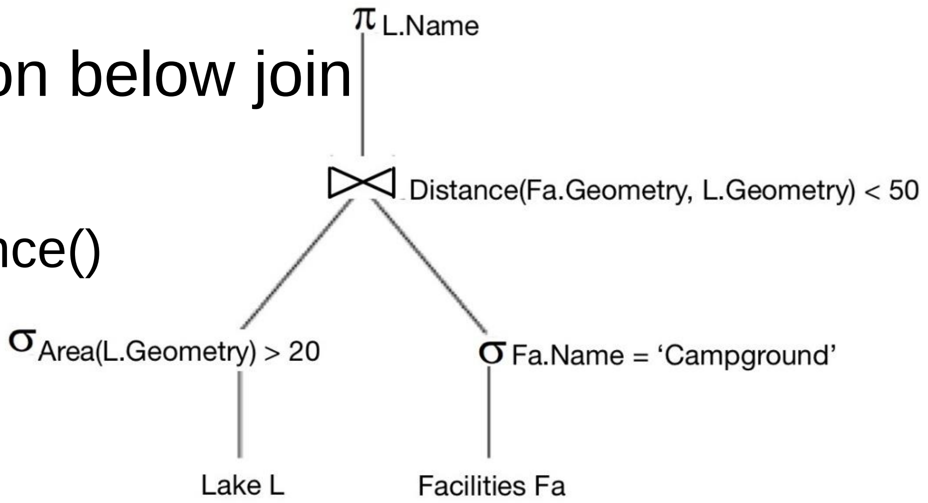
Logical Transformation of Query Trees

- Motivation
  - Transformation do not change the answer of the query
  - But can reduce computational cost by
    - Reducing data produced by sub-queries
    - Reducing computation needs of parent node
- Example Transformation
  - Push down select operation below join
  - Reduces size of table for join operation
- Other common transformations
  - Push project down
  - Reorder join operations
  - ...



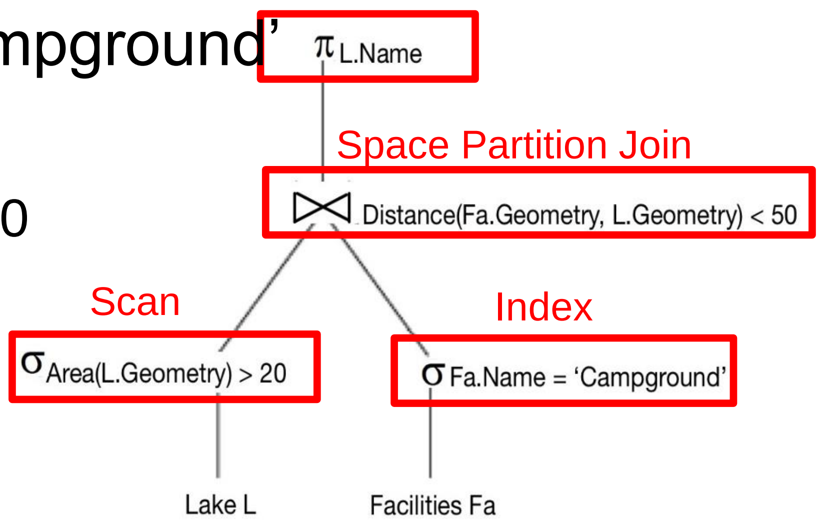
Logical Transformation and Spatial Queries

- Traditional logical transform rules
  - For relational queries with simple data types and operations
    - CPU costs are much smaller and I/O costs
  - Need to be reviewed for spatial queries
    - Complex data types, operations
    - CPU cost is hgher
- Example:
  - Push down spatial selection below join
  - May not decrease cost if
    - area() is costlier than distance()



Execution Plans

- An execution plan has 3 components
  - A query tree
  - A strategy selected for each non-leaf node
  - An ordering of evaluation of non-leaf nodes
- Example
  - Strategies for Query tree
  - Use scan for Area(L.geom) > 20
  - Use index for Fa.Name = 'Campground'
  - Use space-partitioning join for
    - Distance(Fa.geom, L.geom) < 50
  - Use on-the-fly for projection
  - Ordering
    - As listed above



Choosing Strategies for Building Blocks

- A priority scheme
  - Check applicability of each strategies given file-structures and indices
  - Choose highest priority strategy
  - This procedure is fast, Used for complex queries
- Rule based approach
  - System has a set of rules mapping situations to strategy choices
  - Example: Use scan for range query if result size > 10 % of data file
- Cost based approach

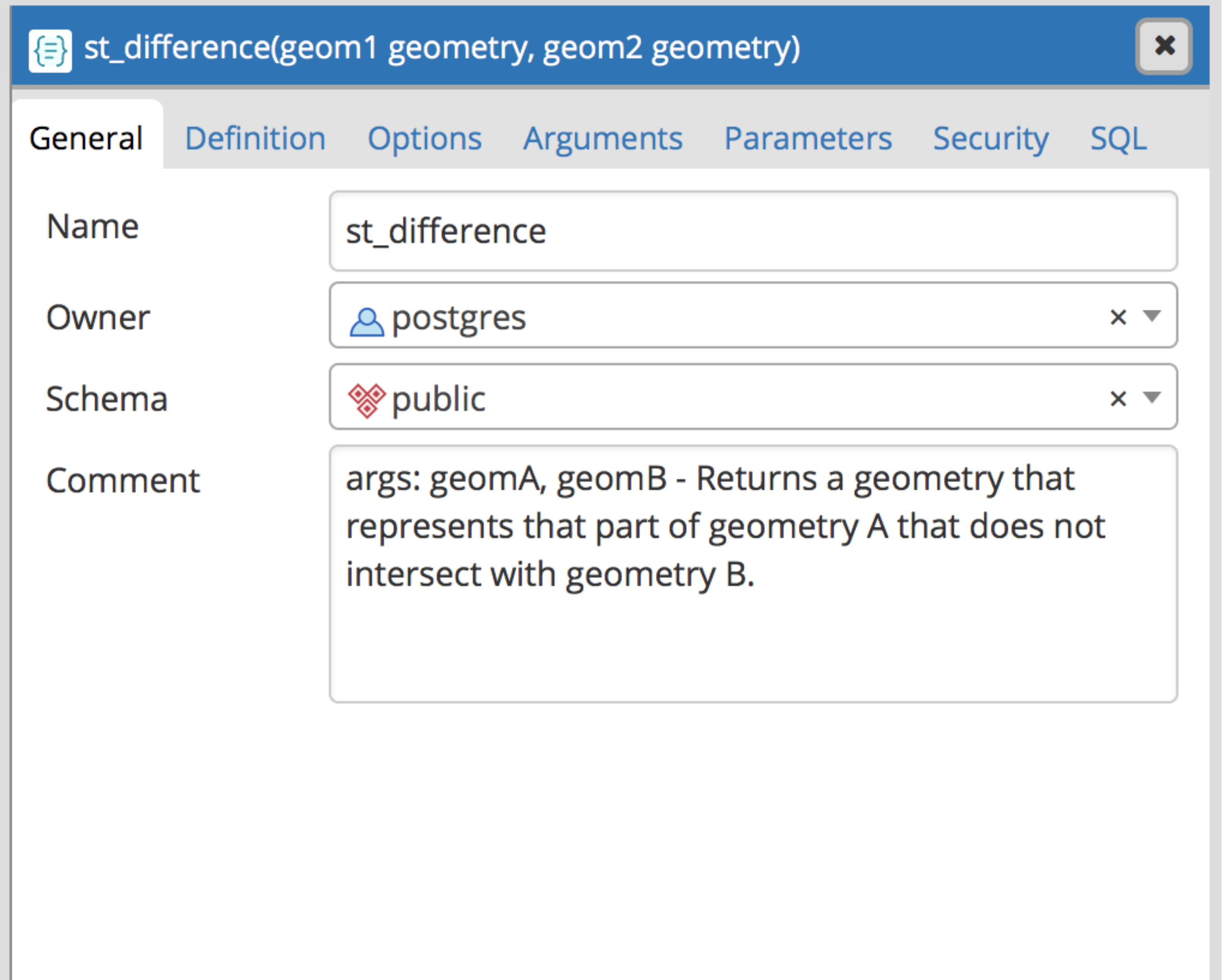
Choosing Strategies for Building Blocks

- Cost model based approach
  - Single building block
    - Use formulas to estimate cost of each strategy, given table size etc.
    - Choose the strategy with least cost
  - A query tree
    - Least cost combination of strategy choices for non-leaf nodes
    - Dynamic programming algorithm
- Commercial practice
  - RDBMS use cost based approach for relational building blocks
  - But cost models for spatial strategies are not mature
  - Rule based approach is often used for spatial strategies

思考：PostgreSQL+PostGIS使用哪种策略？

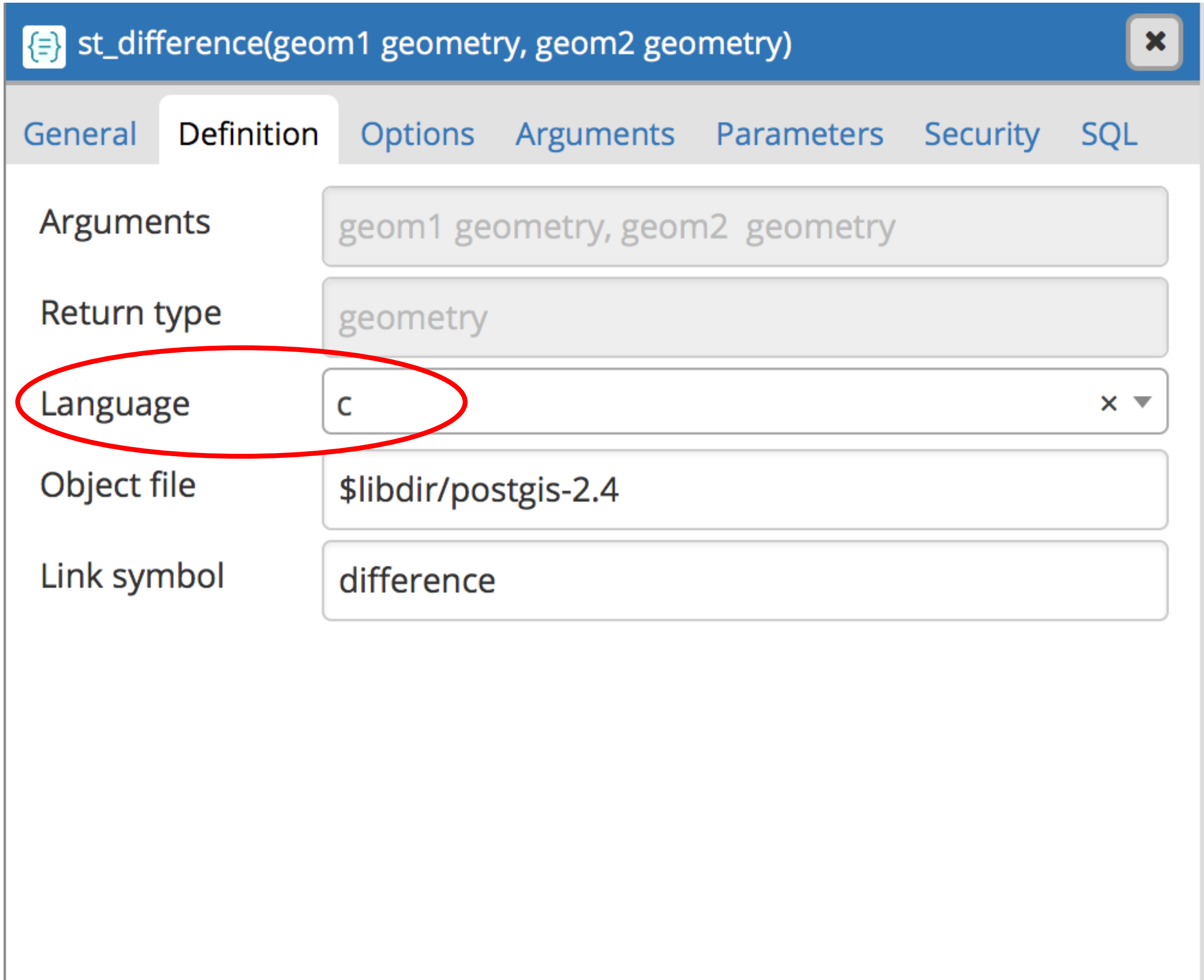
Cost Model Based Approach

- ST_Distance



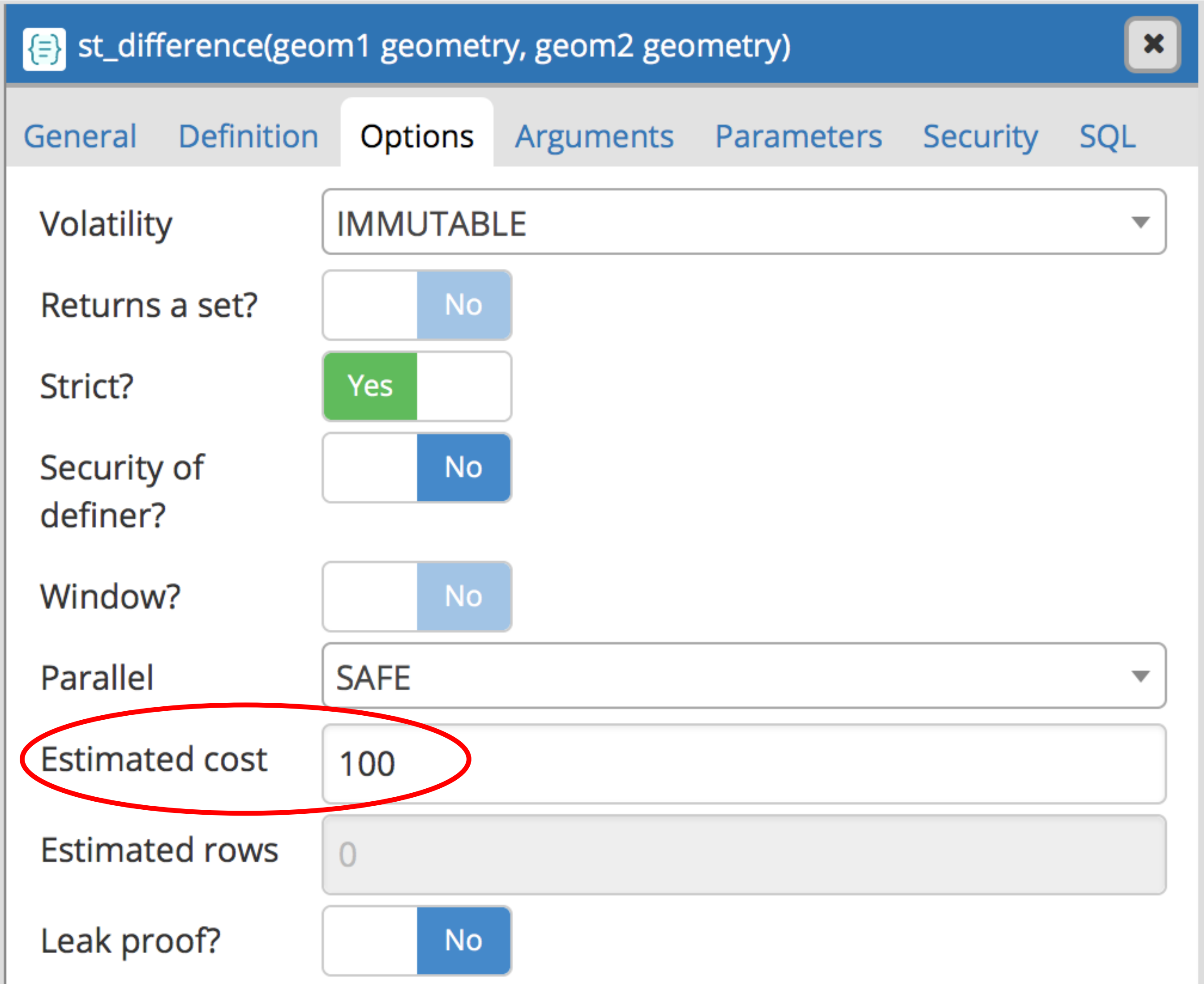
Cost Model Based Approach

- ST_Distance



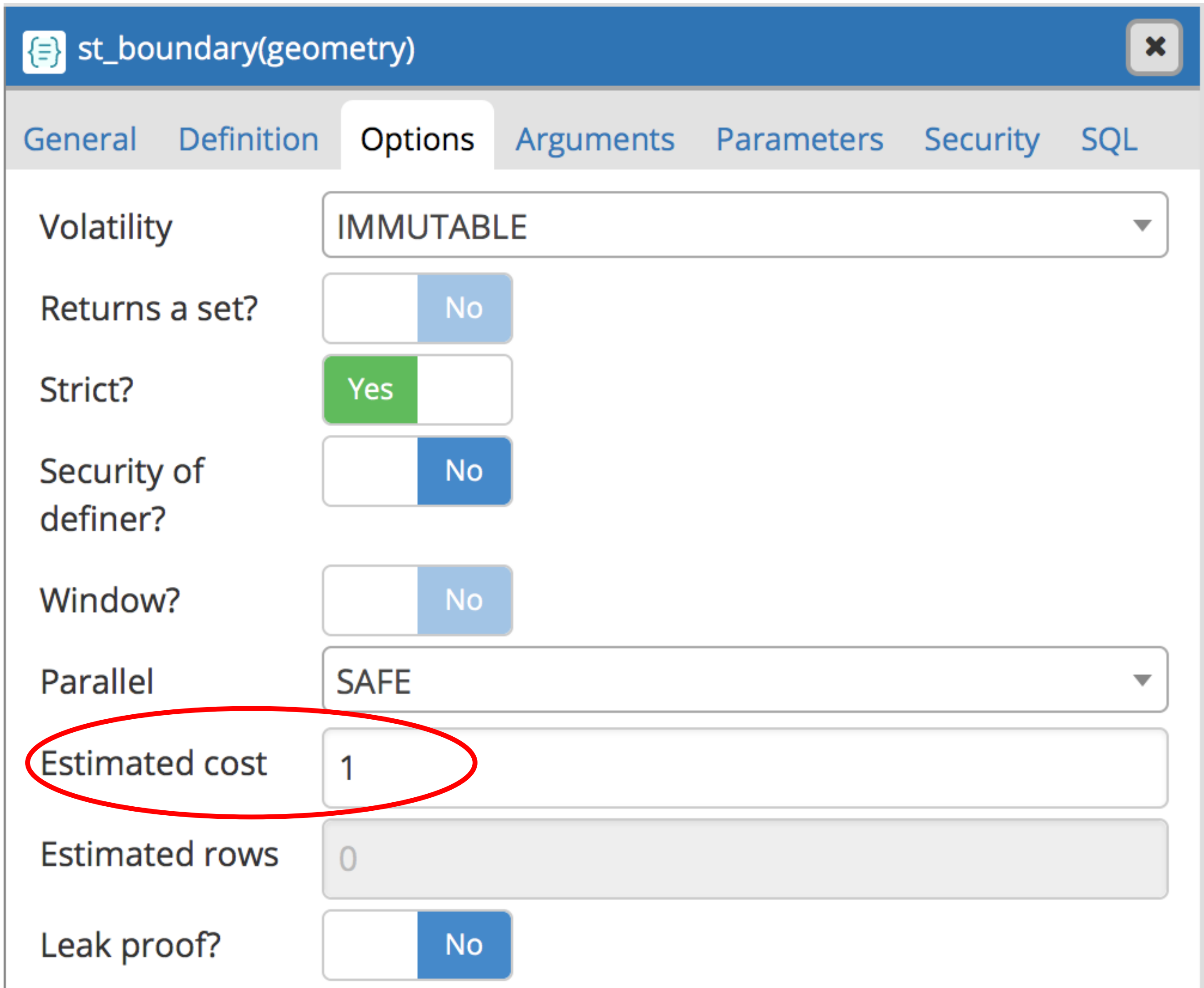
Cost Model Based Approach

- ST_Distance



Cost Model Based Approach

- ST_Boundary



Cost Model Based Approach

- explain select C.name, count(*)
- from ne_10m_admin_0_countries C, ne_10m_populated_places P
- where ST_Within(P.geom, C.geom) group by C.name order by C.name

输出窗口

数据输出 解释 消息 历史

	QUERY PLAN text
1	Sort (cost=497105.20..497105.84 rows=255 width=10)
2	Sort Key: c.name
3	-> HashAggregate (cost=497092.46..497095.01 rows=255 width=10)
4	Group Key: c.name
5	-> Nested Loop (cost=0.00..496886.56 rows=41179 width=10)
6	Join Filter: ((p.geom && c.geom) AND st contains(c.geom, p.geom))
7	-> Seq Scan on ne 10m admin 0 countries c (cost=0.00..72.55 rows=255 width=34734)
8	-> Materialize (cost=0.00..629.15 rows=7343 width=32)
9	-> Seq Scan on ne 10m populated places p (cost=0.00..592.43 rows=7343 width=32)

输出窗口

数据输出 解释 消息 历史

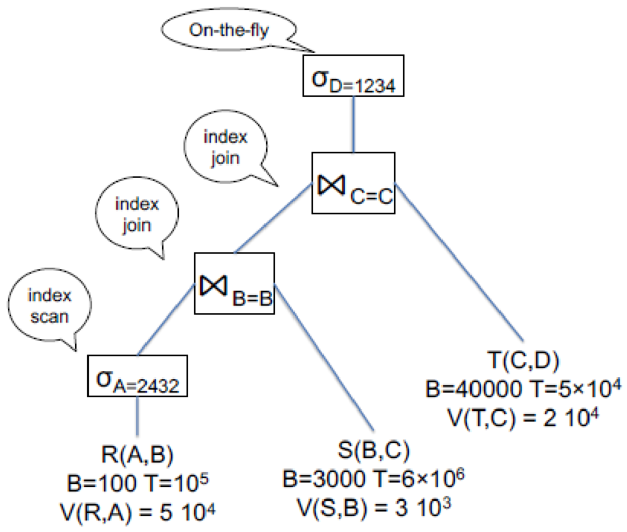
	QUERY PLAN text
1	Sort (cost=1411.56..1412.20 rows=255 width=10)
2	Sort Key: ne 10m admin 0 countries.name long
3	-> HashAggregate (cost=1398.82..1401.37 rows=255 width=10)
4	Group Key: ne 10m admin 0 countries.name long
5	-> Nested Loop (cost=0.15..1192.93 rows=41179 width=10)
6	-> Seq Scan on ne 10m admin 0 countries (cost=0.00..72.55 rows=255 width=34734)
7	-> Index Scan using icity on ne 10m populated places (cost=0.15..4.38 rows=1 width=32)
8	Index Cond: (geom && ne 10m admin 0 countries.geom)
9	Filter: st contains(ne 10m admin 0 countries.geom, geom)

Cost Model Based Approach

- BigData -‘基于代价优化’究竟是怎么一回事？http://mp.weixin.qq.com/s/qP86_R6IPlOBr_TKi_RkWw
- 两个问题，SQL执行引擎如何知晓参与Join的两波数据集大小？衡量两波数据集大小的是物理大小还是纪录多少抑或两者都有？这关系到SQL解析器如何正确选择Join算法的问题。好了，这些就是这篇文章要为大家带来的议题－基于代价优化（Cost-Based Optimization，简称CBO）

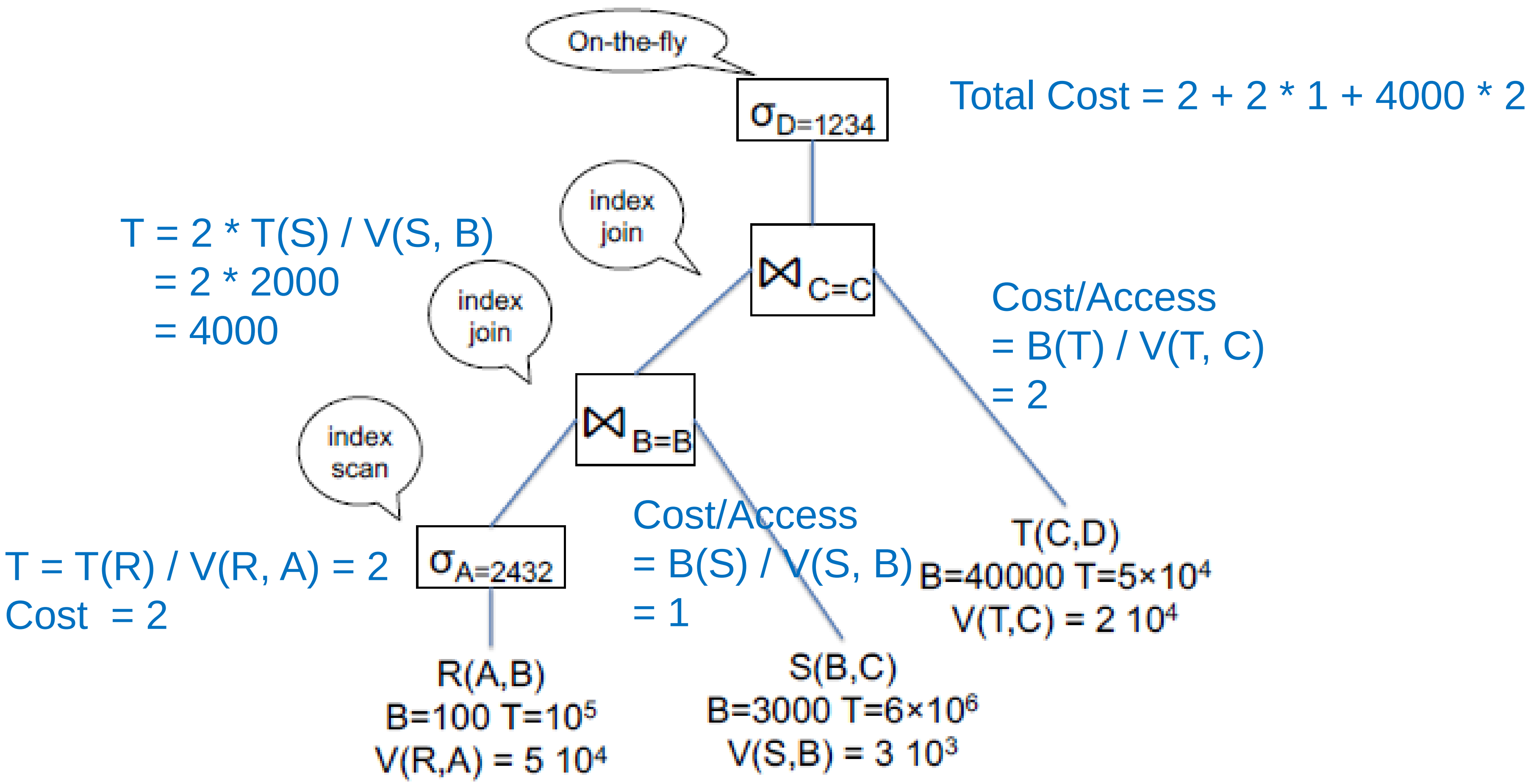
Cost Model Based Approach Example

- 关系R(A, B), S(B, C), T(C, D)统计信息如下
  - 数据量$T(R) = 10^5$, $T(S) = 6 * 10^6$, $T(T) = 5 * 10^4$
  - 数据存储的Block数目$B(R) = 100$, $B(S) = 3000$, $B(T) = 40000$
  - 每个数值的取值数目$V(R, A) = 5 * 10^4$, $V(R,B) = V(S, B) = 3 * 10^3$, $V(S, C) = V(T, C) = 2 * 10^4$, $V(T, D) = 10^4$
  - 非聚集索引R.A, R.B, S.C, T.D
  - 聚集索引S.B, T.C



Cost Model Based Approach Example

- 关系R(A, B), S(B, C), T(C, D)统计信息如下
  - 非聚集索引R.A, R.B, S.C, T.D, 聚集索引S.B, T.C

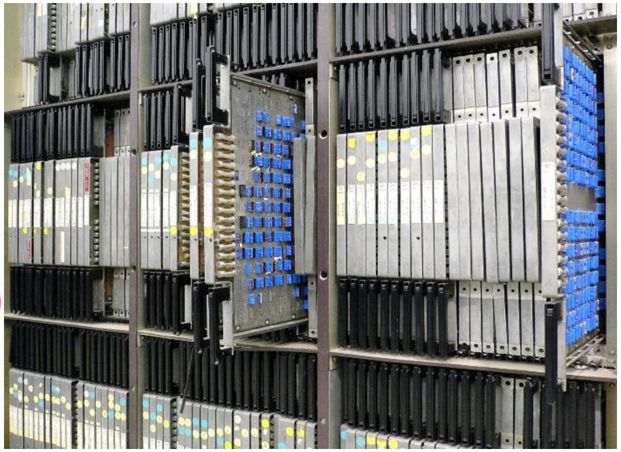


第六章 空间查询处理与优化

- 6.1 查询处理与优化
- 6.2 空间查询处理算法
- 6.3 查询优化
- 6.4 发展趋势 (自学)

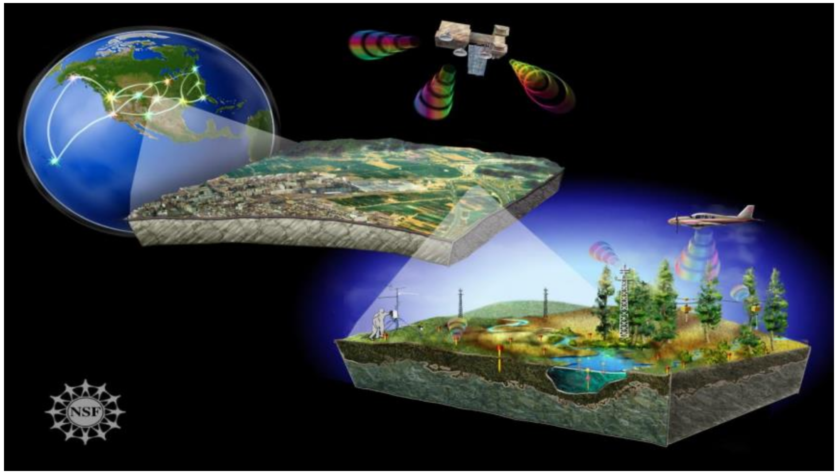
Trends in Query Processing and Optimization

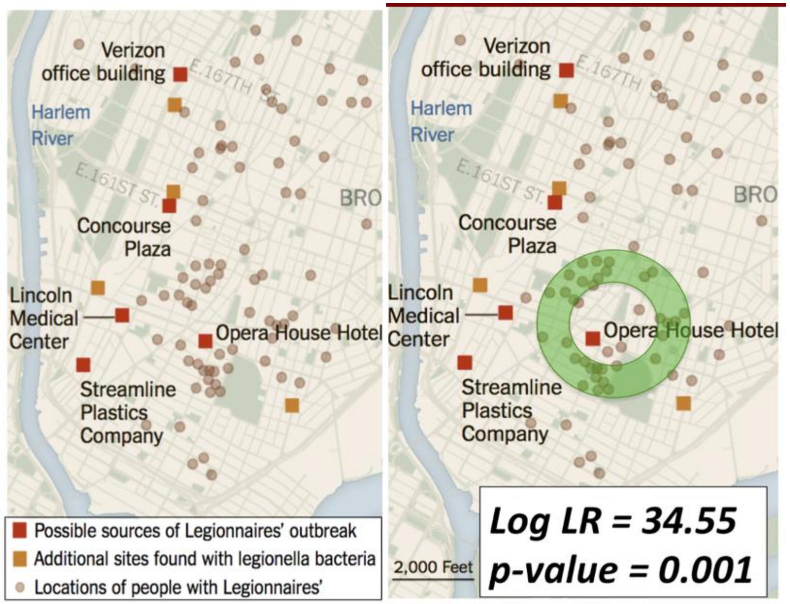
- Motivation
  - SDBMS and GIS are invaluable to many organizations
  - Price of success is to get new requests from customers
    - Support new computing hardware and environment
    - Support new applications
- New computing environments
  - Distributed computing
  - Internet and web
  - Parallel computers



Trends in Query Processing and Optimization

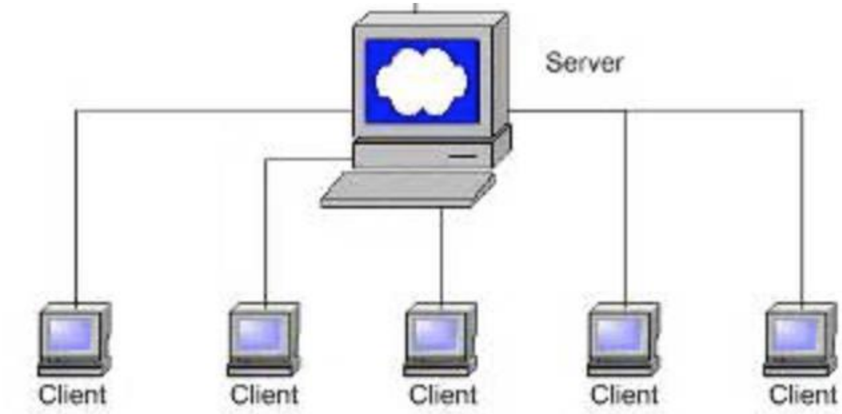
- New applications
  - Location based services, transportation
  - Data Mining
  - Raster data





Distributed Spatial Databases

- Distributed Environments
  - Collection of autonomous heterogeneous computers
  - Connected by networks
  - Client-server architectures
    - Server computer provides well-defined services
    - Client computers use the services

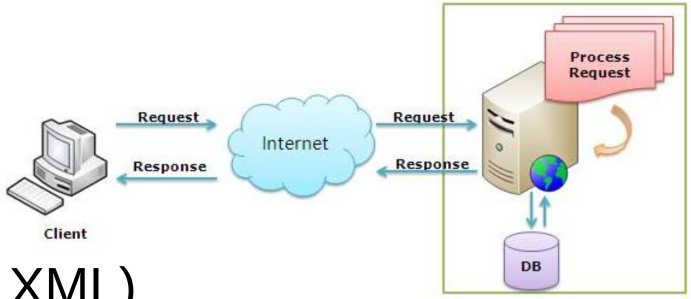


Distributed Spatial Databases

- New issues for SDBMS
  - Conceptual data model
    - Translation between heterogeneous schemas
  - Logical data model
    - Naming and querying tables in other SDBMSs
    - Keeping copies of tables (in other SDBMs) consistent with original table
  - Query Processing and Optimization
    - Cost of data transfer over network may dominate CPU and I/O costs
    - New strategies to control data transfer costs

Internet and (World-wide-)web

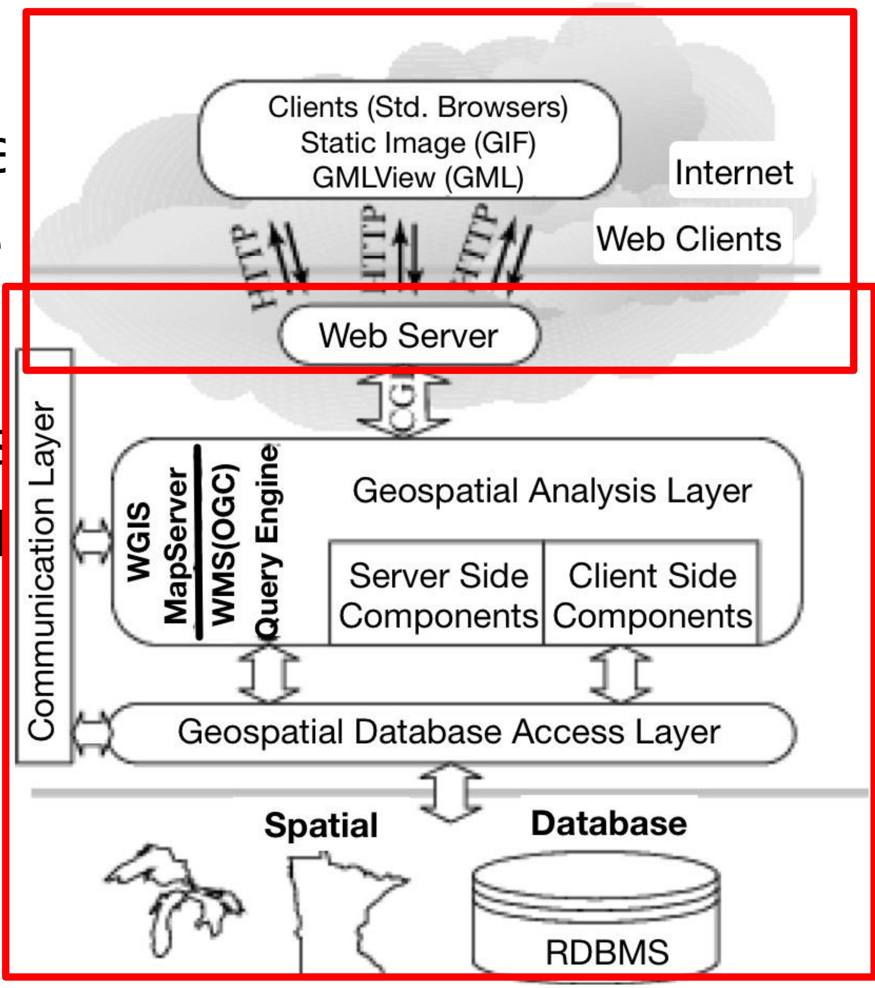
- Internet and Web Environments
  - Very popular medium of information access in last few years
  - A distributed environment
  - Web servers, web clients
    - Common data formats (e.g. HTML, XML)
    - Common communication protocols (e.g. http)
    - Naming - uniform resource locator (url), e.g. www.cs.umn.edu



- New issues for SDBMS
  - Offer SDBMS service on web
  - Use Web data formats, communication protocols etc.
  - Evaluate and improve web for SDBMS clients and servers

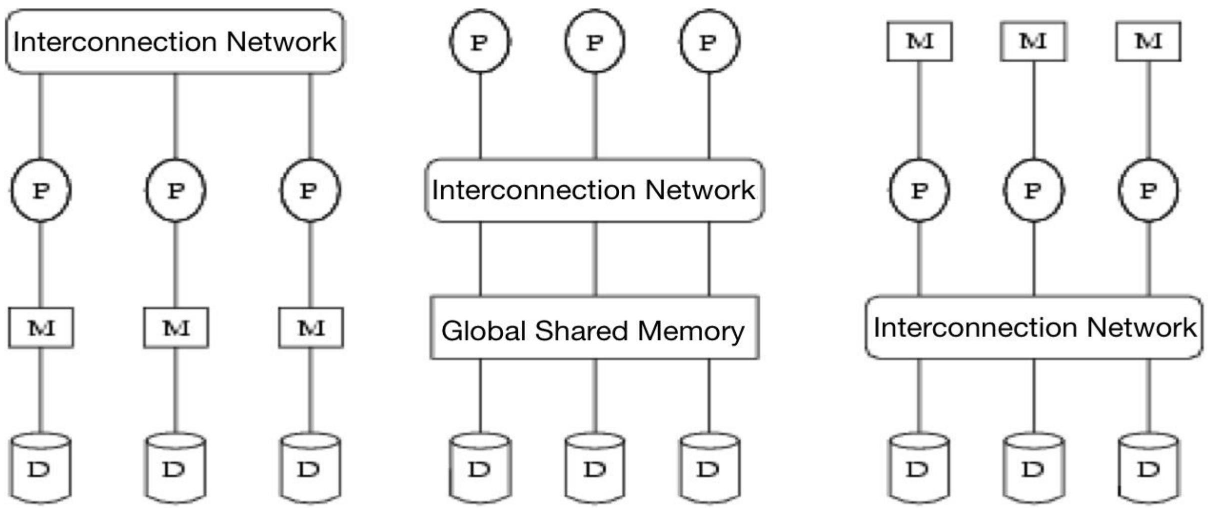
Web-based Spatial Database Systems

- SDBMS on web
  - MapServer case study
  - SDBMS talks to a web serve
  - Web server talks to web clie
- Commercial practice
  - Several web based products
  - Web data formats for spatial
    - GML
    - WMS



Parallel Spatial Databases

- Parallel Environments
  - Computer with multiple CPUs, Disk drives
  - All CPUs and disk available to a SDBMS
  - Can speed-up processing of spatial queries



Shared-Nothing (a) | Shared-Memory (b) | Shared-Disk (c)

Parallel Spatial Databases

- New issues for DBMS
  - Physical Data Model
    - Declustering: How to partition tables, indices across disk drives?
  - Query Processing and Optimization
    - Query partitioning: How to divide queries among CPUs?
    - Cost model of strategies on parallel computers
- Exmaple: Techniques for declustering
  - Simple technique: round robin based on an order (space filling curve)
  - Disk

Declustering for Data Partitioning

- Exmaple
  - A Simple Techniques for declustering
    - 1. Order the spatial objects using a space filling curve
    - 2. Allocate to disk drives in a round robin manner
  - Effective for point objects, e.g. pixels in an image
  - Many queries, e.g. large MBRs are parallelized well
    - Exercise: Consider a query to retrieve dat in bottom-left quarter of the space
    - Two data points retrieved fromeach disk drive for Z-curve

3	4	5	6	7	0	1	2
6	7	0	1	2	3	4	5
1	2	3	4	5	6	7	0
4	5	6	7	0	1	2	3
7	0	1	2	3	4	5	6
2	3	4	5	6	7	0	1
5	6	7	0	1	2	3	4
0	1	2	3	4	5	6	7

Linear Method
disk-id =
(x+5y) mod 8

7	0	1	2	3	4	5	6
6	7	0	1	2	3	4	5
5	6	7	0	1	2	3	4
4	5	6	7	0	1	2	3
3	4	5	6	7	0	1	2
2	3	4	5	6	7	0	1
1	2	3	4	5	6	7	0
0	1	2	3	4	5	6	7

CMD Method
disk-id =
(x+y) mod 8

42	43	46	47	58	59	62	63
40	41	44	45	56	57	60	61
34	35	38	39	50	51	54	55
32	33	36	37	48	49	52	53
10	11	14	15	26	27	30	31
8	9	12	13	24	25	28	29
2	3	6	7	18	19	22	23
0	1	4	5	16	17	21	21

2	3	6	7	2	3	6	7
0	1	4	5	0	1	4	5
2	3	6	7	2	3	6	7
0	1	4	5	0	1	4	5
2	3	6	7	2	3	6	7
0	1	4	5	0	1	4	5
2	3	6	7	2	3	6	7
0	1	4	5	0	1	4	5

Z-Curve Method
disk-id = Z(x,y) mod 8

63	62	49	48	47	44	43	42
60	61	50	51	46	45	40	41
59	56	55	52	33	34	39	38
58	57	54	53	32	35	36	37
5	6	9	10	31	28	27	26
4	7	8	11	30	29	24	25
3	2	13	12	17	18	23	22
0	1	14	15	16	19	20	21

7	6	1	0	7	4	3	2
4	5	2	3	6	5	0	1
3	0	7	4	1	2	6	5
2	1	6	5	0	3	1	2
5	6	1	2	7	4	3	2
4	7	0	3	6	5	0	1
3	2	5	4	1	2	7	6
0	1	6	7	0	3	4	5

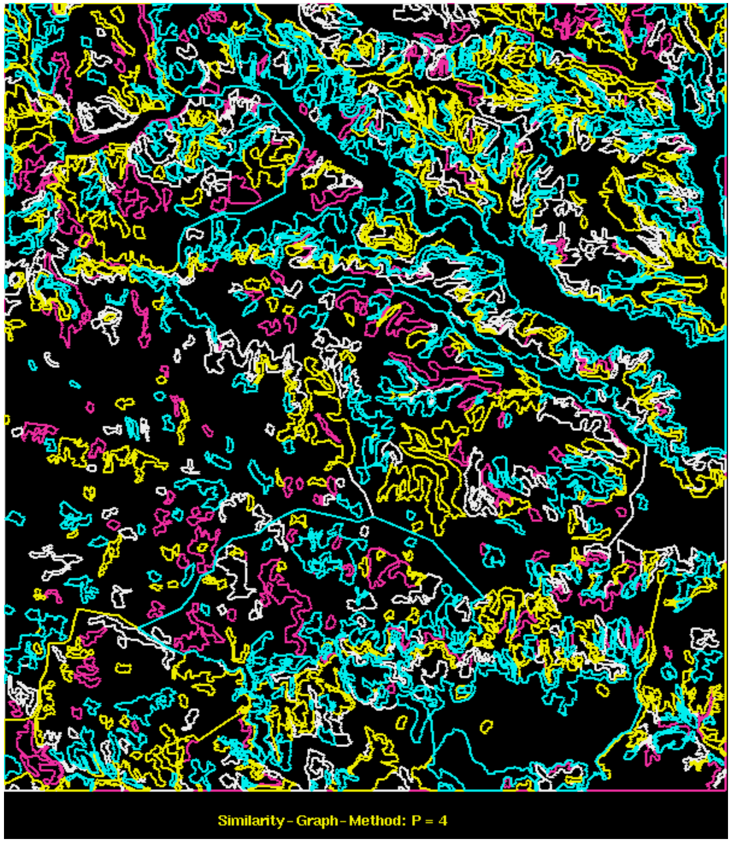
Hilbert Method
disk-id = H(x,y) mod 8

A Case Study: High Performance GIS

- Goal: Meet the response time constraint for real time battlefield terrain visualization in flight simulator
- Methodology:
  - Data-partitioning approach
  - Evaluation on parallel computers
    - e.g. Cray T3D, SGI Challenge
- Significance:
  - A major improvement in capability of geographic information systems for determining the subset of terrain polygons within the view point (Range Query) of a soldier in a flight simulator using real geographic terrain data set

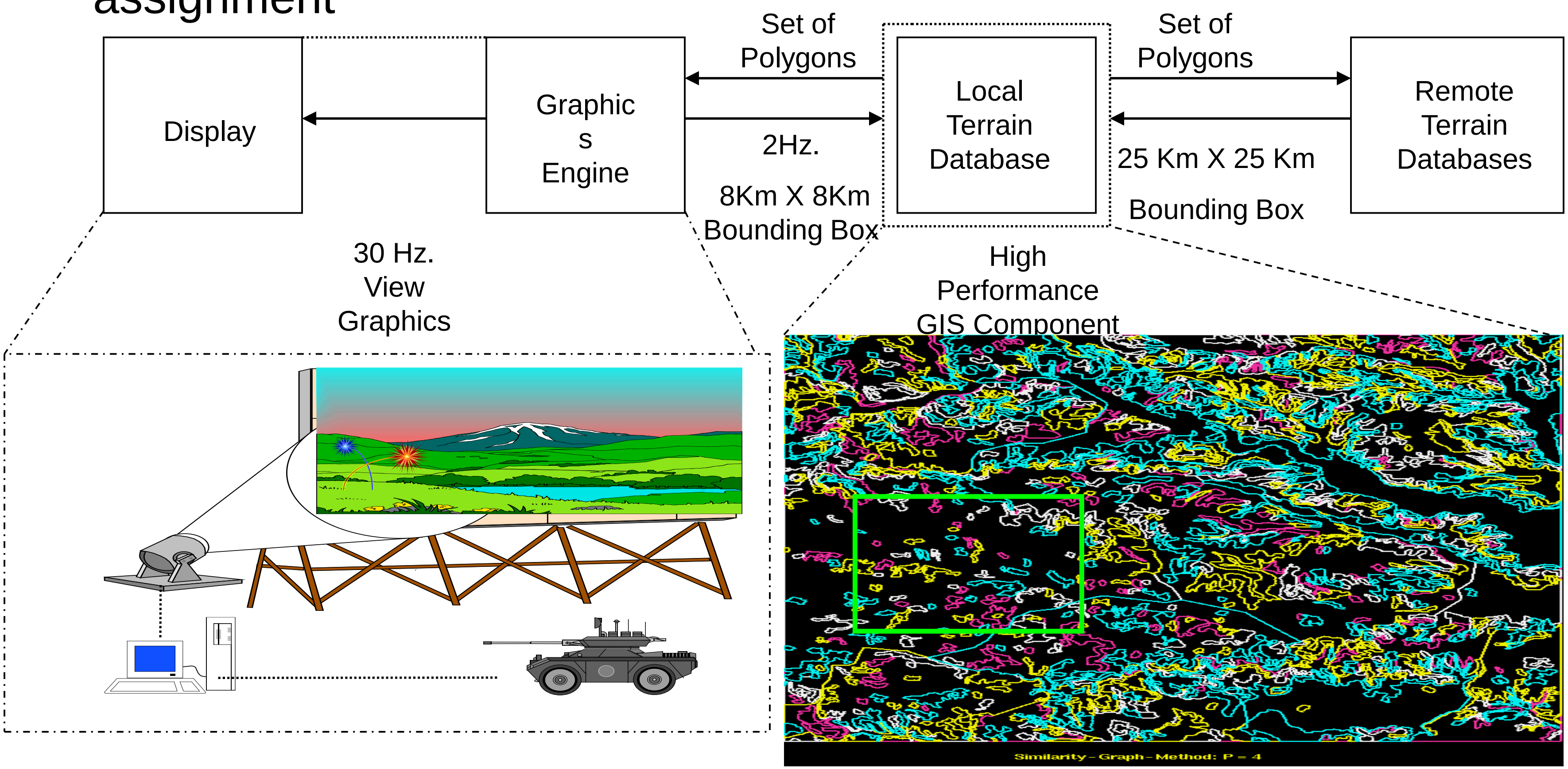
A Case Study: High Performance GIS

- Dividing a Map among 4 processors. Polygons within a processor have common color



A Case Study: High Performance GIS

- (1/30) second Response time constraint on Range Query
- Parallel processing necessary since best sequential computer cannot meet requirement
- Green rectangle = a range query, Polygon colors shows processor assignment



空间查询处理与优化总结

- Query processing and optimization (QPO)
  - Translates SQL Queries to execution plan
- QPO process steps include
  - Creation of a query tree for the SQL query
  - Choice of strategies to process each node in query tree
  - Ordering the nodes for execution
- Key ideas for SDBMS include
  - Filter-Refine paradigm to reduce complexity
  - New building blocks and strategies for spatial queries
  - CPU cost is higher
    - Push down spatial selection below join?
- 4类典型空间查询，3种执行规划选择策略