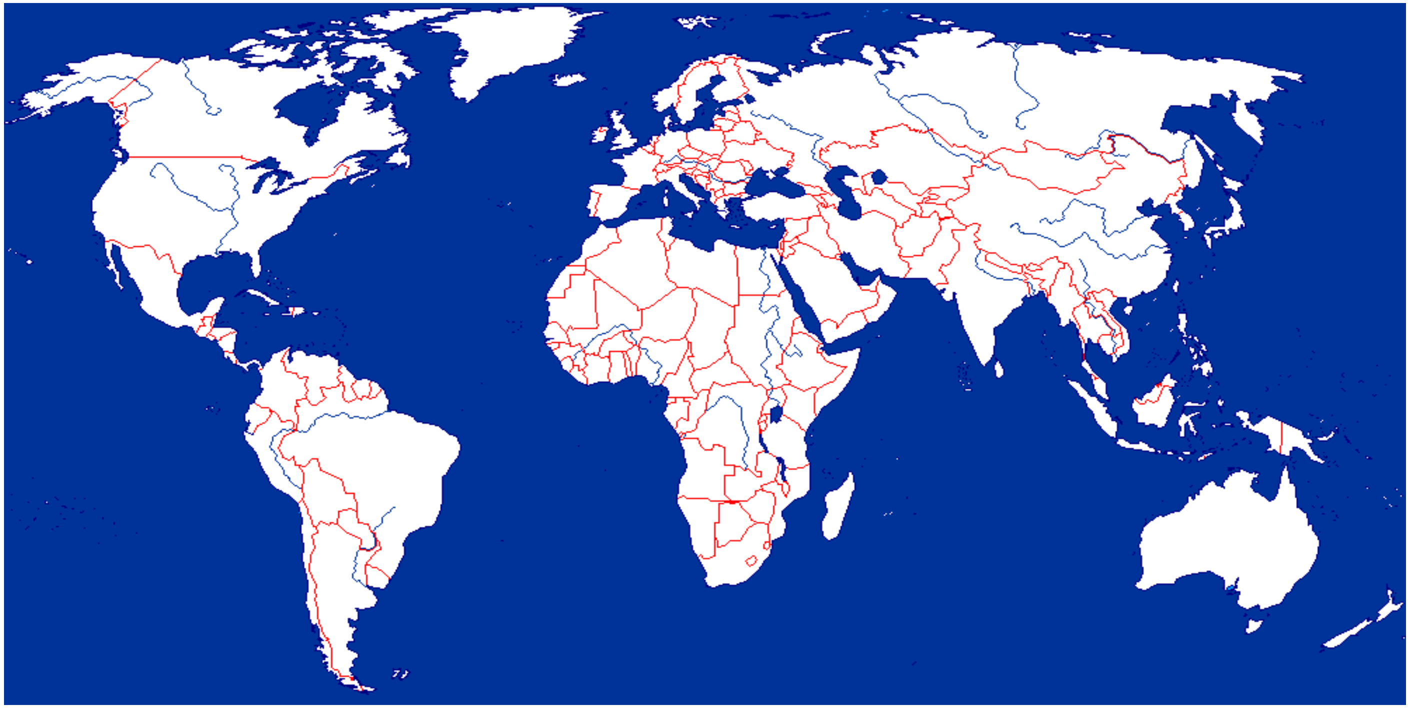
Strategies for Spatial Joins

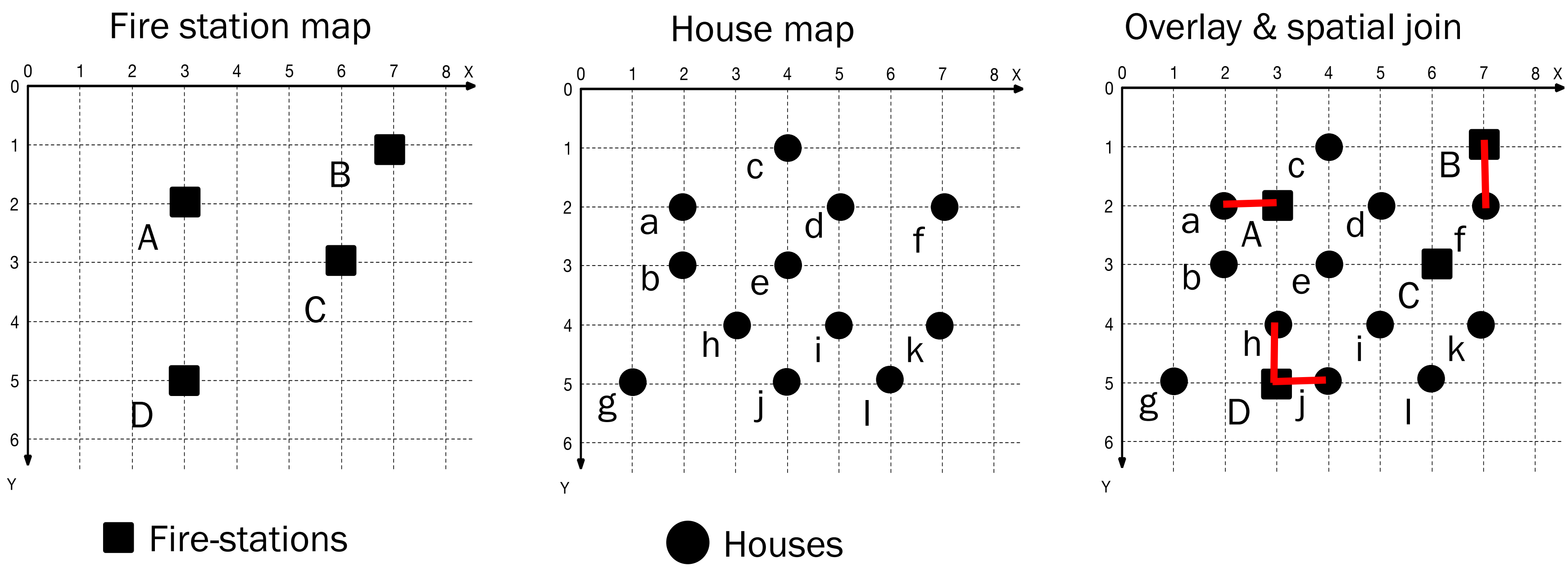
- Recall Spatial Join Example:
  - List all pairs of overlapping rivers and countries.
  - Return pairs from “rivers” table and “countries” table satisfying the “overlap” predicate.



Strategies for Spatial Joins - Continued

- **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, …*

Spatial Join - Running Examples



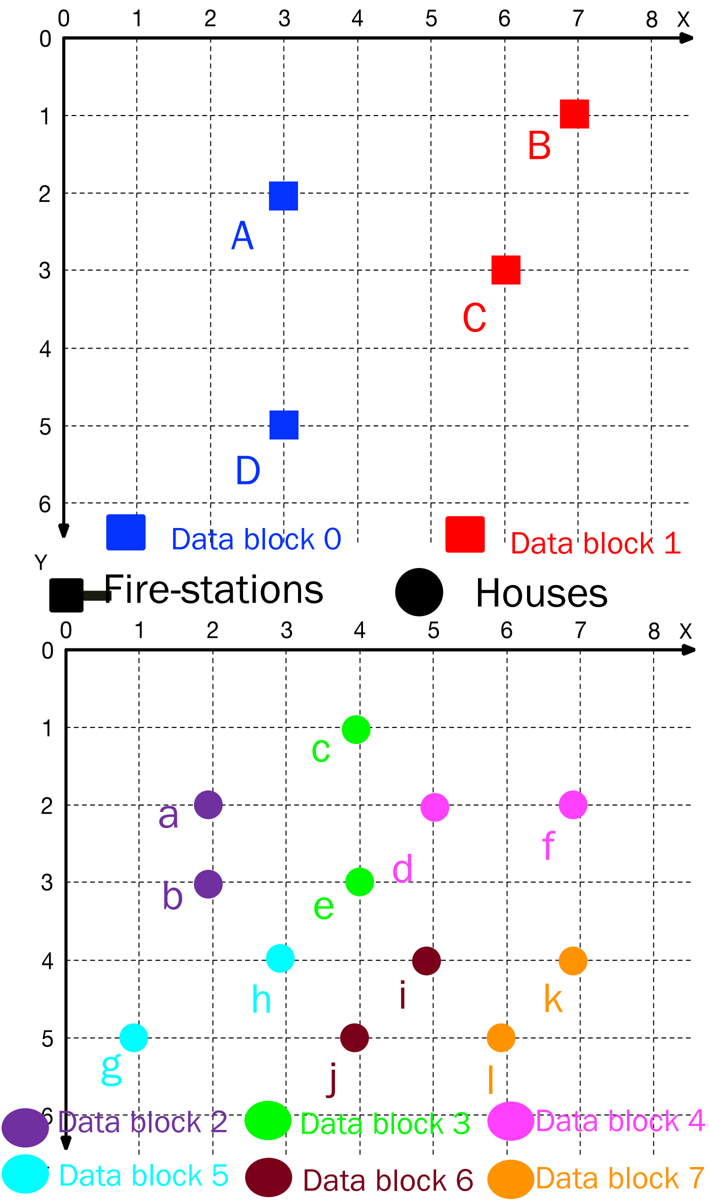
Query:

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

Results:

Fire-stations	Houses
A	a
B	f
D	h
D	j

Nested loop



Query:

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

Suppose: 1) each data block has 2 points
2) the size of memory buffer is 3 blocks
(i.e., 1 for fire-stations, 1 for houses, 1 for results)

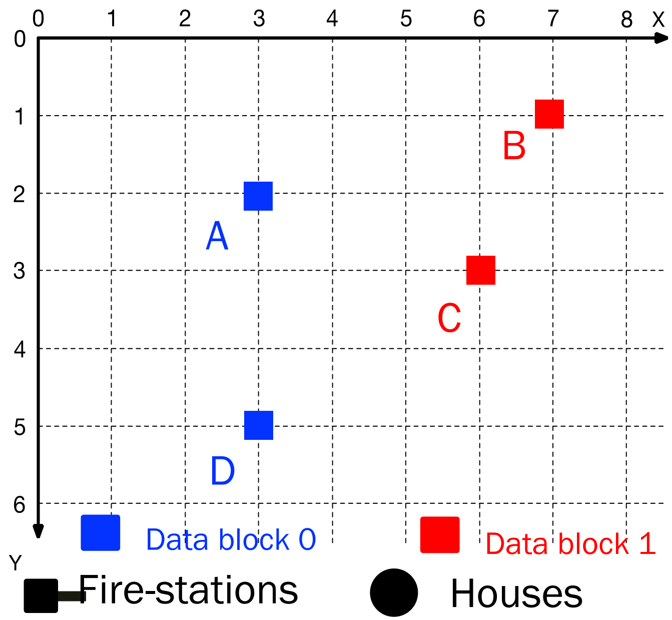
Algorithm:

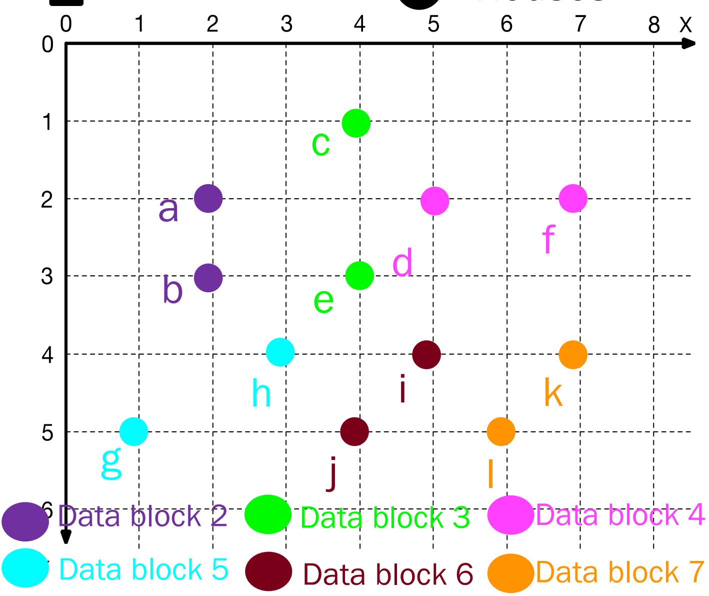
For each block B_{fs} of fire stations

For each block B_h of houses

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

Nested loop





Query:

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

Suppose: 1) each data block has 2 points

2) the size of memory buffer is 3 blocks

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

Algorithm:

For each block B_{fs} of fire stations

For each block B_h of houses

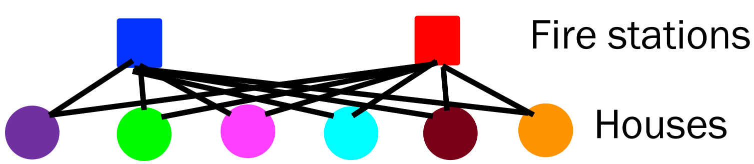
Scan all pairs of fire stats in B_{fs} and houses in B_h

For Block 0, traverse through Blocks 2-7

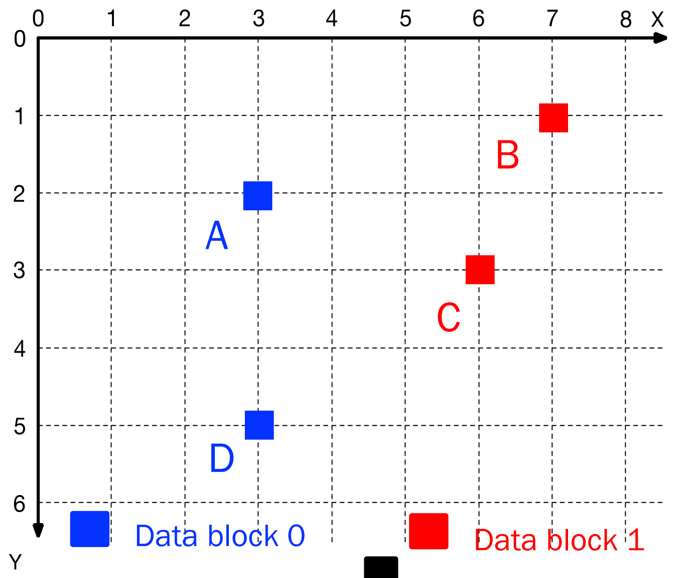
For Block 1, traverse through Blocks 2-7

Cost:

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



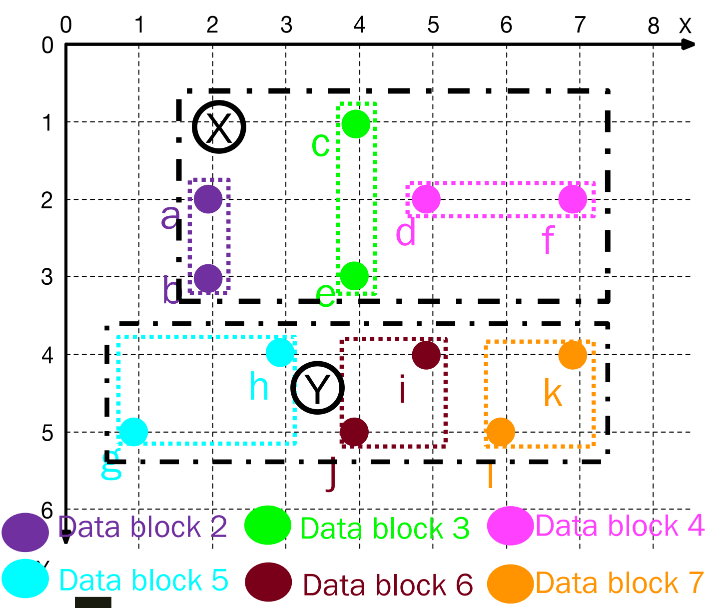
Nested loop with Index for Inner Loop

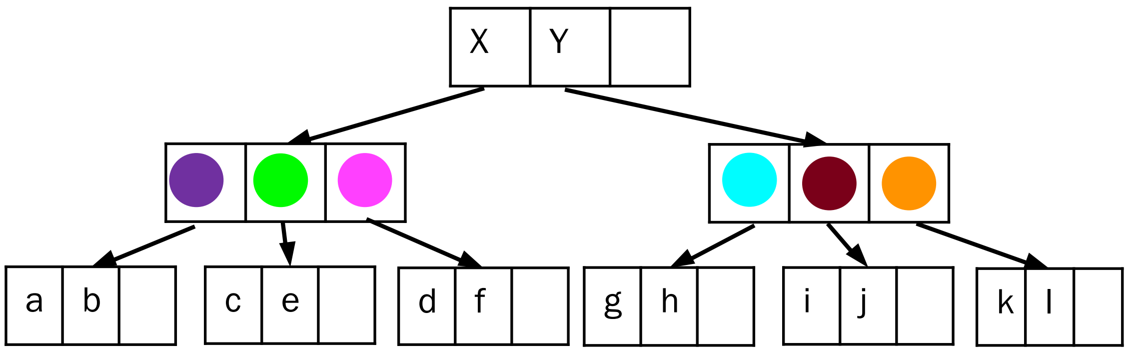


Query:

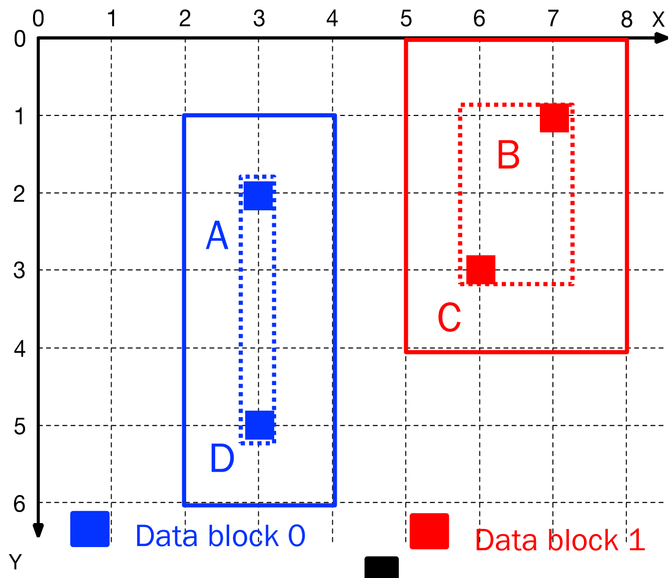
For each fire station, find all the houses within a distance <= 1

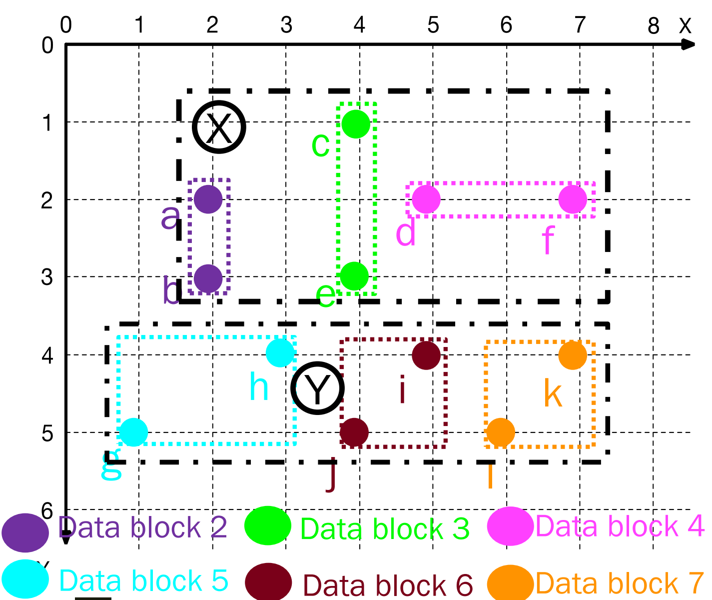
Suppose an R-tree (primary index) is available for the houses.





Nested loop with Index for Inner Loop



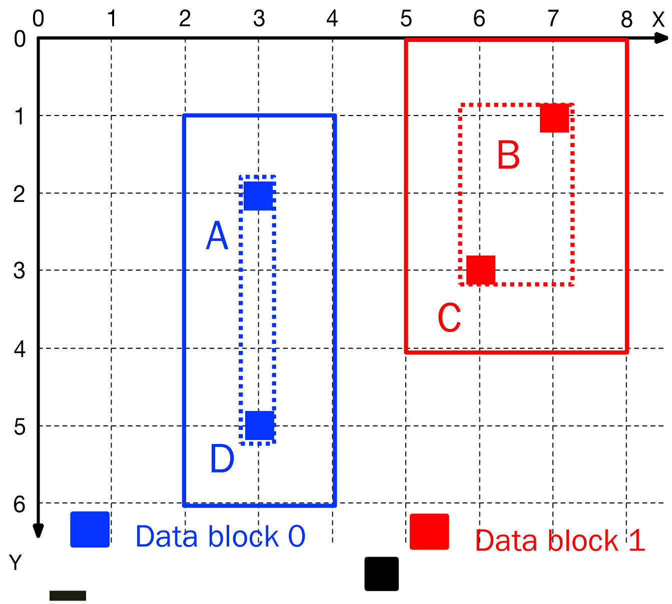


Query:

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

For each block of fire stations, create MOBR with length of 1.

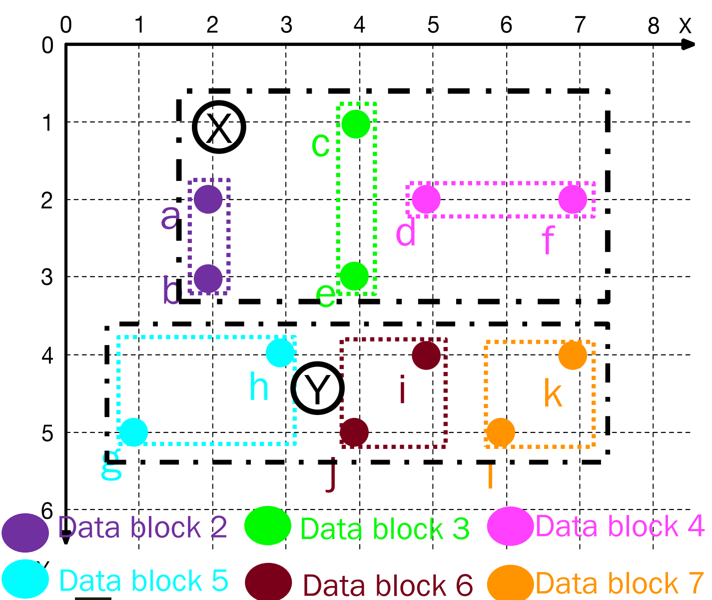
Nested loop with Index for Inner Loop



Query:

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

For each block of fire stations, create MBR with length of 1.

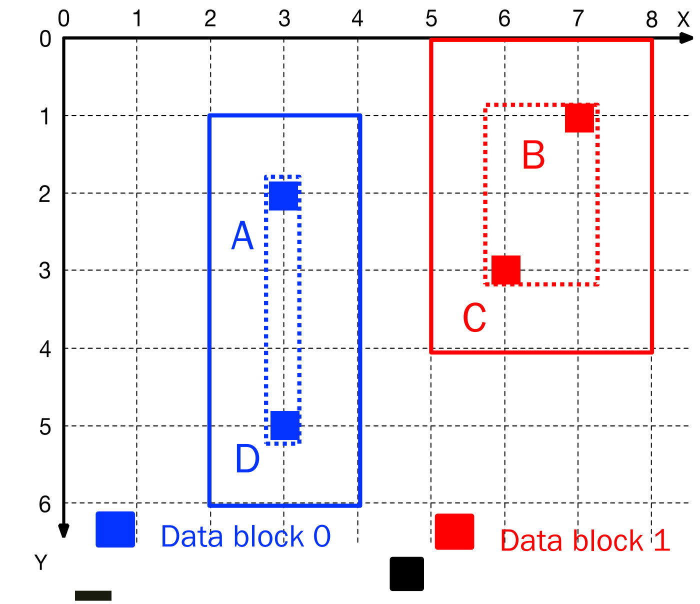


Algorithm:

For each MBR M_{fs} of fire-station blocks

Find overlapped blocks in the R-tree

Nested loop with Index for Inner Loop



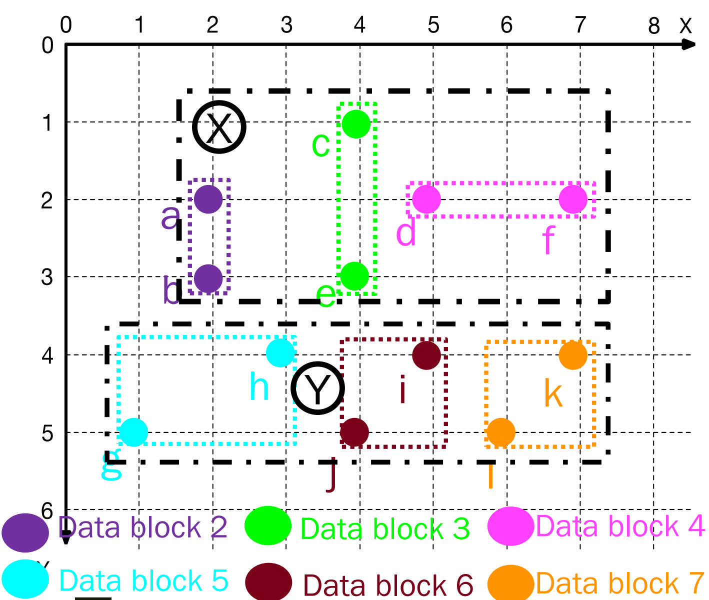
Query:

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

Algorithm:

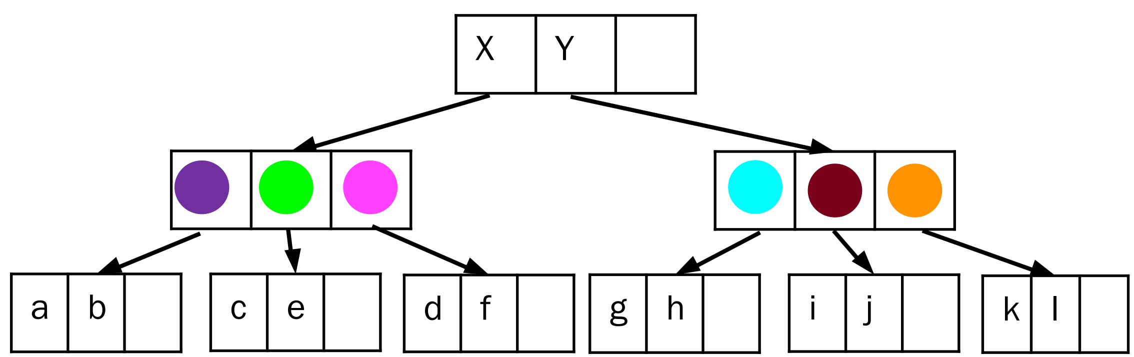
For each MBR M_{fs} of fire-station blocks

Find overlapped blocks in the R-tree

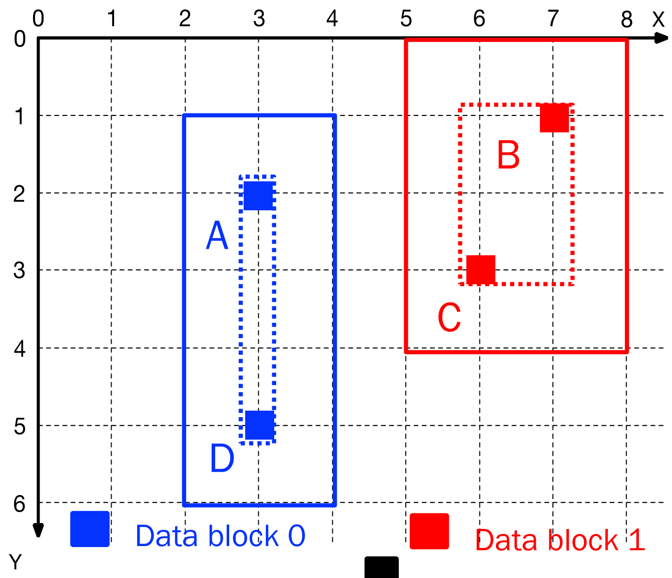


Block 0: Root -> X -> ●, ● -> leaf objs

-> Y -> ●, ● -> leaf objs



Nested loop with Index for Inner Loop



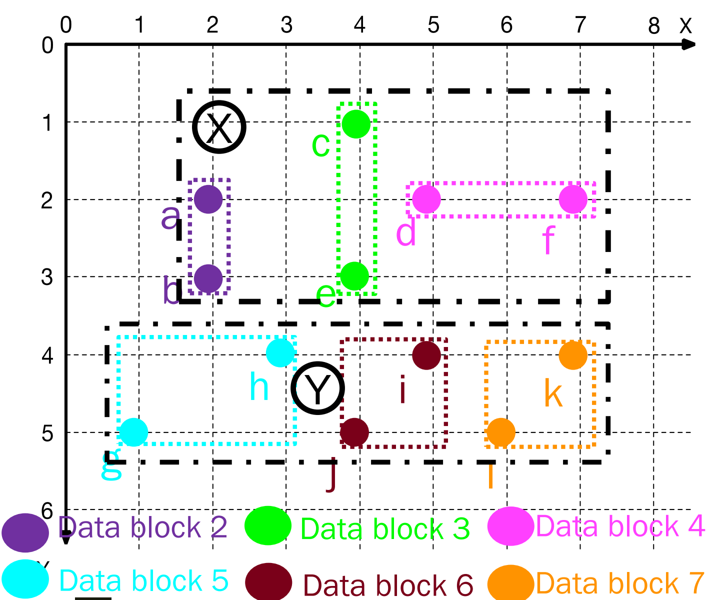
Query:

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

Algorithm:

For each MBR M_{fs} of fire-station blocks

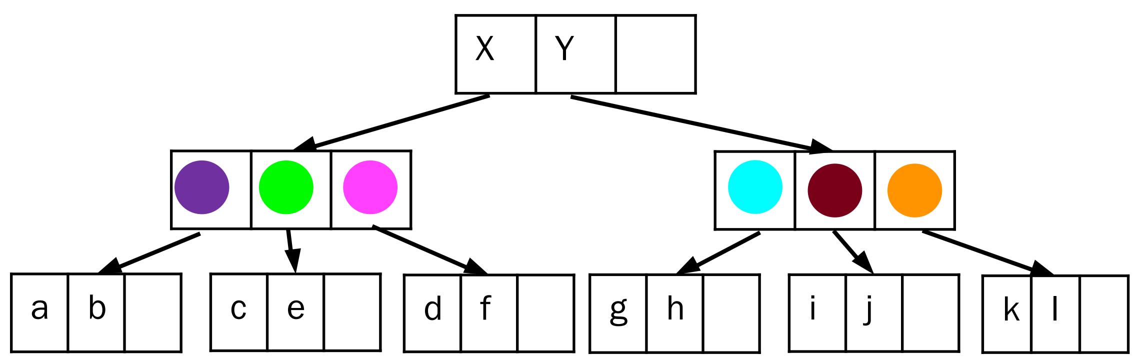
Find overlapped blocks in the R-tree



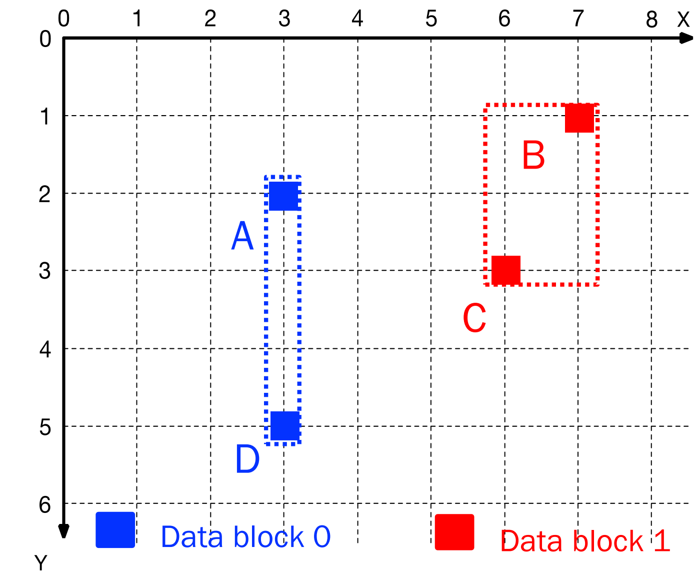
Block 1:

Root -> X -> -> leaf objs

-> Y -> , -> leaf objs



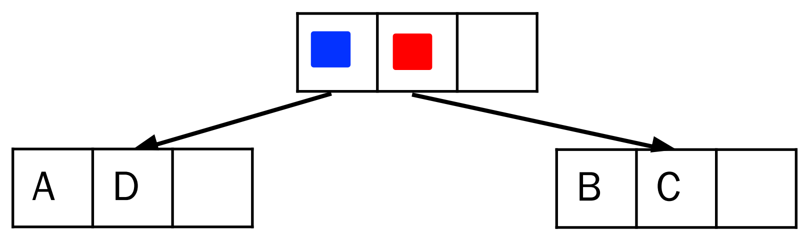
Tree Matching strategy

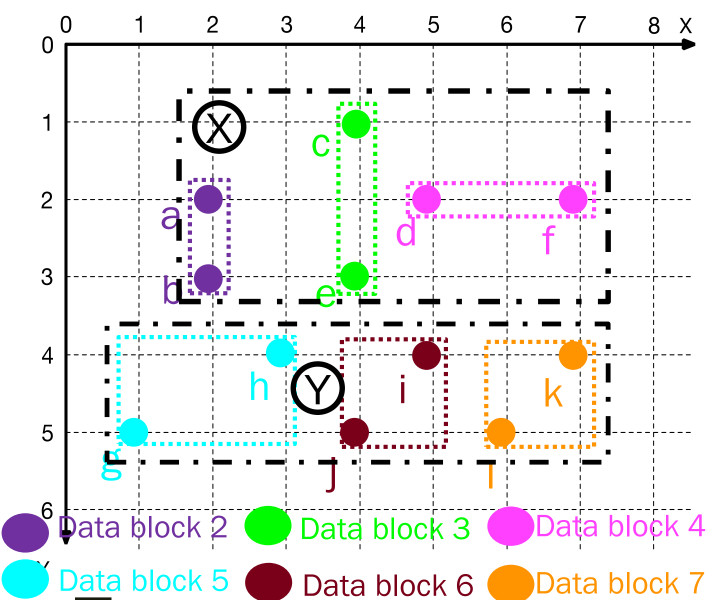


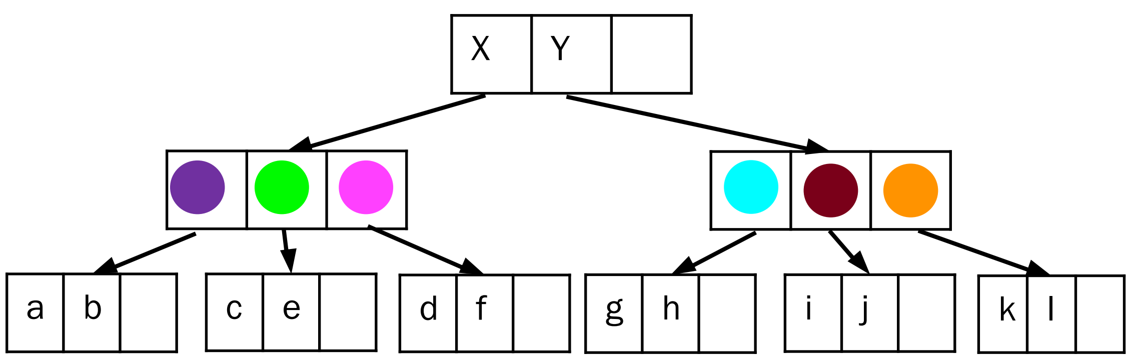
Query:

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

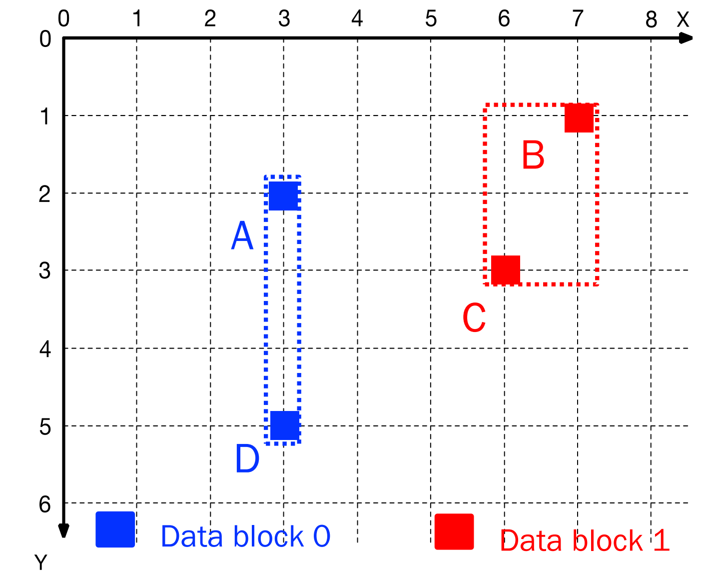
Suppose an R-tree (primary index) is available for fire stations and houses, respectively.

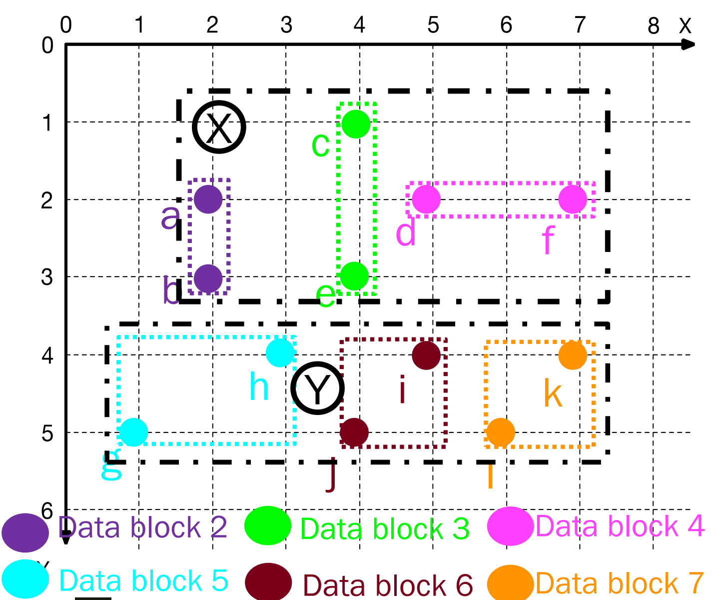






Tree Matching strategy





Query:

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

Suppose an R-tree (primary index) is available for fire stations and houses, respectively.

Algorithm:

Tree Match(Rtree1 node1, Rtree2 node2)

For all MBR M2 of R-tree2 node2

    For all MBR M1 of R-tree1 node1

        IF (if mindist(M2,M1) =< 1)

            If (node1 and node2 are leaves)

                <perform the join>

            Else if (node1 is leaf page)

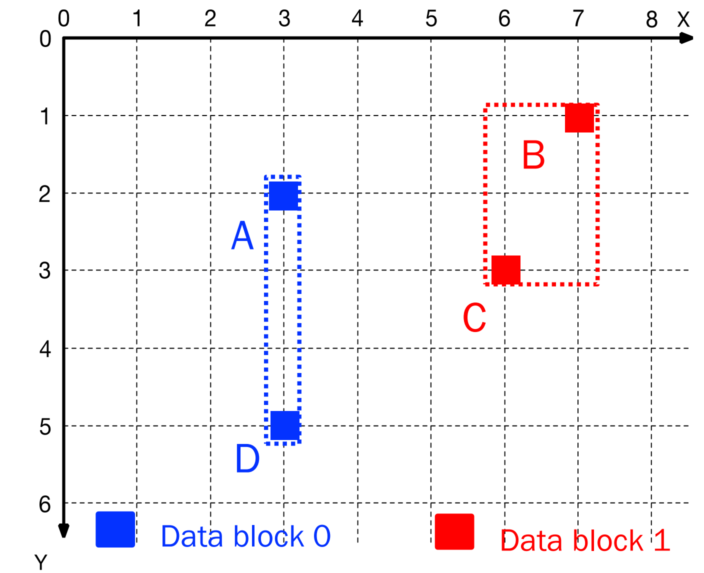
                Read child of M2

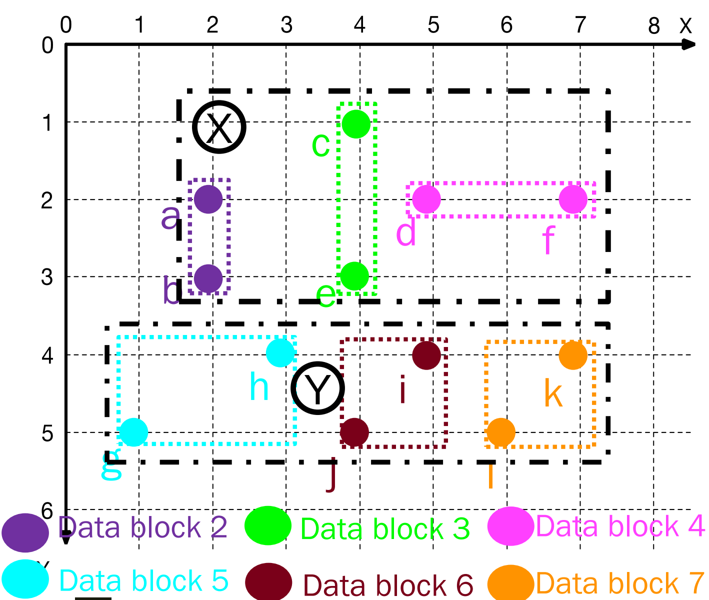
                Tree Match (node1, M2.child)

            Else if(node2 is a leaf page)

                ....

Tree Matching strategy

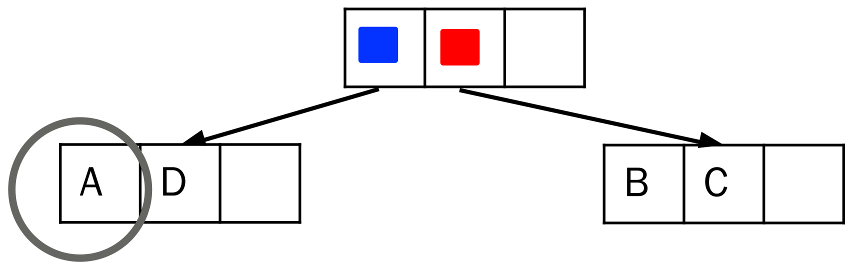


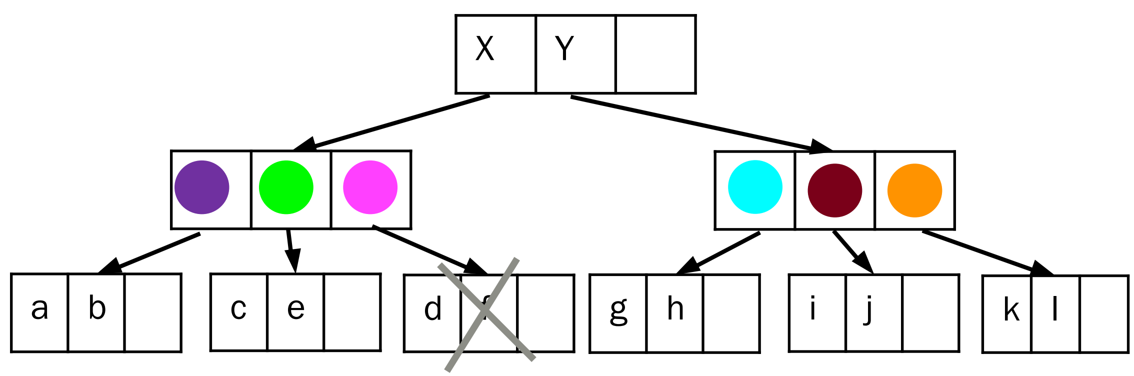


Query:

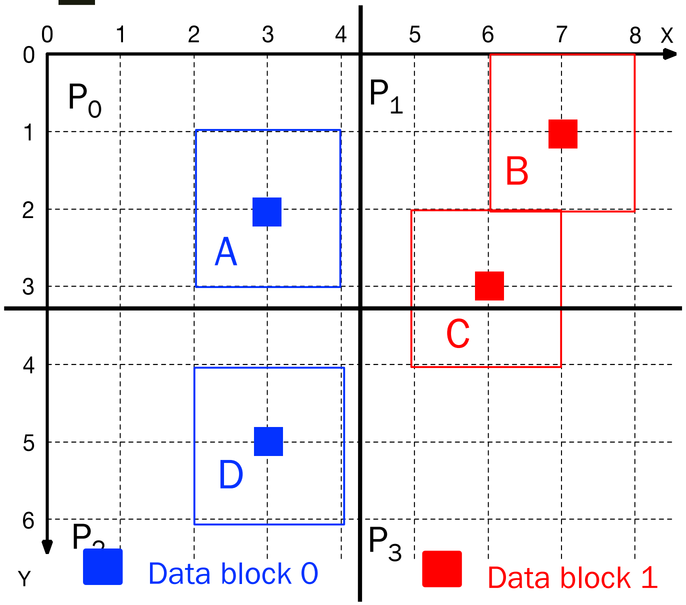
For each fire station, find all the houses within a distance <= 1

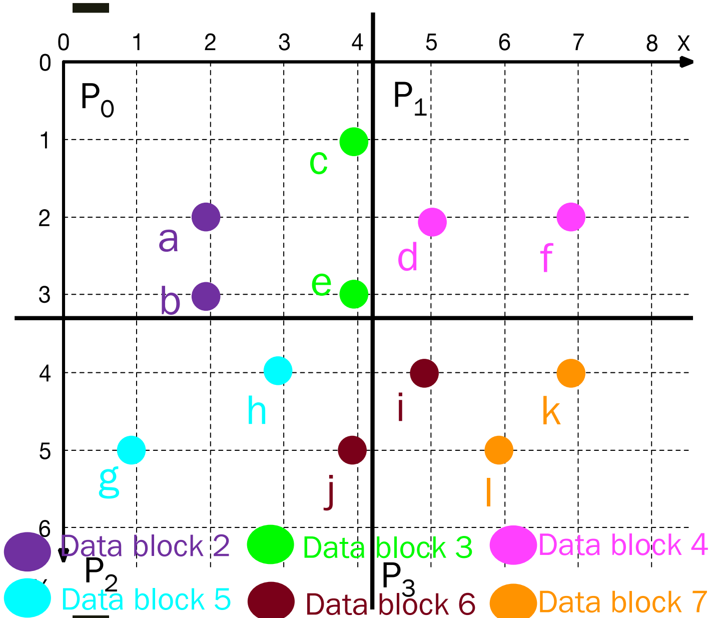
Suppose an R-tree (primary index) is available for fire stations and houses, respectively.





Partitioning based strategy





Partition the study area into 2 * 2 = 4 partitions, P_0, P_1, P_2, P_3

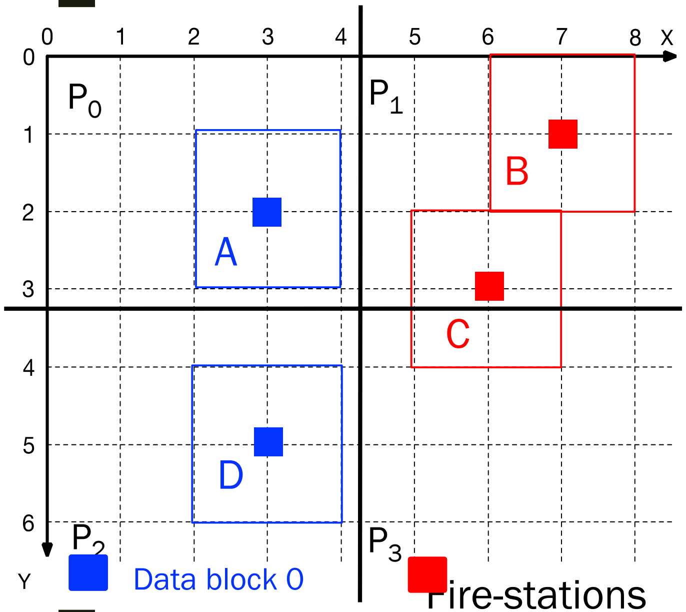
For fire station, create MBR with length of 1.

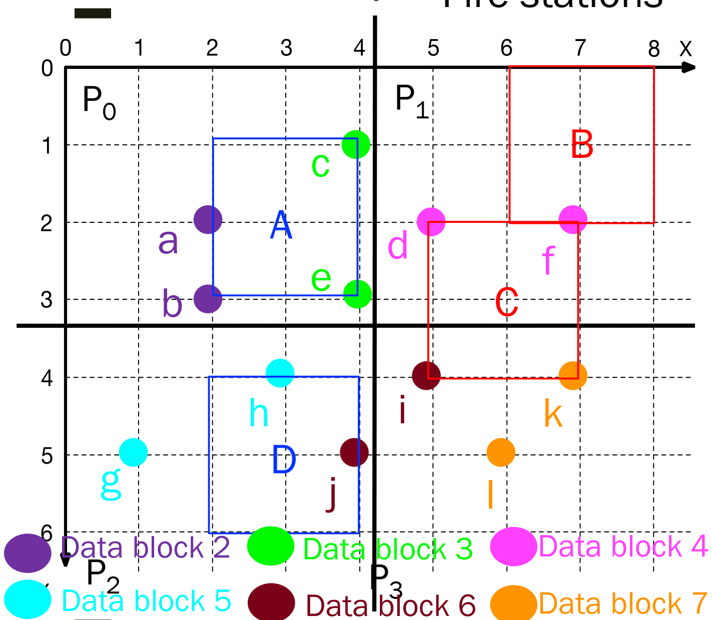
Partitioning results:

Partition	Fire-Stations	Houses
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

MBR of C in both P_1 and P_3 since it overlaps both partitions.

Partitioning based strategy





Query:

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

Algorithm:

For each partition P_i

For each MOBR M_{fs} of fire-station in P_i

Find all the houses in P_i that are overlapped with M_{fs}

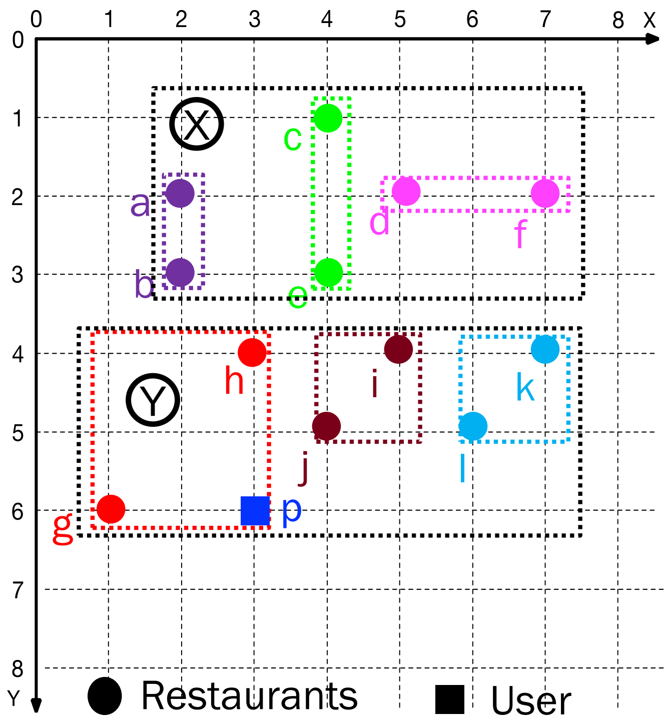
Results from filter phase:

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

Strategies for 1-Nearest Neighbor Queries

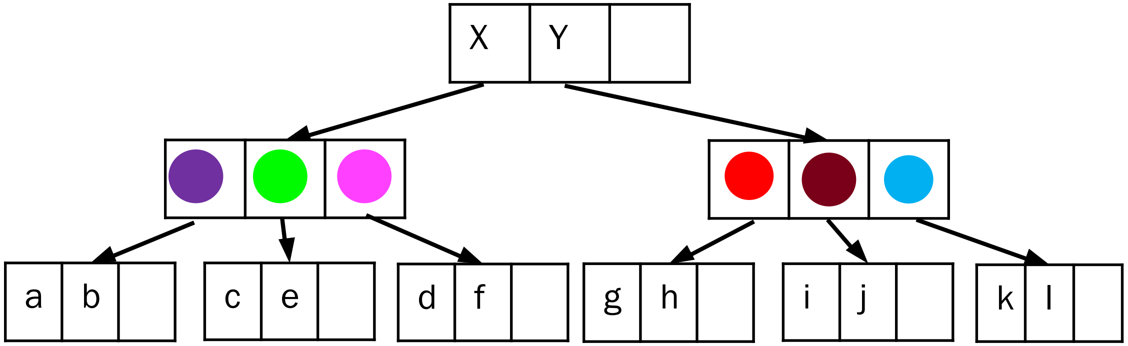
- **Recall Nearest Neighbor Example**
  - *Find the city closest to Chicago.*
  - *Return one spatial object from datafile C*
- **List of strategies**
  - ***Two phase approach***
    - Fetch C's disk sector(s) containing the location Chicago
    - M = minimum distance(Chicago, cities in fetched sectors)
    - Test all cities within distance M of Chicago (Range Query)
  - ***Single phase approach***
    - Recursive algorithm for R-tree
    - First get the closest data point
    - Then eliminate objects based on mindist to MBRs
    - Similar to K-NN algorithm on KD-trees

Two Phase Approach

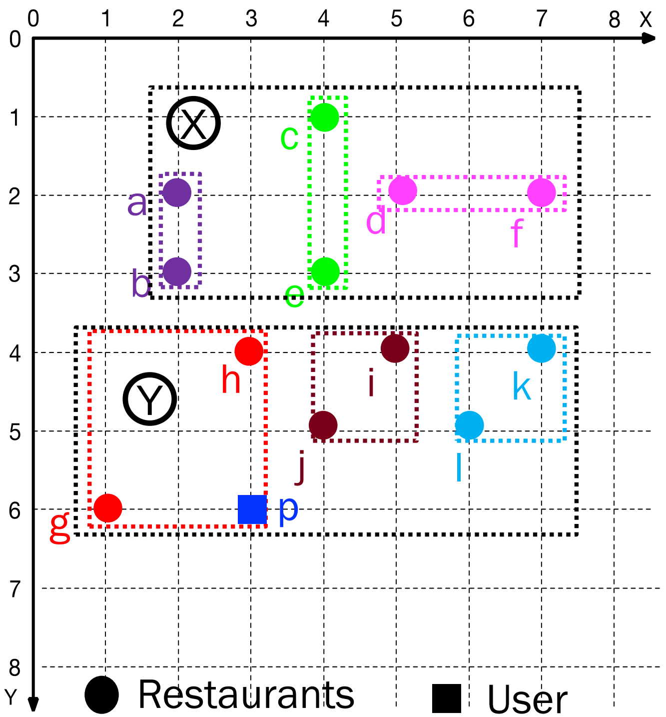


Given the location of a user p, find the nearest restaurant.
(If more than one nearest neighbors, return all results)

Suppose R-tree (primary index) is available on this dataset



Two Phase Approach

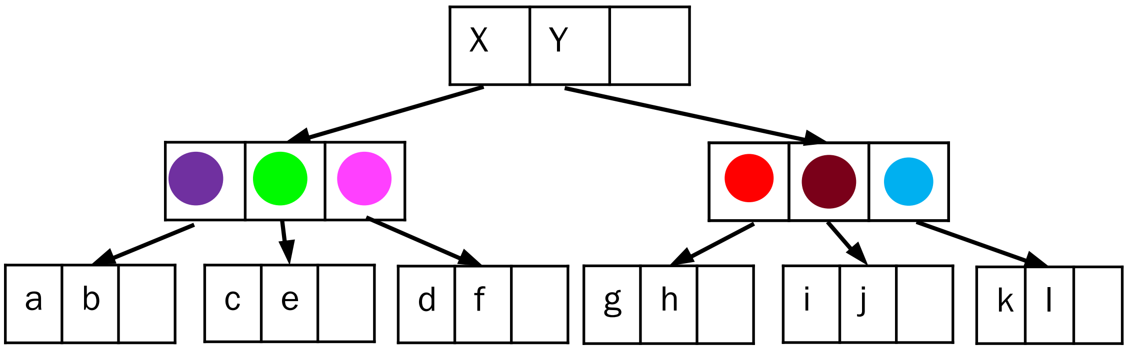


Given the location of a user p, find the nearest restaurant.
(If more than one nearest neighbors, return all results)

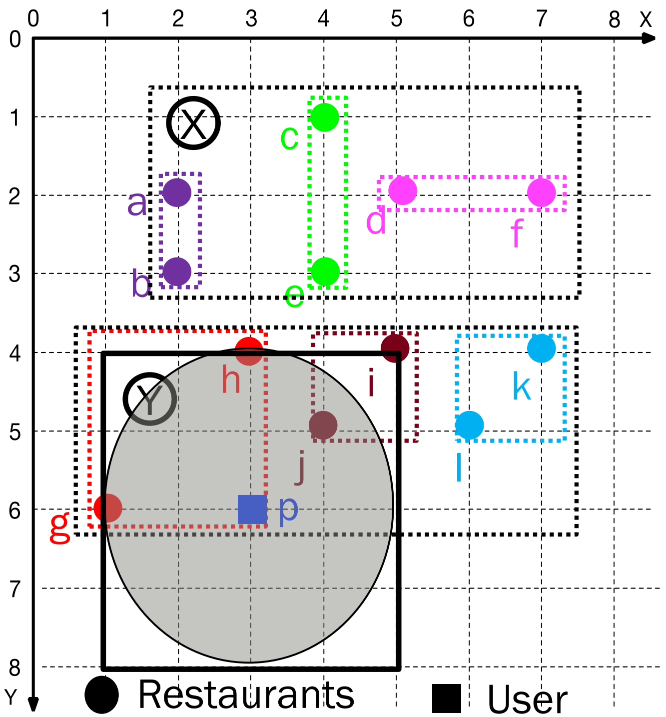
Algorithm:

Find the index leaf containing the query point p

Point g, h are the closest points to p,



Two Phase Approach



Given the location of a user p, find the nearest restaurant.
(If more than one nearest neighbors, return all results)

Algorithm:

Find the index leaf containing the query point p

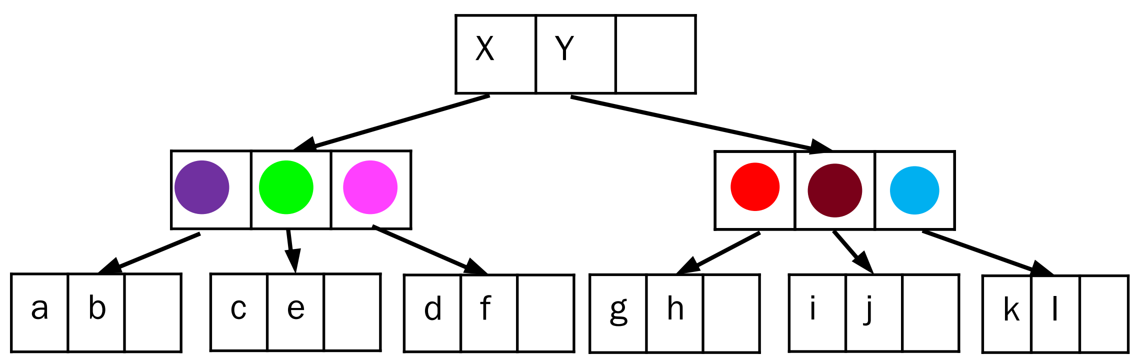
Point g, h are the closest points to p at distance d_B

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

Create the MOBR of $Circle_p$: M_p

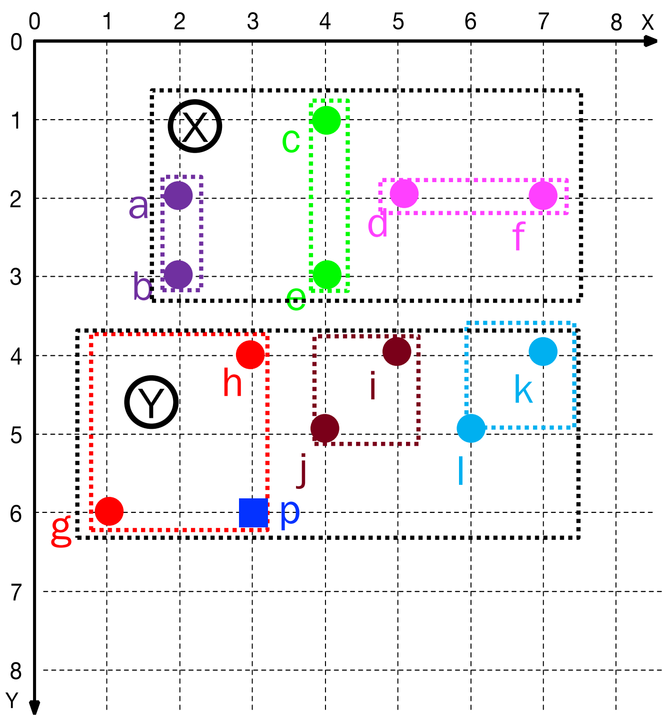
Range query: M_p, and test all points in M_p
Root -> Y -> leaves containing <g,h> and <j,i>

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

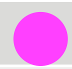
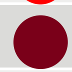


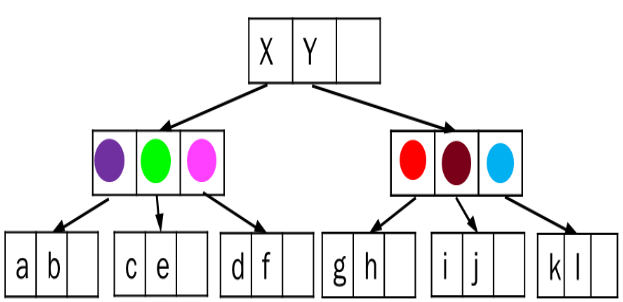
One Phase Approach – Recursive search on R-tree

Given the location of a user p, find the nearest restaurant.
(If more than one nearest neighbors, return all results)



Algorithm:

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



In the first part of the algorithm we get that
Point g, h are the closest points to p, distance = 2