Chap 6: Spatial Networks

Learning Objectives

- Learning Objectives (LO)
  - LO1: Understand the concept of spatial network (SN)
  - LO2 : Learn data models for SN
  - LO3: Learn about query languages and query processing
    - Query building blocks
    - Processing strategies
  - LO4: Learn about trends
- Focus on concepts not procedures!
- Mapping Sections to learning objectives
  - LO1 - 6.1
  - LO2 - 6.2
  - LO3 - 6.3, 6.4
  - LO4 - 6.5

6.4 Query Processing for Spatial Networks

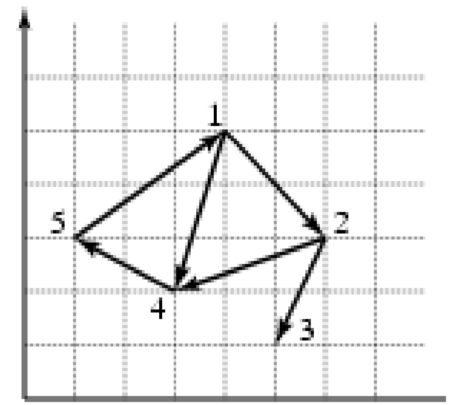
- Connectivity(A, B)
  - Is node B reachable from node A?
- Shortest path(A, B)
  - Identify least cost path from node A to node B

Strategies for Graph Transitive Closure

- Q? Assumption on storage area holding graph tables
    - Main memory algorithms
    - Disk based external algorithms
- Representative strategies for single pair shortest path
  - Main memory algorithms
    - Connectivity: Breadth first search, depth first search
    - Shortest path: Dijktra's algorithm, Best first algorithm
  - Disk based, external
    - Shortest path - Hierarchical routing algorithm
    - Connectivity strategies are already implemented in relation DBMS

Connectivity with SQL

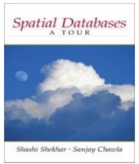
```
With Recursive C(source, dest, path, circle) as
(
    Select source, dest, ARRAY[sp.dest], false From Edge
    Union All
    Select Edge.source, C.dest, Edge.dest || path, Edge.dest=any(path)
    From Edge, C
    Where Edge.dest = C.source And Not circle
)
Select source || path as "Path"
From C
Where source = 1 And dest = 5;
```



Edge (S)

source	dest	distance
1	2	$\sqrt{8}$
1	4	$\sqrt{10}$
2	3	$\sqrt{5}$
2	4	$\sqrt{10}$
4	5	$\sqrt{5}$
5	1	$\sqrt{18}$

Path
"{1,4,5}"
"{1,2,4,5}"



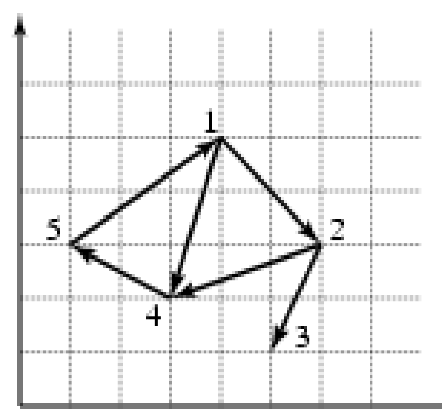
Shortest Path: Dijkstra

```
function Dijkstra(Graph, source):
    for each vertex v in Graph:           // Initializations
        dist[v] := infinity               // Unknown distance function from source to v
        previous[v] := undefined          // Previous node in optimal path from source
    dist[source] := 0                     // Distance from source to source
    Q := the set of all nodes in Graph
    // All nodes in the graph are unoptimized - thus are in Q
    while Q is not empty:                 // The main loop
        u := vertex in Q with smallest dist[]
        if dist[u] = infinity:
            break                         // all remaining vertices are inaccessible from source
        remove u from Q
        for each neighbor v of u:         // where v has not yet been removed from Q.
            alt := dist[u] + dist_between(u, v)
            if alt < dist[v]:             // Relax (u,v,a)
                dist[v] := alt
                previous[v] := u
    return dist[]
```

Example

- Consider shortest_path(1,5) for graph in Figure
- Iteration 1
  - select 1, cost(2) = sqrt(8), prev(2) = 1, cost(4) = sqrt(10), prev(4) = 1
- Iteration 2
  - select 2, c(3) = c(2) + dist(2,3) = sqrt(8)+sqrt(5), prev(3) = 2, no update c(4)
- Iteration 3
  - select 4, c(5) = c(4) + dist(4,5) = sqrt(10)+sqrt(5), prev(5) = 4;
  - Terminate (node 5 has been reached)

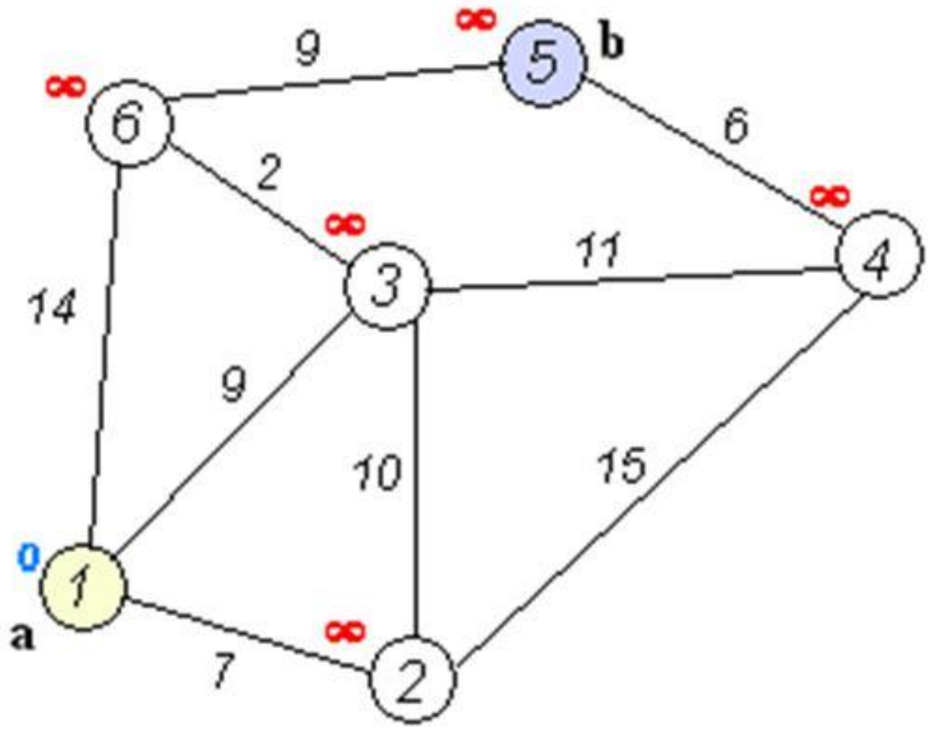
- Answer is the path 1->4->5 (prev(5) = 4, prev(4) = 1) with cost sqrt(10)+sqrt(5)



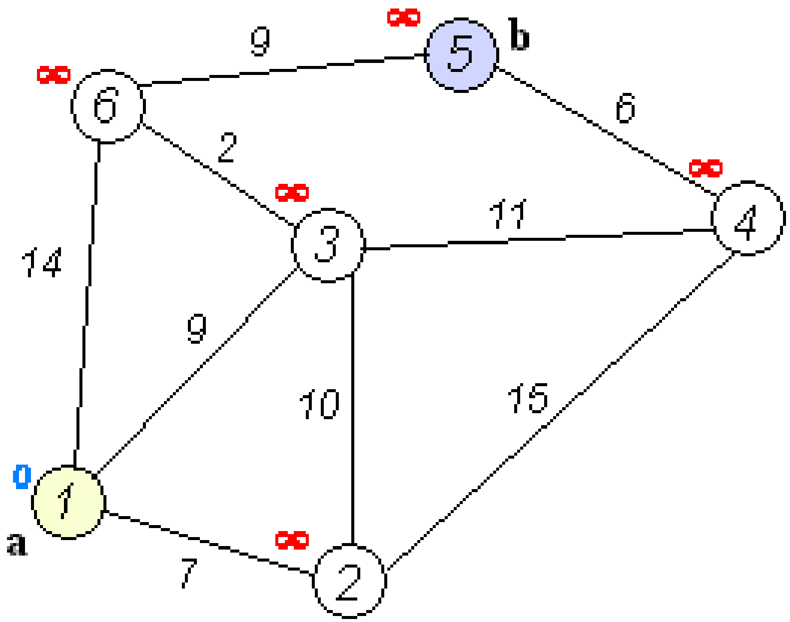
Edge (S)

source	dest	distance
1	2	$\sqrt{8}$
1	4	$\sqrt{10}$
2	3	$\sqrt{5}$
2	4	$\sqrt{10}$
4	5	$\sqrt{5}$
5	1	$\sqrt{18}$

Exercise: shortest_path(1,5)



Exercise: shortest_path(1,5)



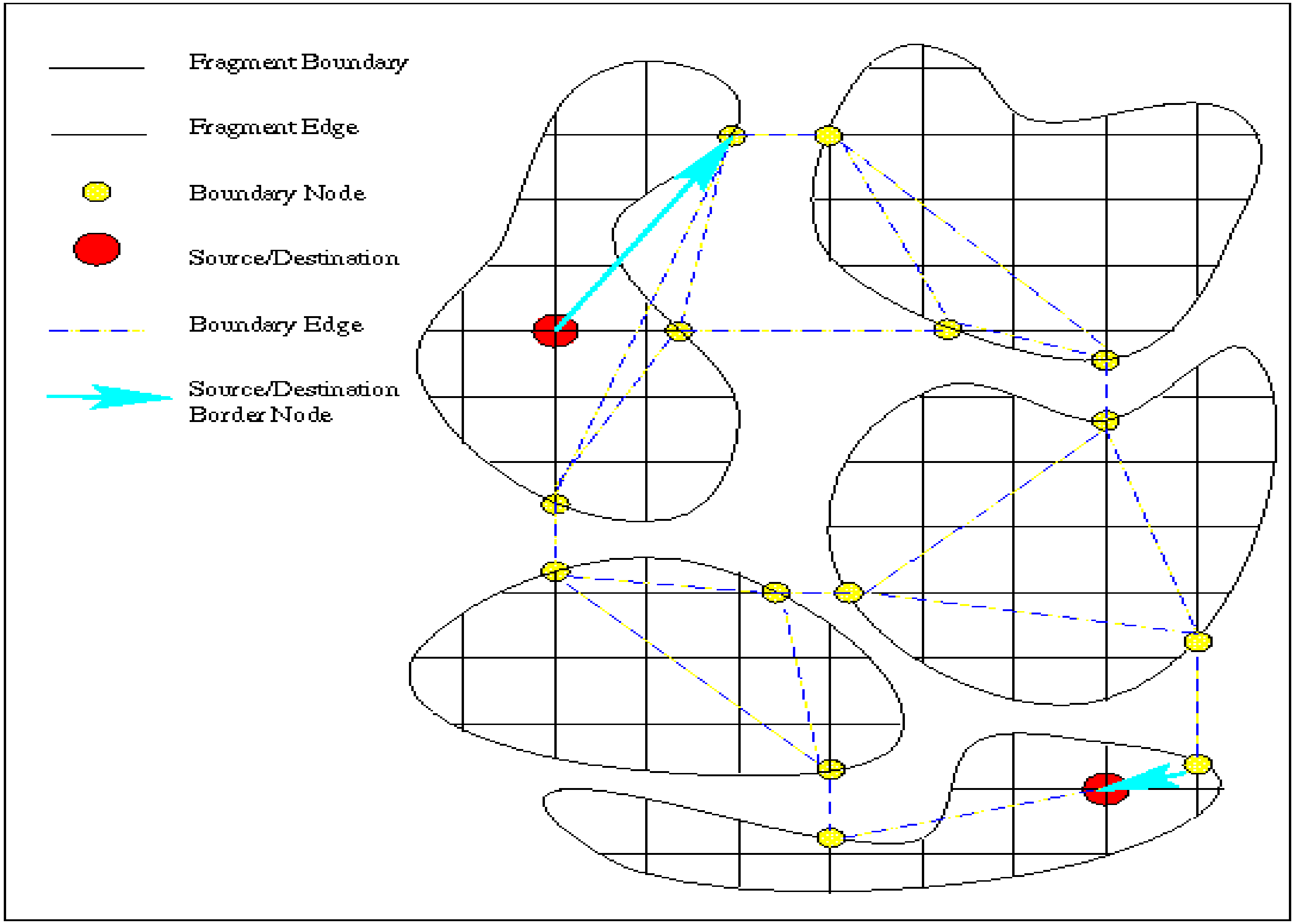
Shortest Path in PostGIS

- **pgRouting** Project: http://pgrouting.postlbs.org/
- http://pgrouting.postlbs.org/wiki/pgRoutingDocs

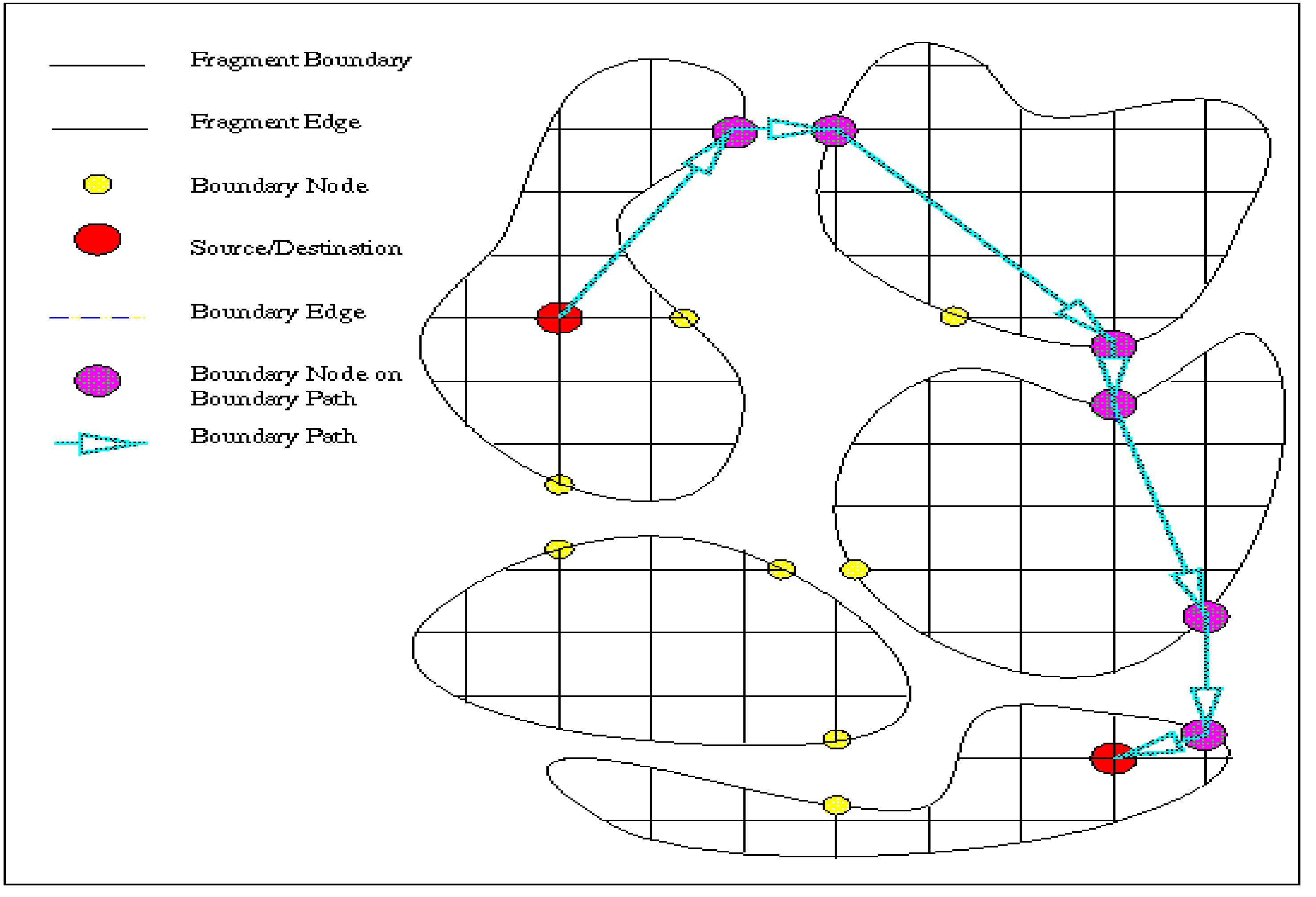
6.4.2 Shortest Path: Alternative Strategies

- Dijktra's and Best first algorithms
    - Work well when entire graph is loaded in main memory
    - Otherwise their performance degrades substantially
- Hierarchical Routing Algorithms
    - Works with graphs on secondary storage
    - Loads small pieces of the graph in main memories
    - Can compute least cost routes
- Key ideas behind Hierarchical Routing Algorithm
    - Fragment graphs - pieces of original graph obtained via node partitioning
    - Boundary nodes - nodes of with edges to two fragments
    - Boundary graph - a summary of original graph
        - Contains Boundary nodes
        - Boundary edges: edges across fragments or paths within a fragment

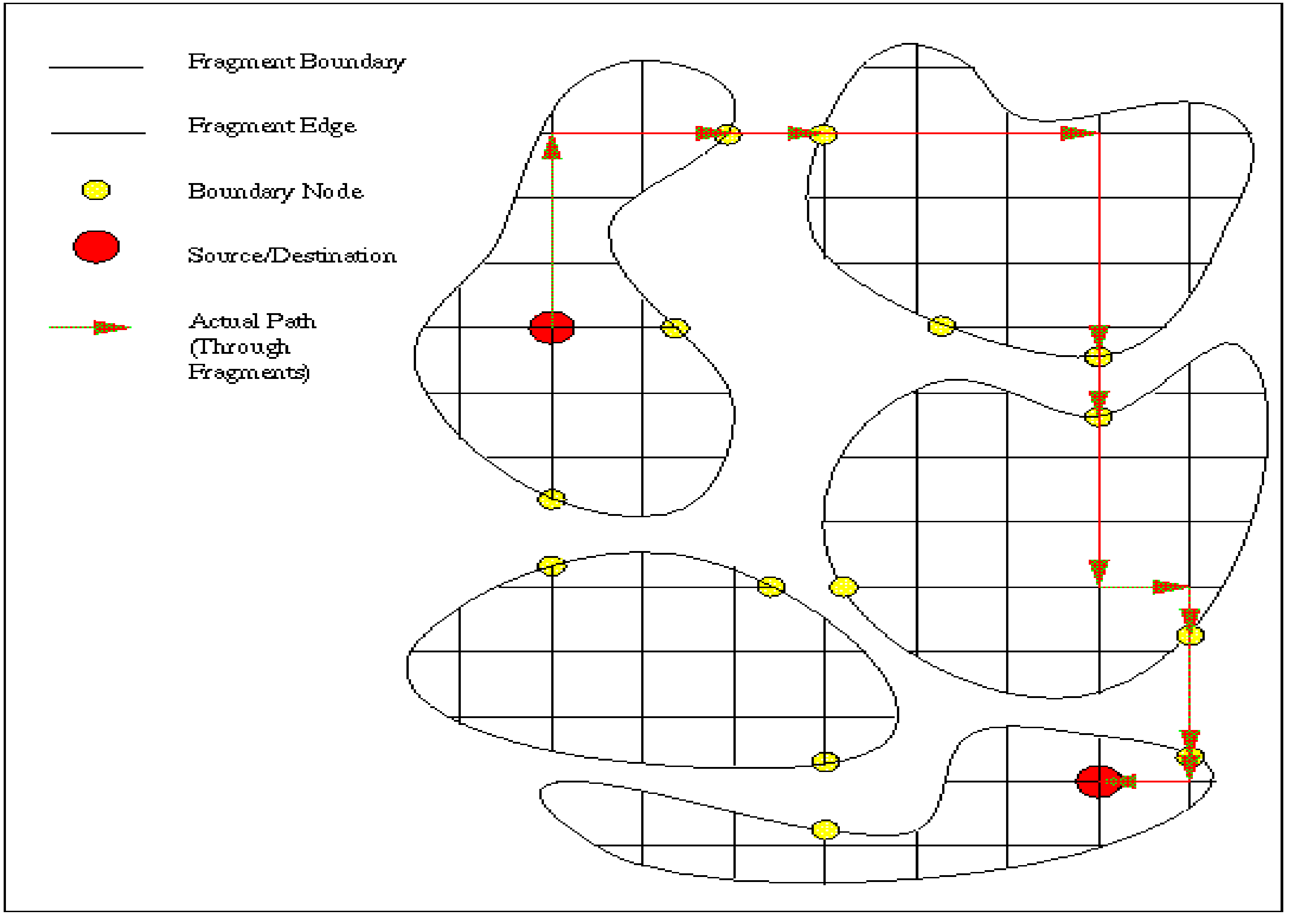
Hierarchical Routing



Hierarchical Routing



Hierarchical Routing

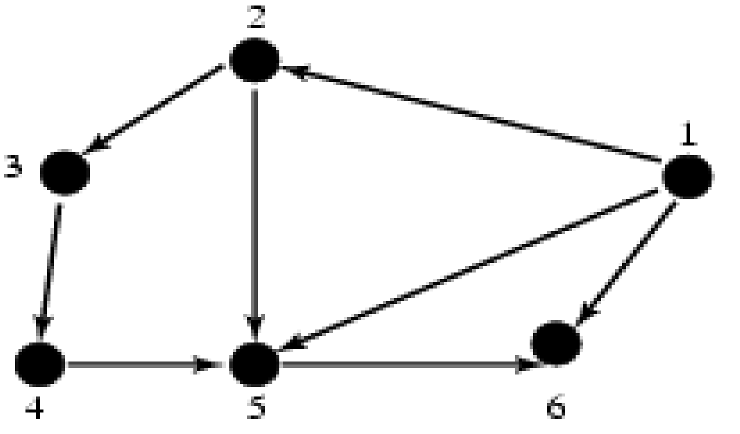


Learning Objectives

- Learning Objectives (LO)
  - LO1: Understand the concept of spatial network (SN)
  - LO2 : Learn about data models for SN
  - LO3: Learn about query languages and query processing
  - LO4: Learn about trends
    - Storage methods for SN
- Focus on concepts not procedures!
- Mapping Sections to learning objectives
  - LO1 - 6.1
  - LO2 - 6.2
  - LO3 - 6.3, 6.4
  - LO4 - 6.5

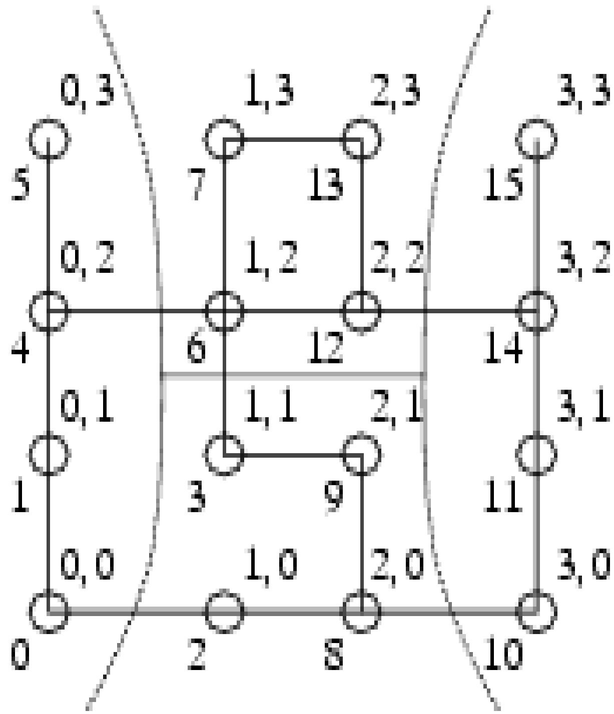
Graph Based Storage - Partitioning

- Insight:
  - CRR = Pr. (node-pairs connected by an edge are together in a disk sector)
- Example:
  - Consider disk sector hold 3 node records
  - 2 sectors are (1, 2, 3), (4,5,6)
  - CRR = 4/8
  - 2 sectors are (1, 5, 6), (2,3,4)
  - CRR = 5/8



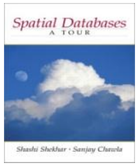
Graph Based Storage - Partitioning

- Example



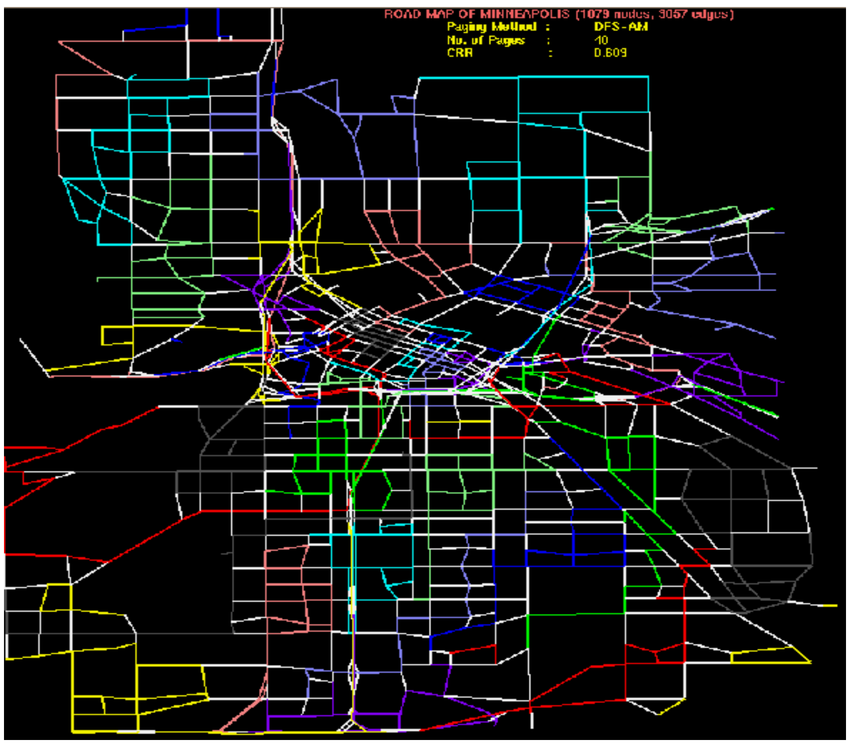
Partitioning algorithms

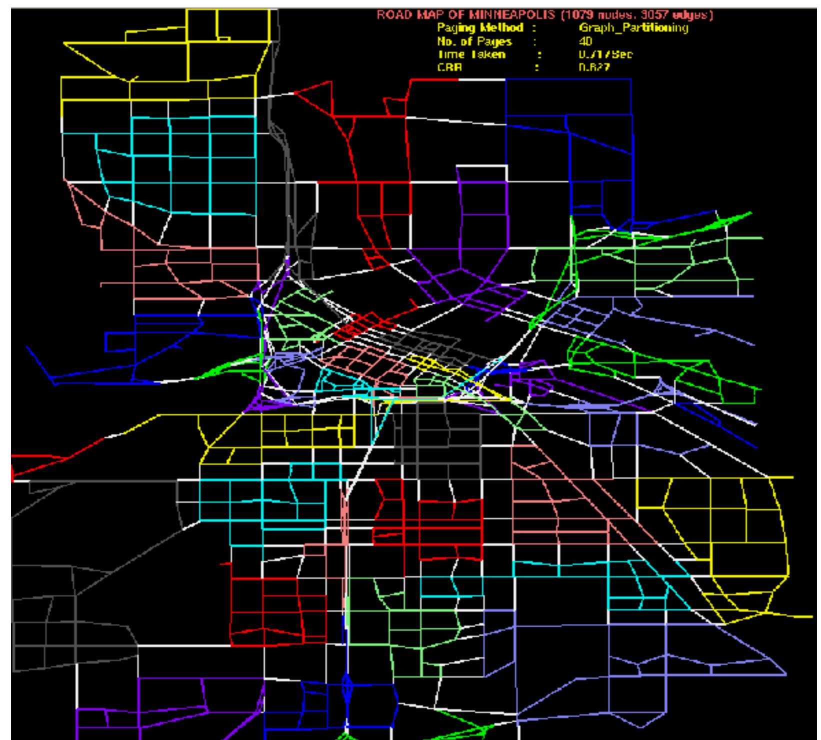
- The graph partitioning problem consists of dividing a graph into pieces, such that the pieces are:
  - of about the same size **(in our case: need also to consider the fixed page size constraint)**
  - there are few connections between the pieces
- Graph partitioning is known to be NP-complete
  - Fast heuristics work well in practice
  - http://www.sandia.gov/~bahendr/partitioning.html
- Large scale? (million nodes)
  - Yes!



Graph Based Storage - Partitioning

- Large-scale example: Consider two paging of a spatial network
  - non-white edges => node pair in same page
  - File structure using node partitions on right is preferred
    - it has fewer white edges => higher CRR





Summary

- Spatial Networks are a fast growing applications of SDBs
- Spatial Networks are modeled as graphs
- Graph queries, like shortest path, are transitive closure
  - not supported in relational algebra
  - SQL features for transitive closure: CONNECT BY, WITH RECURSIVE
- Graph Query Processing
  - Building blocks - connectivity, shortest paths
  - Strategies - Best first, Dijktra's and Hierachical routing
- Storage and access methods
  - Minimize CRR