

Chapter 6: Spatial Networks

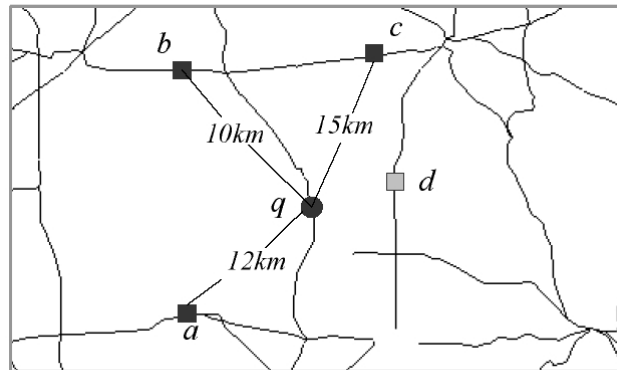


Motivation: Navigation Systems

- In-vehicle navigation systems
 - Offered on many high-end cars
 - Services:
 - map destination given a street address
 - compute the shortest route to a destination
 - Help drivers follow selected route
- Recall that many maps and GIS were made for navigation
 - 16th century shipping, trade and treasure hunts
 - Maps were used to find destination and avoid hazards
- Navigation and transportation are important applications!



Motivation: Navigation Systems (Cont.)



How important are Spatial Networks?

- Web based navigation services attract large audience
 - Example: Google Maps, Yahoo! maps, etc.
 - Rated among most popular internet services!
- Transportation sector among application of GIS
 - Already a major segment of GIS market
 - Is among three fastest growing segments
- Spatial networks in business and government
 - Largest companies in the world in following sectors of economy
 - Car, ship, airplane, oil, electrical power, natural gas, telephone
 - Logistics groups in Retailers (e.g. Wal-mart), Manufacturers (e.g. GE)
 - Many departments of government
 - Transportation (USDOT), Logistics of deployment (USDOD),
 - City (utilities like water, sewer, ...)



6.1 Example Spatial Networks

Road, River, Railway networks



(b) River Network

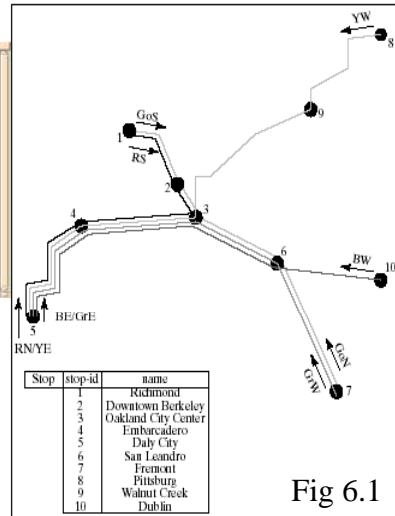


Fig 6.1



Queries on Example Networks

•Road Network

1. Find shortest path from my current location to a destination.
2. Find nearest hospital by distance along road networks.
3. Find shortest route to deliver goods to a list of retail stores.
4. Allocate customers to nearest service center using distance along roads

•Railway network

1. Find the number of stops on the Yellow West (YW) route.
2. List all stops which can be reached from Downtown.
3. Find the last stop on the Blue West (BW) route.

•River network

1. List the names of all direct and indirect tributaries of the Mississippi river
2. List the direct tributaries of the Colorado.
3. Which rivers could be affected if there is a spill in river P1?

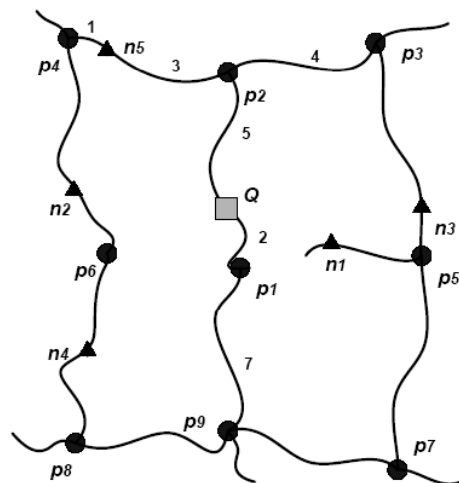


Spatial network query - NN

- Nearest neighbor queries.
 - Euclidean distance NN
 - Network distance NN (+ realistic, - complicated)
 - More accurate solutions ?
- Technology advances:
 - Personal locator systems (e.g., GPS)
 - Wireless communication (e.g., IEEE 802.11x)
 - Peer-to-Peer data sharing



Incremental Network Expansion



Legends

- ▲ Point of interest
- Junction

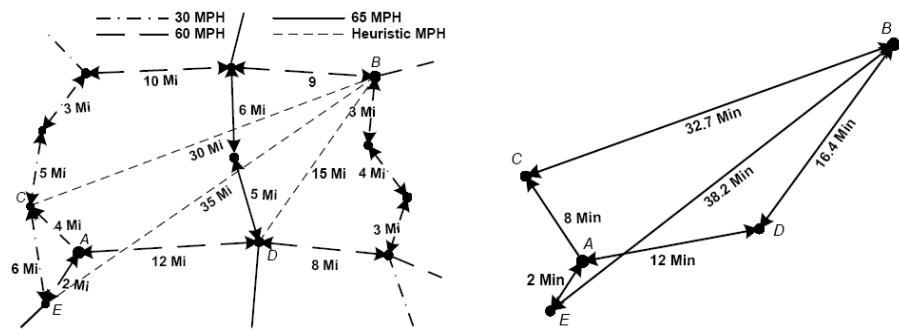


Local-based Solution

- Short range ad hoc communication technology is utilized to exchange traffic information in a local area.
- Steps:
 - Compute a NN via TTN NN queries
 - Incrementally update local traffic information and Cost function = g (actual cost) + h (heuristic cost)
 - Greedily find the shortest path to the NN



Local-based Solution (Cont.)





Traffic Events

- Category 1 – Congestion events
 - The real traffic speed is much lower than the speed limit.
- Category 2 – Detour events
 - A road segment is closed and the mobile host has to detour.
- Category 3 – Closure events
 - The selected POI is closed.
- Category 4 – Recovery events
 - A mobile host can recover its local TTN from previous events.



Reaction to Traffic Events

- Categories 1 and 2 – Update local TTN or recalculate the driving time from current location to the destination (D) and Launch a TTN NNQ with the remaining travel time as the [search upper bound](#).
- Category 3 – If the closed POI is D, launch a travel time network NNQ.
- Category 4 – Update local TTN and launch a TTN NNQ with the remaining travel time as the search upper bound.



Global-based Adaptive NN Queries

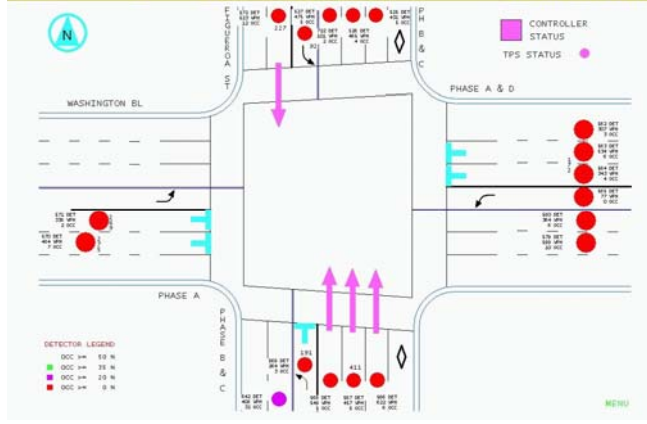
- A **Traffic Information Server (TIS)** which aggregates real-time traffic events and road segment speeds.
- Steps:
 - Executes the TTN NN query algorithm for retrieving a POI as the current destination D.
 - Update TTN and/or D with new traffic information from TIS.
 - Follow the currently selected route until the mobile host reaches D.



Global-based Adaptive NN Queries (Cont.)

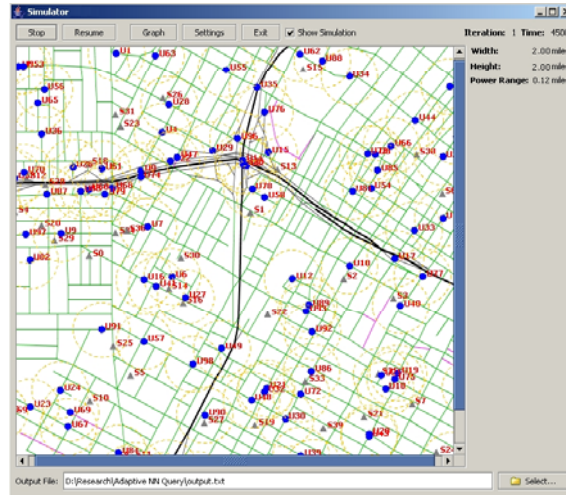
INTERVAL	1	2	3	4	5	6	7	8	9	10	11	12	13	14
PHASE C	A	A	A	A	B	B	B	C	C	C	D	D	D	D
INT. TYPE	FIX	VAR	FIX	FIX	FIX	VAR	FIX	FIX	VAR	FIX	FIX	VAR	FIX	FIX
MINIMUM	5	7	13	5	3	12	4	5	2	17	5	3	5	4
CURRENT	5	7	13	5	3	12	4	5	2	17	5	3	5	4
INT. TRM										15				
MINIMUM										20				
CURRENT										19				

INTERSECTION SP-091			
WASHINGTON BL			
FIGUEROA ST			
CON: ONLINE	CLC	CLC	ADP
N/A	OFF	OFF	OFF
2070	SCN: 2	CM: 13-2	
CLC	OFF	FLN	TPS
90	41	3	TOD: 58





Simulator Implementation



6.2 Conceptual Data Models

- Conceptual Data Model for Spatial Networks
 - A graph, $G = (V, E)$
 - V = a finite set of vertices
 - E = a set of edges E , between vertices in V
- Example: two graph models of a roadmap
 - 1. Nodes = road-intersections, edges = road segment between intersections
 - 2. Nodes = roads (e.g. Route 66), edge(A, B) = road A crosses road B
- Classifying graph models
 - Do nodes represent spatial points? - spatial vs. abstract graphs
 - Are vertex-pair in an edge order? - directed vs. undirected



6.2 Logical Data Model - Data types

- Common data types
 - Vertex: attributes are label, isVisited, (location for spatial graphs)
 - DirectedEdge : attributes are start node, end node, label
 - Graph : attributes are setOfVertices, setOfDirectedEdges, ...
 - Path : attributes are sequenceOfVertices



6.2 Logical Data Model - Operations

- Low level operations on a Graph G
 - IsDirected - return true if and only if G is directed
 - Add, AddEdge - adds a given vertex, edge to G
 - Delete, DeleteEdge - removes specifies node (and related edges), edge from G
 - Get, GetEdge - return label of given vertex, edge
 - Get-a-successor, GetPredecessors - return start or end vertex of an edge
 - GetSuccessors - return end vertices of all edge starting at a given vertex
- Building blocks for queries
 - shortest_path(vertex start, vertex end)
 - shortest_tour(vertex start, vertex end, setOfVertices stops)
 - locate_nearest_server(vertex client, setOfVertices servers)
 - allocate(setOfVertices servers)

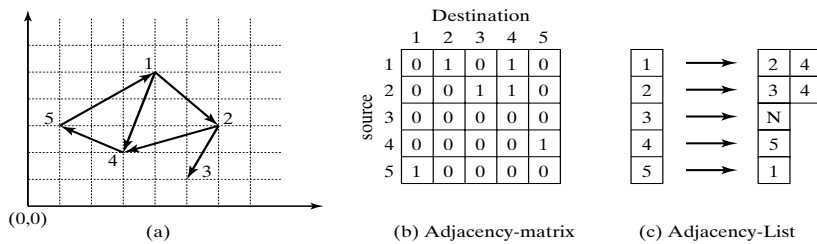


6.2 Physical Data Models

- Categories of record/file representations
 - Main memory based
 - Disk based
- Main memory representations of graphs
 - Adjacency matrix $M[A, B] = 1$ if and only if edge(vertex A, vertex B) exists
 - Adjacency list : maps vertex A to a list of successors of A
- Disk based
 - Normalized - tables, one for vertices, other for edges
 - Denormalized - one table for nodes with adjacency lists



6.2.2 Physical Data Models



Node (R)			Edge (S)			Denormalized Node Table	
id	x	y	source	dest	distance	id	Successors
1	4.0	5.0	1	2	$\sqrt{8}$	1	(2,4)
2	6.0	3.0	1	4	$\sqrt{10}$	2	(3,4)
3	5.0	1.0	2	3	$\sqrt{5}$	3	()
4	3.0	2.0	2	4	$\sqrt{10}$	4	(5)
5	1.0	3.0	4	5	$\sqrt{5}$	5	(1,2)
			5	1	$\sqrt{18}$		(1)
							(4)

(d) Node and Edge Relations

(e) Denormalized Node Table



6.3 Query Languages For Graphs

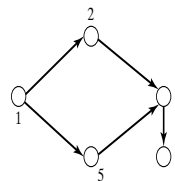
- Recall Relation algebra (RA) based languages
 - Can not compute transitive closure (Section 3.1)
 - SQL3 provides support for transitive closure on graphs
 - supports shortest paths
- SQL support for graph queries
 - SQL2 - CONNECT clause in SELECT statement
 - For directed acyclic graphs, e.g. hierarchies
 - SQL 3 - WITH RECURSIVE statement
 - Transitive closure on general graphs
 - SQL 3 -user defined data types
 - Can include shortest path operation!



Concept of Transitive Closure

- Consider a graph $G = (V, E)$
- Let $G^* =$ Transitive closure of G
- Then $T =$ graph (V^*, E^*) , where
 - $V^* = V$
 - (A, B) in E^* if and only if there is a path from A to B in G .

- Example in Figure 6.4
 - G has 5 nodes and 5 edges
 - G^* has 5 nodes and 9 edges
 - Note edge $(1,4)$ in G^* for
 - path $(1, 2, 3, 4)$ in G .

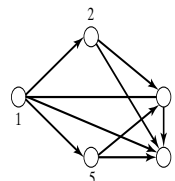


(a) Graph G

Fig 6.4

R	
SOURCE	DEST
1	2
1	5
2	3
3	4
5	3

(b) Relation form



(c) Transitive closure (G) = Graph G^*

X	
SOURCE	DEST
1	2
1	5
2	3
3	4
5	3
1	3
2	4
5	4
1	4

(d) Transitive closure in relational form



6.3.2 SQL2 Connect Clause

- Syntax details
 - FROM clause a table for directed edges of an acyclic graph
 - PRIOR identifies direction of traversal for the edge
 - START WITH specifies first vertex for path computations
- Semantics
 - List all nodes reachable from first vertex using directed edge in specified table
 - Assumption - no cycle in the graph!
 - Not suitable for train networks, road networks
- Example
 - Query: List direct and indirect tributaries of Mississippi
 - Table = FallsInto(source, dest) defined in Table 6.3 (pp. 157)
 - edge direction is from "source" to "dest" (source falls into dest)
 - start node is Mississippi (river id = 1)



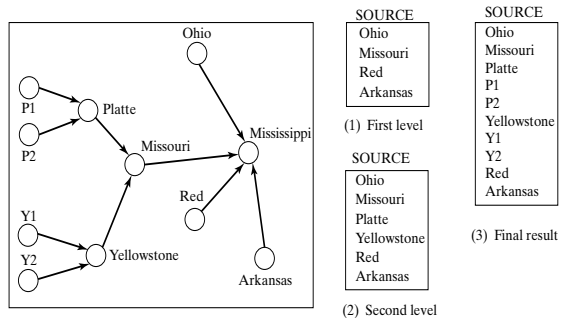
SQL Connect Clause - Example

- SQL expression on right
- Execution trace of paths
 - starts at vertex 1 (Mississippi)
 - adds children of 1
 - adds children of Missouri
 - adds children of Platte
 - adds children of Yellowstone
- Result has edges
 - from descendants
 - to Mississippi

```

SELECT source
FROM FallsInto
CONNECT BY PRIOR source = dest
START WITH dest = 1

```



(a) Mississippi network (Y1 = Bighorn river, Y2 = Powder river, P1 = Sweet water river, P2 = Big Thompson river).

(b) The sequence of the CONNECT clause

Fig 6.5



6.3.2 SQL3: With Recursive Statement

- Syntax
 - WITH RECURSIVE <Relational Schema>
 - AS <Query to populate relational schema> Syntax details
 - <Relational Schema> lists columns in result table with directed edges
 - <Query to populate relational schema> has UNION of nested sub-queries
 - Base cases to initialize result table
 - Recursive cases to expand result table
- Semantics
 - Results relational schema say X(source, dest)
 - Columns source and dest come from same domain, e.g. Vertices
 - X is a edge table, X(a,b) directed from a to b
 - Result table X is initialized using base case queries
 - Result expanded using X(a, b) and X(b, c) implies X(a, c)



6.3.3 SQL3 Recursion - Example

- Revisit Figure 6.4 (pp. 158) on transitive closures
- Computing table X in Fig. 6.4(d), from table R in Fig. 6.4(b)
WITH RECURSIVE X(source,dest)
AS (**SELECT** source,dest **FROM** R)
UNION
(**SELECT** R.source,X.dest **FROM** R,X **WHERE** R.dest=X.source);
- Meaning
 - Initialize X by copying directed edges in relation R
 - Infer new edge(a,c) if edges (,b) and (b,c) are in X
 - Declarative query does not specify algorithm needed to implement it
- Exercise:
 - Write a SQL expression using WITH RECURSIVE to determine
 - all direct and indirect tributaries of the Mississippi river
 - all ancestors of the Missouri river



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, Dijkstra's and Hierarchical routing