GLOBAL JOURNAL OF ENGINEERING, DESIGN & TECHNOLOGY
(Published By: Global Institute for Research & Education)
www.gifre.org

Proficient Maintenance of Path Queries over Updated Collections of Routes

M.Raghavendhar[#1], Dr D.Vasumathi[*2], & Dr. A.V.Krishna Prasad[#3]
[#1] M.Tech Student, JNTUH, Hyderabad, Andhra Pradesh, India
[*2]Professor, Department of CSE, JNTUH, Hyderabad, Andhra Pradesh, India
[#3]Associate Professor, Department of IT, MVSR engineering College, Hyderabad, Andhra Pradesh, India

Abstract
In the existing system collection of routes is difficult. It consumes more computation cost. It could not store updated route collections. The evaluation was done based on sequence of nodes only. Source and destination nodes are chosen. There are many routes from source and destination resources. One of the routes is chosen for exchange of resources. After some time it requires update. Select of new update route require more amount of disk space. It takes more amount of searching time here. There is also problem with addition and deletion of the links. It is very difficult under route maintenance. All the problems are solved by the proposed System.

Keywords-- *GPS Technology, Searching Algorithms, Indexing Techniques.*

Introduction
The GPS system provides the services under route collections cannot give the exact interested location results. It can give the results as a large route. Any user can require any path from source location to destination location automatically find out sequence of interest points. Using sequence of interest points generate the path. Any user can require update route it takes more amount of time for searching here. Sometimes it can give the wrong result generation also here. It can takes more amount of time and spend computation cost is high here. Sometimes query is not terminated for providing the result. Storing all the routes require more amount of space here. The disadvantages are Maintenance cost is high; Search time takes more, less scalability, frequent identification of routes is not possible here. This paper addresses the problems listed above by introducing new generic searching algorithms. Different kinds of techniques are implemented here like traversal search and transitivity property. Every time it finds the route based on successor nodes utilization. After identification successor nodes create the links from one node to another node. It's possible for reaching the target node very faster way in implementation part specification here. It is based on two algorithms those are called indexing techniques. Those are T-Index, R-Index. These two techniques perform the good tuning operations. It can give stronger result in output generation. Any update time also gives the result with less amount time utilization here. Every operations work based on GPS technology.

The Advantages of the proposed system are less amount of maintenance cost with GPS technology, it takes less search time, it has more scalability, and frequent identification of routes is possible here.

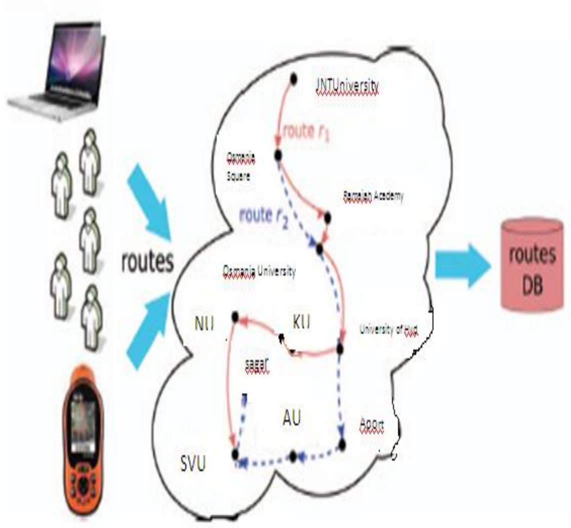


Figure 1 Architecture of the proposed System

Consider the route collection of Figure 1 and a path query: "Find a path from Nagarjuna University to JNT University following existing routes". Note that a path may contain nodes from different routes, since reaching KU from NU may require changing routes using links, i.e., nodes shared among routes.

This work targets path query evaluation on large disk resident route collections that are frequently updated. Updates involve additions and deletions of routes. A route collection can be trivially transformed to a graph; hence, path queries can be evaluated using standard graph search techniques. Such methods follow one of two paradigms. The first employs graph traversal methods, such as depth-first search. The second compresses the graph's transitive closure, which contains reachability information, i.e., whether a path exists between any pair of nodes.

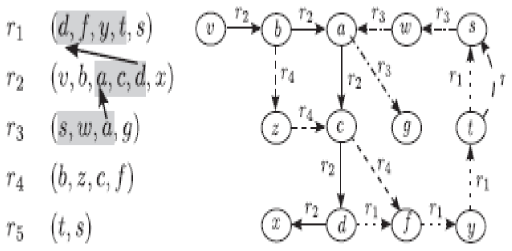


Figure 2: Transition diagram

Technical Work

1) Route Collection with sequence of nodes:

Select n number of nodes each as a distinct node here. Nodes are participating in routing and generate the routes collection. It can contain the communication from source node to destination node. Generate the route with sequence of nodes collection process. This is different path collection. This is the answer result of routes.

2) Route Traversal Search:

Its starts the search process based on depth first search. It starts the search process from current node. It identifies the successor nodes. This type of searching process is applied till termination of all routes. This is the shortest result identification result. All shortest results are displayed inverted R- index. The R-Index for the Route Collection R shown in the table 1.

Table – 1. R-Index for the Route Collection R

node	Routes[] list
a	<r2:3>,<r3,3>
b	<r2:2>,<r4:1>
c	<r2:4>,<r4:3>
d	<r1:1>,<r2:5>
f	<r1:2>,<r4:4>
g	<r3:4>
s	<r1:5>,<r3:1>,<r5:2>
t	<r1:4>,<r5:1>
v	<r2:1>
w	<r3:2>
x	<r2:6>
y	<r1:3>
z	<r4:2>

3) Route Traversal Search with Transition

This module presents the Route Traversal Search with Transitions algorithm for path queries. RTST expands the search as RTS, but employs a stronger termination check based on the transitions between routes. This additional reachability information is modeled by the transition graph GT, and is explicitly materialized in the T -Index structure

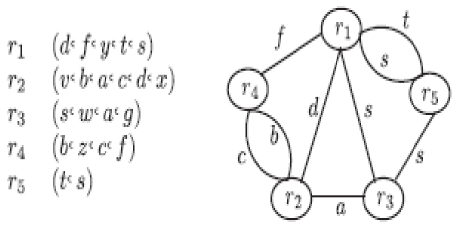


Figure 3 : Transition Graph for the Route collection R

Table 2: T-Index of Transition Graph

route	Trans[] list of Routes in Transition Graph
r1	<r2,d:1:5>,<r3,s:5:1>,<r4,f:2:4>,<r5,t:4:1>,<r5,s:5:2>
r2	<r1,d:5:1>,<r3,a:3:3>,<r4,b:2:1>,<r4,c:4:3>
r3	<r1,s:1:5>,<r2,a:3:3>,<r5,s:1:2>
r4	<r1,f:4:2>,<r2,b:1:2>,<r2,c:3:4>
r5	<r1,t:1:4>,<r1,s:2:5>,<r3,s:2:1>

4) Link Traversal Search:

Using depth first search remove the redundant links of nodes. Start the links of communications from first current node, automatically identifies the nearest nodes. It can establish the shortest route. Using R-index$^+$ list retrieves the T-index information. All subsequences of routes are identified as a final list.

Table 3: R-Index$^+$ for the Route Collection R

Node	Routes$^+$[] list
a	<r2:3,c>,<r3:3,Ø>
b	<r2:2,a>,<r4:1,c>
c	<r2:4,d>,<r4:3,f>
d	<r1:1,f>,<r2:5, Ø>
f	<r1:2,t>,<r4:4, Ø>
g	<r3:4, Ø>
s	<r1:5, Ø>,<r3:1,a>,<r5:2, Ø>
t	<r1:4,s>,<r5:1,s>
v	<r2:1,b>
w	<r3:2,a>
x	<r2:6, Ø>
y	<r1:3,t>
z	<r4:2,c>

5) Link Traversal Search with Transition

The Link Traversal Search with transitions algorithm enforces a stronger termination check than LTS using the Transition graph of route Collection It uses R-Index$^+$ and T-index.

The proposed system implementation part is presented with Figures 4, 5, 6, 7 and 8 by showing the Class diagram, Sequence diagram, Database diagram, Screen shot 1, screen shot2 of the application.

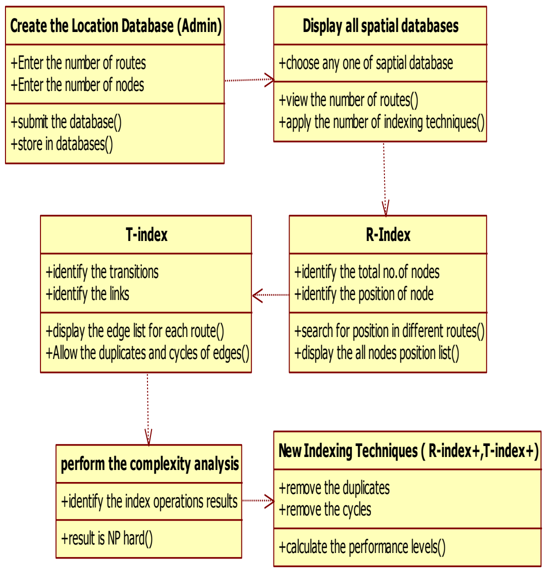

Create the Location Database (Admin)
+Enter the number of routes
+Enter the number of nodes
+submit the database()
+store in databases()
Display all spatial databases
+choose any one of saptial database
+view the number of routes()
+apply the number of indexing techniques()
T-index
+identify the transitions
+identify the links
+display the edge list for each route()
+Allow the duplicates and cycles of edges()
R-Index
+identify the total no.of nodes
+identify the position of node
+search for position in different routes()
+display the all nodes position list()
perform the complexity analysis
+identify the index operations results
+result is NP hard()
New Indexing Techniques (R-index+,T-index+)
+remove the duplicates
+remove the cycles
+calculate the performance levels()

Figure 4: Class Diagram

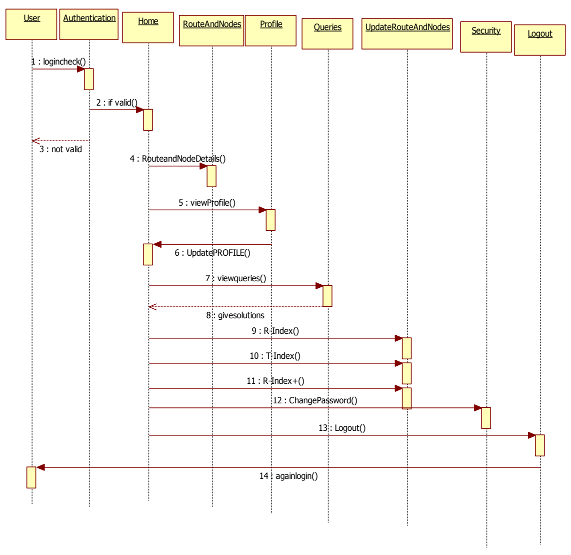


Figure 5: Sequence Diagram for Application

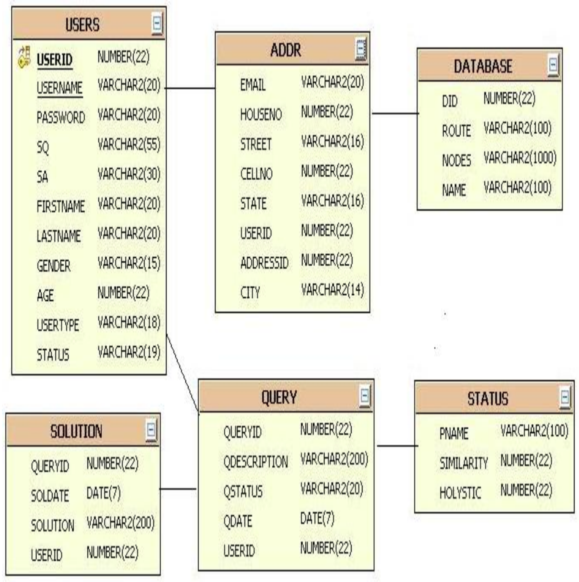
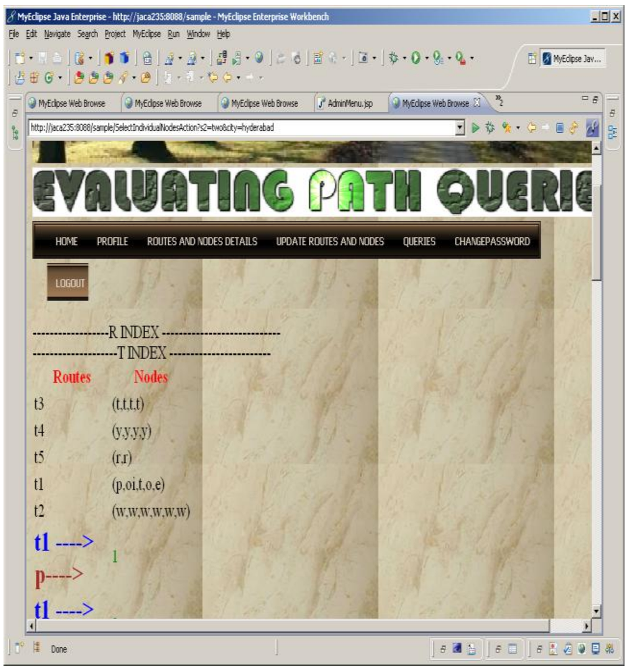


Figure 6 Database Diagram

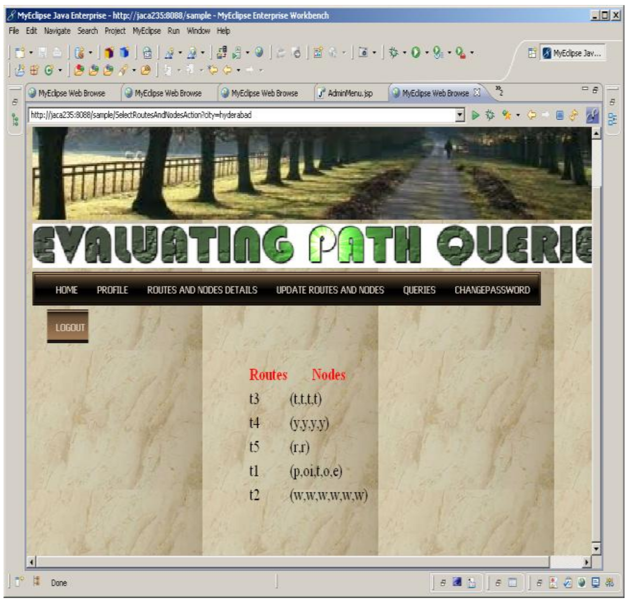

MyEclipse Java Enterprise - http://jaca235:8088/sample - MyEclipse Enterprise Workbench
File Edit Navigate Search Project MyEclipse Run Window Help
MyEclipse Web Browse
AdminMenu.jsp
http://jaca235:8088/sample/SelectIndividualNodesAction?s2=two&city=hyderabad
EVALUATING PATH QUERIE
HOME PROFILE ROUTES AND NODES DETAILS UPDATE ROUTES AND NODES QUERIES CHANGEPASSWORD
LOGOUT
Done

-----------------R INDEX---------------------------
-----------------T INDEX-----------------------

Routes	Nodes
t3	(t,t,t,t)
t4	(y,y,y,y)
t5	(r,r)
t1	(p,o,i,t,o,e)
t2	(w,w,w,w,w,w)

t1 ---->
1
p---->
t1 ---->

Figure 7 Screen shot 1


MyEclipse Java Enterprise - http://jaca235:8088/sample - MyEclipse Enterprise Workbench
File Edit Navigate Search Project MyEclipse Run Window Help
MyEclipse Web Browse
AdminMenu.jsp
http://jaca235:8088/sample/SelectRoutesAndNodesAction?city=hyderabad
EVALUATING PATH QUERIE
HOME PROFILE ROUTES AND NODES DETAILS UPDATE ROUTES AND NODES QUERIES CHANGEPASSWORD
LOGOUT
Done

Routes	Nodes
t3	(t,t,t,t)
t4	(y,y,y,y)
t5	(r,r)
t1	(p,o,i,t,o,e)
t2	(w,w,w,w,w,w)

Figure 8 Screen shot 2

Conclusion

With this system routes are computed and maintained with less amount of time and cost. By using the algorithms R-Index, R-index+, T-Index and Transition Graph, proper routes are computed with high scalability. It uses updated route collections and indexing techniques. It uses GPS (Global Positioning Systems) which is very popular.

References

[1]. P. Bouros, S. Skiadopoulos, T. Dalamagas, D. Sacharidis, and T. K.Sellis, "Evaluating reachability queries over path collections," inSSDBM, 2009, pp. 398–416.
[2]. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, Second Edition. The MIT Press and McGraw-Hill Book Company, 2001.
[3]. E. Cohen, E. Halperin, H. Kaplan, and U. Zwick, "Reachabilityand distance queries via 2-hop labels," in SODA, 2002, pp. 937946.
[4]. R. Schenkel, A. Theobald, and G. Weikum, "Hopi: An efficientconnection index for complex xml document collections," in EDBT, 2004, pp. 237–255.
[5]. "Efficient creation and incremental maintenance of the hopi index for complex xml document collections," in ICDE, 2005, pp.360–371.
[6]. J. Cheng, J. X. Yu, X. Lin, H. Wang, and P. S. Yu, "Fast computation of reachability labeling for large graphs," in EDBT, 2006, pp. 961979.
[7]. "Fast computing reachability labelings for large graphs with high compression rate," in EDBT, 2008, pp. 193–204
[8]. R. Bramandia, B. Choi, and W. K. Ng, "On incremental maintenance of 2-hop labeling of graphs," in WWW, 2008, pp. 845–854.
[9]. R. Jin, Y. Xiang, N. Ruan, and D. Fuhry, "3-hop: a high compression indexing scheme for reachability query," in SIGMOD Conference, 2009, pp. 813–826.
[10]. R. Agrawal, A. Borgida, and H. V. Jagadish, "Efficient management of transitive relationships in large data and knowledge bases," in SIGMOD Conference, 1989, pp. 253–262.
[11]. H. Wang, H. He, J. Yang, P. S. Yu, and J. X. Yu, "Dual labeling: Answering graph reachability queries in constant time," in ICDE, 2006, p. 75.