International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue-17
December 2017

Road Network Navigation by Using Path Planning with Cache and GPS

Viriventi Amani Sneha , G. Sravan Kumar, Dr.Suresh Akella

[1]M.Tech Computer Science Engineering Sreyas Institute of Engineering and Technology, Beside INDU Aranya,Nagole, Hyderabad M.Tech.,(Ph.D.)

[2]Associate Professor, Department of Computer Science And Engineering Sreyas Institute of Engineering and Technology, Beside INDU Aranya,Nagole, Hyderabad

[3]Principal Sreyas Institute of Engineering and Technology, Beside INDU Aranya,Nagole, Hyderabad

Abstract: *Path planning is a basic operation of route navigation services. It finds out route between our required starting place and ending place. Now there are number of applications is present like GPS and digital mapping. But due to unexpected variation in driving direction, losing of GPS signal like many issues we need to propose this path planning method. We propose a system, cash based path planning method. It respond to us with respect our query, also it return result that previously queried that stored in our database. Our system technique decreases almost fifty percentage of computation latency on average.*

Key Words: Spatial database, path planning, cache.

I. INTRODUCTION

In advance of GPS and other path finding methods we propose another system technique is path planning by cache supported. Due to unexpected variation in driving direction, losing of GPS signal like many issues we need to propose this path planning method. It find rout network between specified starting and ending places. Also we can store our queries in database and can use in later. In our cache supported technique, cached query will result when it completely matches to latest query. So system only needs to compute in case of unmatched query. Therefore here we can reduce the overall system latency. The requirement of timeliness is even more challenging when an overwhelming number of path planning queries is submitted to the server, e.g., during peak hours. As the response time is critical to user satisfaction with personal navigation services, it is a mandate for the server to efficiently handle the heavy workload of path planning requests. To meet this need, we propose a system, namely, Path Planning by Caching (PPC), that aims to answer a new path planning query efficiently by caching and reusing historically queried paths (queried-paths in short). Unlike conventional cache-based path planning systems where a cached query is returned only when it matches completely with a new query, PPC leverages partially matched

queried-paths in cache to answer part(s) of the new query. As a result, the server only needs to compute the unmatched path segments, thus significantly reducing the overall system workload.

Benefaction of our work includes:

Path planning by caching (PPC) that return answer for path planning queries, here we save approximately 32 % of time by comparing with other path finding systems. Ppattern, path that cached will share segments with other path In the basis of user priority of roads on different types, we also introduced cache replacement operation

II. RELATED WORKS

"Shared execution of road queries on highway networks," Authors: H. Mahmud, A. M. Amin, M. E. Ali, and T. Hashem

The development of mobile technologies and also the proliferation of map-based applications have enabled a user to use a wide variety of solutions which range from info queries to navigation systems. Because of the acceptance of map-based uses among the users, the service provider often requires responding to a large a selection of simultaneous queries. Thus, processing queries effectively on spatial networks (i.e., road networks) have grown to be an essential research subject in recent years. In this particular paper, we focus on path queries that will find probably the shortest course between a destination and a source of the user. Particularly, we tackle the issue of locating the shortest paths for a big selection of simultaneous path queries in highway networks. Traditional systems that consider one query at a time are not appropriate for many uses as a result of high computational and service expenses. These systems can't promise needed response time in increased load circumstances. We propose an efficient group-based strategy which offers a practical option with the reduced price. The important idea for our strategy is to group queries that share a typical travel path then compute probably the shortest path for the team. Experimental results indicate that the approach of ours is on an average 10 times faster than the standard approach in exchange of sacrificing the precision of 0.5 % in the most detrimental case, which is actually appropriate for the majority of the users.

"An effective path computation model for hierarchically structured topographical road maps,"

In this paper, we have developed a HiTi (Hierarchical MulTi) a graph type for structuring sizeable topographical roadmaps to speed up the least price route computation. The HiTi graph model provides a novel approach to abstracting and structuring a topographical highway map in a hierarchical manner. We propose a new shortest path algorithm called SPAH, which utilizes HiTi graph type of a topographical roadmap for its computation. We give the evidence for the optimality of SPAH. Our performance evaluation of SPAH on the grid graphs proved that it substantially lowers the search area over existing methods. We also, provide an in-depth experimental analysis of HiTi graph approach by comparing it with various others the same works on grid graphs. Within the HiTi graph framework, we also suggest a parallel shortest path algorithm called ISPAH. Experimental results show that inter-query shortest path problem provides much more opportunity for scalable parallelism as opposed to the intraquery shortest path issue.

"Computing probably the shortest path: A research fulfills graph theory," Authors: A. V. C and Goldberg. Harrelson

We proposes shortest path algorithms that use A search in conjunction with a new graph-theoretic lower bounding method based on landmarks as well as the triangle inequality. Optimal shortest paths and work on any directed graph are computed by our algorithms. We give experimental results indicating that the most effective of our latest algorithms outperforms last algorithms, in specific A* search with Euclidean bounds, by a large margin on road networks and on a few artificial problem families.

"Reach-based routing: A brand new approach to shortest path algorithms enhanced for street networks," Authors: R. Gutman

The issue of determining probably the shortest path along a road network is a fundamental the condition in network analysis, ranging from route assistance in a navigation structure to solving spatial allocation issues. Since this particular problem type is resolved so frequently, it's essential to craft an approach which is as effective as they can. Based upon past the investigation, it's generally recognized that several effective implementations of the Dijkstra the algorithm is probably the fastest at optimally solving the' one-to-one' least path issue We show that the most effective state-of-the-art implementations of Dijkstra could be raised if you take advantage of network attributes related to GIS sourced data. The outcomes of this particular paper, derived from tests of many algorithmic approaches on real road networks are going to be incredibly valuable for software developers and researchers in the GIS community.

III. PROPOSED METHOD

Fig.1 represents the overall functioning of our system. It mainly consists of three components. That is (i) PPattern Detection (path pattern), (ii) Shortest Path Estimation, and (iii) Cache Management. Here firstly find out and return number of historical path in cache that is called PPattern. Fig.2 shows an example of illustration of PPattern (PPatternis another path which shares at least two consecutive nodes). In figure the paths ps,t and ps',t' are2-PPattern to each other as they share a common sub-path pa,b . Node ais the pattern head and node b is the pattern tail. In shortest path estimation part, that build candidate path for new query and select shortest one. Here if PPattern is same as query, system immediately returns to user. In other case system compute unmatched path segment between PPattern and query. And if cache is full cache management part determine which queried-paths in cache should be deleted.

International Journal of Research
Available at https://edupediapublications.org/journals

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue-17
December 2017

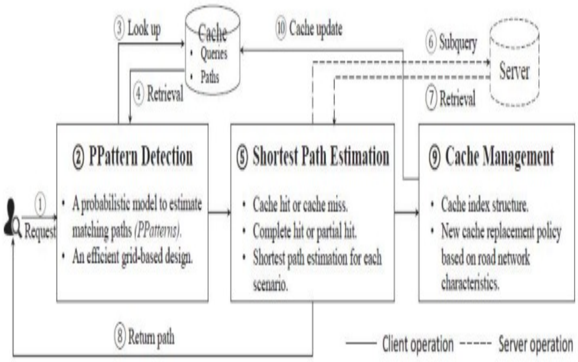

Fig. 1: overview of the ppc system

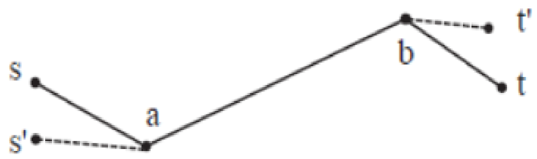

Fig.2: Example of illustration of PPattern

When implementing our paper, it includes mainly four modules. That is system construction Module, Probabilistic model for PPattern detection, an efficient grid-based solution and cache construction and update.

A. System Construction Module

In system construction module, we develop the system with the required entities to implement our proposed model and evaluate the effectiveness of the system. The main goal in this work is to reduce the server workload by leveraging the queried-paths in cache to answer a new path planning query. An intuitive solution is to check whether there exists a cached queried-path perfectly matching the new query. Here, a perfect match means that the source and destination nodes of the new query are the same as that of a queried-path in cache.

B. Probabilistic model for PPattern detection

To detect the best PPatterns, an idea is to calculate the estimation distance based on each cached path, and select the cached path with the shortest distance. It faces many challenges. Firstly, the distance estimation requires the server to compute the unshared segments. Therefore, it incurs significant computation to exhaustively examine all cached paths. Secondly, such an exhaustive operation implicitly assumes that each cached path is a PPattern candidate to the query.

Algorithm of PPattern detection:

STEP 1: If distance between nodes s(source) and t (destination) less than threshold (Dl) value, return the result

PPattern = NULL

STEP 2: Divide the target space by grid cell size

STEP 3: Find out source(gs) and destination(gt) grid

STEP 4: Assign Qs = Logged queries whose paths pass start grid(gs)

AndQt = Logged queries whose paths pass destination grid(gt)

STEP 5: Store the intersect value of Qs and Qt into Q

STEP 6: Assign path from source grid to destination grid for each query into PT

STEP 7: Return cached path (PT).

C. Efficient Grid Based Solution

To retrieve these patterns, we invent a grid-based solution to further improve the system efficiency. Here divide the whole space into equally sized grid cells, here endpoints of all paths are mapped to the grid cells. By counting the total number of covered grids we will get the distance measure.

D. Cache Construction and Update

It is an important part of cache management. Here we invent a cache replacement method by taking unique characteristics of road pattern. By observing many users, we found that certain routes are selected by most of the users. Most of them select main roads than branch roads. Because of the efficiency, popularity and capacity of these major roads. In a road network G = (V, E), each edge from node vmto vnis associated with a weight Wm,n it is a system computed value corresponding to the road type. If Wm,n is high indicates that the corresponding road type is high.

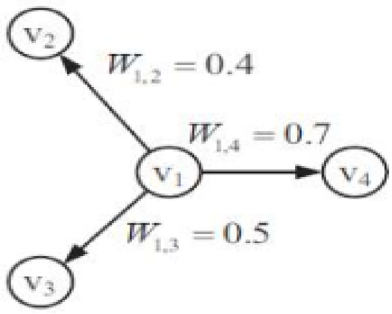

Fig.3: Example for edge and node weight

Fig. 3 shows an example for node and edge weight to road types. Here edges are connected with node v1. The weights of three edges are W1,2= 0.4, W1,3= 0.5, and

W1,4= 0.7, respectively. The weight of W1is set to W1,4= 0.7.Nodeweight represents how path planning query with it as the source node to be issued later. Information like this can be used to propose the cache replacement policy.

Algorithm of PPattern detection:

STEP 1: Assign PPatterns Detection into variable PT.

STEP 2: Assign Shortest Path Estimation from PT into variable p.

STEP 3: if cache C is not full then insert p into cache and return C

STEP 4: Otherwise calculate usability for cached path and stored into {μ} and path with minimum usability and stores it into p*

STEP 5: Check whether the usability value of the current path p is larger than the minimum usability value in the current cache. If so, we place the current query into C. and return C.

IV. CONCLUSION

With this paper, a cached path planning method was implemented by us. In addition, all downside of the current system is solved by it. That's, our implemented system decreases the time period complexity, the Cached query is actually returned when it partly matches with a brand new query too, Cache written content is actually updated as well as the cost of building cache is actually low. Here server just has to calculate unmatched path segments. And so here workload of the method is really law.

REFERENCES

[1] H. Mahmud, A. M. Amin, M. E. Ali, and T. Hashem, "Shared execution of path queries on road networks," Clinical Orthopaedics Related Res., vol. abs/1210.6746, 2012.

[2] L. Zammit, M. Attard, and K. Scerri, "Bayesian hierarchical modelling of traffic flow - With application to Malta's road network," in Proc. Int. IEEE Conf. Intell. Transp. Syst., 2013, pp. 1376–1381

[3] S. Jung and S. Pramanik, "An efficient path computation model for hierarchically structured topographical road maps," IEEE Trans. Knowl. Data Eng., vol. 14, no. 5, pp. 1029–1046, Sep. 2002.

[4] E. W. Dijkstra, "A note on two problems in connexion with graphs," Num. Math., vol. 1, no. 1, pp. 269–271, 1959.

[5] U. Zwick, "Exact and approximate distances in graphs – a survey," in Proc. 9th Annu. Eur. Symp. Algorithms, 2001, vol. 2161, pp. 33–48.

[6] A. V. Goldberg and C. Silverstein, "Implementations of Dijkstra's algorithm based on multi-level buckets," Network Optimization, vol. 450, pp. 292–327, 1997.

[7] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," IEEE Trans. Syst. Sci. Cybern., vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[8] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A search meets graph theory," in Proc. ACM Symp. Discr. Algorithms, 2005, pp. 156–165.

[9] R. Gutman, "Reach-based routing: A new approach to shortest path algorithms optimized for road networks," in Proc. Workshop Algorithm Eng. Experiments, 2004, pp. 100–111.

[10] A. V. Goldberg, H. Kaplan, and R. F. Werneck, "Reach for A*: Efficient point-to-point shortest path algorithms," in Proc. Workshop Algorithm Eng. Experiments, 2006, pp. 129–143.

[11] S. Jung and S. Pramanik, "An efficient path computation model for hierarchically structured topographical road maps," IEEE Trans. Knowl. Data Eng., vol. 14, no. 5, pp. 1029–1046, Sep. 2002.

[12] R. Goldman, N. Shivakumar, S. Venkatasubramanian, and H. Garcia-Molina, "Proximity search in aatabases," in Proc. Int. Conf.
Very Large Data Bases, 1998, pp. 26–37.

[13] N. Jing, Y.-W. Huang, and E. A. Rundensteiner, "Hierarchical optimization of optimal path finding for transportation applications," in Proc. ACM Conf. Inf. Knowl. Manage., 1996, pp. 261–268.

[14] N. Jing, Y. wu Huang, and E. A. Rundensteiner, "Hierarchical encoded path views for path query processing: An optimal model and its performance evaluation," IEEE Trans. Knowl. Data Eng., vol. 10, no. 3, pp. 409–432, May/Jun. 1998.

[15] U. Demiryurek, F. Banaei-Kashani, C. Shahabi, and A. Ranganathan, "Online computation of fastest path in time-dependent spatial networks," in Proc. 12th Int. Conf. Adv. Spatial Temporal Databases, 2011, pp. 92–111