EFFICIENT CACHE SUPPORTED PATH PLANNING ROADS

BATHINA LAKSHMI SOWJANYA, V. NAGA SRINIVAS
Mtech Scholar, Assistant Professor
Department of of CSE
ISTS Women's Engineering College, Rajanagaram, Rajamahendravaram, A.P, India.

ABSTRACT:

Owing to the wide availability of the global positioning system (GPS) and digital mapping of roads, road network navigation services have become a basic application on many mobile devices. Path planning, a fundamental function of road network navigation services, finds a route between the specified start location and destination. The efficiency of this path planning function is critical for mobile users on roads due to various dynamic scenarios, such as a sudden change in driving direction, unexpected traffic conditions, lost or unstable GPS signals, and so on. In these scenarios, the path planning service needs to be delivered in a timely fashion. In this paper, we propose a system, namely, Path Planning by Caching (PPC), to answer a new path planning query in real time by efficiently caching and reusing historical queried-paths. Unlike the conventional cache-based path planning systems, where a queried-path in cache is used only when it matches perfectly with the new query, PPC leverages the partially matched queries 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. Comprehensive experimentation on a real road network database shows that our system outperforms the state-of-the-art path planning techniques by reducing 32 percent of the computation latency on average.

INTRODUCTION

With the advance of the global positioning system (GPS) and the popularity of mobile devices, we have witnessed a migration of the conventional Internet-based on-line navigation services (*e.g.*, Mapquest) onto mobile platforms (*e.g.*, Google Map). In mobile navigation services, on-road path planning is a basic function that finds a route between a queried start location and a destination. While on roads, a path planning query may be issued due to dynamic factors in various scenarios, such as a sudden change in driving direction, unexpected traffic conditions, or lost of GPS signals. In these scenarios, path planning needs to be delivered in a *timely* fashion.

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. Figure 1 provides an overview of the proposed PPC system framework, which consists of three main components (in rectangular boxes, respectively):

(i) PPattern Detection, (ii) Shortest Path Estimation, and (iii) Cache Management. Given a path planning query (see Step (1)), which contains a source location and a destination location, PPC firstly determines and retrieves a number of historical paths in cache, called

PPatterns, that may match this new query with high probability (see Steps (2)-(4)). 1 The idea of PPatterns is based on an observation that similar starting and destination nodes of two queries may result in similar shortest paths (known as the *path coherence* property [1]). In the component PPatern Detection, we propose a novel probabilistic model to estimate the likelihood for a cached queried-path to be useful for answering the new query by exploring their geospatial characteristics. To facilitate quick detection of PPatterns, instead of exhaustively scanning all the queried-paths in cache, we design a grid-based index for the Pattern Detection module. Based on these detected PPatterns, the *Shortest Path Estimation* module (see Steps (5)-(8)) constructs candidate paths for the new query and chooses the best (shortest) one. In this component, if a PPattern perfectly matches the query, we immediately return it to the user; otherwise, the server is asked to compute the unmatched path segments between the PPattern and the query (see Steps (6)-(7)). Because the unmatched segments are usually only a smaller part of the original query, the server only processes a **"**smaller subquery**"**, with a reduced workload. Once we return the estimated path to the user, the *Cache Management* module is triggered to determine which queried-paths in cache should be evicted if the cache is full. An important part of this module is a new cache replacement policy which takes into account the unique characteristics of road networks. Through an empirical study, we find that common road segments in various queried-paths usually have road types of higher importance and capacity [2], [3].2 This inspires us to define a usability value for each path by considering both of its road type and historical frequency of use. The main contributions made in this work are summarized as follows:

• We propose an innovative system, namely, *path planning by caching*(PPC), to efficiently answer a new path planning query by using cached paths to avoid undergoing a time-consuming shortest path computation. On average, we save up to 32% of time in comparison with a conventional path planning system (without using cache).

• We introduce the notion of *PPattern*, *i.e.*, a cached path which shares segments with other paths. PPC supports partial hits between PPatterns and a new query. Our experiments indicate that partial hits constitute up to 92.14% of all cache hits on average.

• A novel probabilistic model is proposed to detect the cached paths that are of high probability to be a PPattern for the new query based on the coherency property of the road networks. Our experiments indicate that these PPatterns save retrieval of path nodes by 31.69% on average, representing a ten-fold improvement over the 3.04% saving achieved by a complete hit.

• We have developed a new cache replacement mechanism by considering the user preference among roads of various types. A usability measure is assigned for each query by addressing both the road type and query popularity. The experimental results show that our new cache replacement policy increases the overall cache hit ratio by 25.02% over the state-of-the-art cache replacement policies.

SYSTEM STUDY

2.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

2.2 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.3 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

LITERATURE SURVEY

1) Interoperability of personal health records

AUTHORS: J. L ahteenm€ aki, J. Leppanen, and H. Kaijanranta,

The establishment of the Meaningful Use criteria has created a critical need for robust interoperability of health records. A universal definition of a personal health record (PHR) has not been agreed upon. Standardized code sets have been built for specific entities, but integration between them has not been supported. The purpose of this research study was to explore the hindrance and promotion of interoperability standards in relationship to PHRs to describe interoperability progress in this area. The study was conducted following the basic principles of a systematic review, with 61 articles used in the study. Lagging interoperability has stemmed from slow adoption by patients, creation of disparate systems due to rapid development to meet requirements for the Meaningful Use stages, and rapid early development of PHRs prior to the mandate for integration among multiple systems. Findings of this study suggest that deadlines for implementation to capture Meaningful Use incentive payments are supporting the creation of PHR data silos, thereby hindering the goal of high-level interoperability.

2) Applying cloud computing model in PHR architecture.

AUTHORS: S. Kikuchi, S. Sachdeva, and S. Bhalla,

In recent years, some practical and commercial Personal Health Records and some related services such as Google Health [1] and Microsoft HealthVault [2] have been launched. On the other hand, Cloud Computing has matured more and become the major streams to realize a more effective operational environment. However so far, there have been few studies in regards to applying Cloud architecture in the PHR explicitly despite generating volume data. In this paper, we review our trial on the general architecture design by applying the Cloud components for supporting healthcare record areas and clarify the required conditions to realize it.

3) Health Information Privacy, Security, and Your EHR.

AUTHORS: M. Bellare

If your patients lack trust in Electronic Health Records (EHRs) and Health Information Exchanges (HIEs), feeling that the confidentiality and accuracy of their electronic health information is at risk, they may not want to disclose health information to you. Withholding their health information could have life-threatening consequences. To reap the promise of digital health information to achieve better health outcomes, smarter spending, and healthier people, providers and individuals alike must trust that an individual's health information is private and secure.

Your practice, not your EHR developer, is responsible for taking the steps needed to protect the confidentiality, integrity, and availability of health information in your EHR system.

4) A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud

AUTHORS: C. Ng and P. Lee. Revdedup

Benefited from cloud computing, users can achieve an effective and economical approach for data sharing among group members in the cloud with the characters of low maintenance and little management cost. Meanwhile, we must provide security guarantees for the sharing data files since they are outsourced. Unfortunately, because of the frequent change of the membership, sharing data while providing privacy-preserving is still a challenging issue, especially for an untrusted cloud due to the collusion attack. Moreover, for existing schemes, the security of key distribution is based on the secure communication channel, however, to have such channel is a strong assumption and is difficult for practice. In this paper, we propose a secure data sharing scheme for dynamic members. Firstly, we propose a secure way for key distribution without any secure communication channels, and the users can securely obtain their private keys from group manager. Secondly, our scheme can achieve fine-grained access control, any user in the group can use the source in the cloud and revoked users cannot access the cloud again after they are revoked. Thirdly, we can protect the scheme from collusion attack, which means that revoked users cannot get the original data file even if they conspire with the untrusted cloud. In our approach, by leveraging polynomial function, we can achieve a secure user revocation scheme. Finally, our scheme can achieve fine efficiency, which means previous users need not to update their private keys for the situation either a new user joins in the group or a user is revoked from the group

5) **ADVANCE SECURITY TO CLOUD DATA STORAGE AUTHORS:** P. Lee, and W. Lou
The proposed system is an effective and flexible distributed Scheme with explicit dynamic data support to ensure the correctness of user's data in the cloud. To fully ensure the data integrity and save the cloud users computation it is of critical importance to enable public auditing service for cloud data storage, so that users may depend on independent third party auditor to audit the outsourced data. The Third party auditor can periodically check the integrity of all the data stored in the cloud .which provides easier way for the users to ensure their storage correctness in the cloud.

SYSTEM DESIGN AND DEVELOPMENT

3.1 INPUT DESIGN

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.
This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design. Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

3.2 OUTPUT DESIGN

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.
The application starts running when it is executed for the first time. The server has to be started and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

SYSTEM ANALYSIS

5.1 EXISTING SYSTEM:

- Path planning needs to be delivered in a timely fashion. 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.
- Jung and Pramanik propose the HiTi graph model to structure a large road network model. HiTi aims to reduce the search space for the shortest path computation. While HiTi achieves high performance on road weight updates and reduces storage overheads, it incurs higher computation costs when computing the shortest paths than the HEPV and the Hub Indexing methods.

  To compute time-dependent fast paths, Demiryurek et al. propose the B-TDFP algorithm by leveraging backward searches to reduce the search space. It adopts an area-level partition scheme which utilizes a road hierarchy to balance each area.

5.2 DISADVANTAGES OF EXISTING SYSTEM:

A cached query is returned only when it matches completely with a new query.

The time complexity is high.

The cache content may not be up to date to respond to recent trends in issued queries.

The cost of constructing a cache is high, since the system must calculate the benefit values for all sub-paths in a full-path of query results.

5.3 PROPOSED SYSTEM:

To meet existing 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).

The proposed system consists of three main components: (i) PPattern Detection, (ii) Shortest Path Estimation, and (iii) Cache Management.

Given a path planning query, which contains a source location and a destination location, PPC firstly determines and retrieves a number of historical paths in cache, called PPatterns, that may match this new query with high probability.

The idea of PPatterns is based on an observation that similar starting and destination nodes of two queries may result in similar shortest paths (known as the path coherence property).

In the component PPatern Detection, we propose a novel probabilistic model to estimate the likelihood for a cached queried-path to be useful for answering the new query by exploring their geospatial characteristics.

To facilitate quick detection of PPatterns, instead of exhaustively scanning all the queried paths in cache, we design a grid-based index for the PPattern Detection module. Based on these detected PPatterns, the Shortest Path Estimation module (see Steps (5)-(8)) constructs candidate paths for the new query and chooses the best (shortest) one.

In this component, if a PPattern perfectly matches the query, we immediately return it to the user; otherwise, the server is asked to compute the unmatched path segments between the PPattern and the query (see Steps (6)-(7)). Because the unmatched segments are usually only a smaller part of the original query, the server only processes a **"**smaller subquery**"**, with a reduced workload.

Once we return the estimated path to the user, the Cache Management module is triggered to determine which queried-paths in cache should be evicted if the cache is full. An important part of this module is a new cache replacement policy which takes into account the unique characteristics of road networks.

In this paper, we provide a new framework for reusing the previously cached query results as well as an effective algorithm for improving the query evaluation on the server.

5.4 ADVANTAGES OF PROPOSED SYSTEM:

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.

We propose an innovative system, namely, path planning by caching, to efficiently answer a new path planning query by using cached paths to avoid undergoing a time-consuming shortest path computation.

On average, we save up to 32 percent of time in comparison with a conventional path planning system (without using cache).

We introduce the notion of Pattern, i.e., a cached path which shares segments with other paths. PPC supports partial hits between Patterns and a new query. Our experiments indicate that partial hits constitute up to 92.14 percent of all cache hits on average.

A novel probabilistic model is proposed to detect the cached paths that are of high probability to be a PPattern for the new query based on the coherency property of the road networks. Our experiments indicate that these PPatterns save retrieval of path nodes by 31.69 percent on average, representing a 10-fold improvement over the 3.04 percent saving achieved by a complete hit.

We have developed a new cache replacement mechanism by considering the user preference among roads of various types. A usability measure is assigned for each query by addressing both the road type and query popularity. The experimental results show that our new cache replacement policy increases the overall cache hit ratio by 25.02 percent over the state-of-the-art cache replacement policies.

CONCLUSION

In this paper, we propose a system, namely, Path Planning by Caching (PPC), to answer a new path planning query with rapid response by efficiently caching and reusing the historical queried-paths. Unlike the conventional cache-based path planning systems, where a queried-path in cache is used only when it matches perfectly with the new query, PPC leverages the partially matched cached queries to answer part(s) of a new query. As a result, the server only needs to compute the unmatched segments, thus significantly reducing the overall system workload. Comprehensive experimentation on a real road network database shows that our system outperforms the state-of-the-art path planning techniques by reducing 32% of the computational latency on average.

REFERENCES

[1] H. Mahmud, A. M. Amin, M. E. Ali, and T. Hashem, “Shared Execution of Path Queries on Road Networks,” Clinical Orthopaedics and Related Research, 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 International IEEE Conference on Intelligent Transportation Systems, 2013, pp. 1376–1381.

[3] S. Jung and S. Pramanik, “An Efficient Path Computation Model for Hierarchically Structured Topographical Road Maps,” IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 5, pp. 1029–1046, 2002.

[4] E. W. Dijkstra, “A Note on Two Problems in Connexion with Graphs,” Numerische Mathematik, vol. 1, no. 1, pp. 269–271, 1959.

[5] U. Zwick, “Exact and approximate distances in graphs – a survey,” in Algorithms – ESA 2001, 2001, vol. 2161, pp. 33–48.

[6] A. V. Goldberg and C. Silverstein, “Implementations of Dijkstra’s Algorithm Based on Multi-Level Buckets,” in Network Optimization,

[7] P. Hart, N. Nilsson, and B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100–107, 1967.

[8] A. V. Goldberg and C. Harrelson, “Computing the Shortest Path: A Search Meets Graph Theory,” in ACM Symposium on Discrete Algorithms, 2005.

[9] R. Gutman, “Reach-Based Routing: A New Approach to Shortest Path Algorithms Optimized for Road Networks,” in Workshop on Algorithm Engineering and Experiments, 2004.

[10] A. V. Goldberg, H. Kaplan, and R. F. Werneck, “Reach for A*: Efficient Point-to-Point Shortest Path Algorithms,” in Workshop on Algorithm Engineering and Experiments, 2006, pp. 129–143.

[11] S. Jung and S. Pramanik, “An Efficient Path Computation Model for Hierarchically Structured Topographical Road Maps,” IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 5, pp. 1029–1046, 2002.

[12] R. Goldman, N. Shivakumar, S. Venkatasubramanian, and H. Garcia-Molina, “Proximity Search in Aatabases,” in International Conference on 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 ACM Conference on Information and Knowledge Management, 1996.

[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 Transactions on Knowledge and Data Engineering, vol. 10, pp. 409–432, 1998.

[15] U. Demiryurek, F. Banaei-Kashani, C. Shahabi, and A. Ranganathan, “Online Computation of Fastest Path in Time- Dependent Spatial Networks,” in International Conference on Advances in Spatial and Temporal Databases, 2011.

[16] H. Gonzalez, J. Han, X. Li, M. Myslinska, and J. P. Sondag, “Adaptive Fastest Path Computation on a Road Network: a Traffic Mining Approach,” in International Conference on Very Large Data Bases, 2007.

[17] J. R. Thomsen, M. L. Yiu, and C. S. Jensen, “Effective caching of shortest paths for location-based services,” in ACM International Conference on Management of Data, 2012.

[18] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, Third Edition, 2009.

[19] . Markatos, “On Caching Search Engine Query Results,” Computer Communications, vol. 24, no. 2, pp. 137–143, 2001.

[20] R. Ozcan, I. S. Altingovde, and O. Ulusoy, “A Cost-Aware Strategy for Query Result Caching in Web Search Engines,” in Advances in Information Retrieval, 2009, vol. 5478, pp. 628–636.

[21] R. Baeza-Yates, A. Gionis, F. Junqueira, V. Murdock, V. Plachouras, and F. Silvestri, “The Impact of Caching on Search Engines,” in International ACM Conference on Research and Development in Information Retrieval, 2007.

[22] R. Baeza-Yates and F. Saint-Jean, “A Three Level Search Engine Index Based in Query Log Distribution,” in String Processing and Information Retrieval, 2003, vol. 2857, pp. 56–65.

[23] R. Ozcan, I. S. Altingovde, B. B. Cambazoglu, F. P. Junqueira, and zgr Ulusoy, “A Five-Level Static Cache Architecture for Web no. 5, pp. 828 – 840, 2012.

[24] R. Ozcan, I. S. Altingovde, and O. Ulusoy, “Static Query Result Caching Revisited,” in International Conference on World Wide Web, 2008.