

Fall 2010

Jeppe Rishede Thomsen
Department of Computing
Hong Kong Polytechnic University

Abstract

All the papers either solve the problem of a more efficient cache in a specific domain, or use the network domain, which are both relevant, but not really useful when looking at shortest path caching.

The papers show some interesting ways to use cache, but ultimately their approaches are very domain or query specific so their approaches to caching and cache replacement/invalidation can not be applied directly.

1. Introduction

2. Problem

2.1. Definitions and problem setting

We assume a setting where owners of mobile, positioning enabled, devices want route planning assistance. We assume users prefer online route planning services over offline solutions. We expect users to use network enabled capable of determining and visualizing users location and route. Users want fast response times from online services, comparable to using an offline application [?]. Using a cache reduces the computational burden [?] on an online service, providing faster end-user response time [?] by both freeing up computational resources to calculate new routes, as well as being able to immediately provide the shortest path result from the cache. We assume a scenario using only server side caching.

2.2. methods

2.2.1. Baseline. The straight forward baseline solution to implement a server side shortest path cache would be to store query results in the cache and only consider direct hits, and using a simple cache policy such as LRU or FIFO. The advantage of this solution is clear: it is simple and easily implemented. This simplicity is however obviously also its main disadvantage, as it is

too simple and very inefficient in terms of the utility the cache provides. Using items in the cache only when there is an exact match makes it exceedingly unlikely to get a cache hit due to the nature of route planning (many people share parts of routes, but few the same start and end points) and the sheer number of start-/end-point combinations possible. The

2.2.2. Baseline, improved. One way to possibly increase the utility of a naive cache as proposed in 2.2.1 would be to exploit the optimal substructure [?] property of the cache items. There is a significant increase in cache hits to be expected by utilizing the optimal substructure of shortest path cache items since it is unlikely many people will plan a route from/to the same place, but it is very likely that some sub-paths will be shared among users, and some users' full path laying within a longer path already calculated. The disadvantage of doing this is the additional computational resources required to examine the substructure of cached shortest path search results. It is currently not known if doing this is worth the effort, compared to just calculating the route, possibly multiple times.

2.2.3. Number 3. Using the optimal substructure of each cache item it is possible to have much fewer items in cache and still retain a high cache hit percentage (assumption, need test result or proof), this percentage can further be improved by preferring longer paths in cache over shorter ones. A further improvement is to also prefer paths with sub-paths shared by many users. By adding a more intuitive cache replacement policy which takes in to consideration both the usage of each cache item, as well as the coverage of previously often seen queries it is likely that the utility of the cache would be much higher.

Number 3, ideas. use optimal substructure
prefer having longer paths in cache
prefer having paths with subpaths shared by many users
look at utility of cache items, not just usage when designing cache replacement policy.

- prefer often used cache items
- prefer items which cover routes/areas often used in previous queries

3. Related work reference

reference support for related work section.

3.0.4. On effective presentation of graph patterns: a structural representative approach. They develop an approach that combine two focuses when mining patterns in graphs. 1. they introduce a method to relax the tightness of the pattern subgraph pattern matching, so they can have high support for subgraphs which are very similar, but not exact. 2. as many mining approaches return allot (often very similar) patterns, they propose a method to collapse similar patterns so the user is presented with something that is easier to get an overview of and gain an understanding of the data. [1]

3.1. Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments

They develop two cache replacement and invalidation techniques for mobile clients communicating with a LBS. They argue that in the setting of spatial data and LBS then it is important to consider more than just the access time when doing cache replacement. They look at the spatial area where an object in the cache is valid as well as the direction the user is moving. They do this besides calculating the probability that this object will be accessed again.

Assumes all POI objects are fixed size and no updates will be made. [2]

3.2. Nearest-Neighbor Caching for Content-Match Applications

[3]

3.3. Caching Content-based Queries for Robust and Efficient Image Retrieval

They study how to do caching with Content-based Image Retrieval, and they support range and kNN queries. They focus on how to do caching when many of the queries are similar, but not the same (e.g. picture cropped or color changes) without polluting the cache. Their approach works in metric space and they develop

an approximate method to check if the result can be satisfied by the cache. They archive good results, getting few direct cache hits, but still satisfying many queries from similar queries in the cache.

[4]

3.4. Caching Complementary Space for Location-Based Services

They develop the notion of Complementary Space(CS) to help better use a cache on a mobile client. CS is different levels for representing the objects on a map within MBRs. At the lowest level they just show the object, and as the levels go up they include more and more objects within MBRs, looking at the trade of in communication up/down link from a mobile client. They always have the entire world represented within the clients cache, at different levels, and offer no solution to how they will handle server updates to the map.

This is very similar to [5], although the approach does not formally depend on an R-tree, they still use one and offer no viable alternative, which lessens the difference even more. Their results are better than their competitors, including [5], though it seems that they stop their graphs just before [5] beats them.

3.5. Proactive Caching for Spatial Queries in Mobile Environments

They develop an approach which uses the index of an R-tree to add context to a cache of spatial object on a mobile client. They develop several communication and space saving techniques by representing less important parts of the R-tree in more compact ways, or just not storing the lower nodes/leaves of the tree. They also formally prove the assymtotic bounds of their algorithms.

[5]

3.6. Cache-Oblivious Data Structures and Algorithms for Undirected Breadth-First Search and Shortest Paths

[6]

3.7. Cached Shortest-Path Tree: An Approach to Reduce the Influence of Intra-Domain Routing Instability

They assume a network setting and try to reduce the time and computational load it takes when network

topology changes, as well as prevent any links from being unreachable if the topology changes often. The propose a cache with shortest-path trees, arguing that even if the topology changes often, then it is mostly between the same configurations (e.g. a computer/router is turned off/on) meaning that a cache with the most common seen configurations will be able to drastically reduce the amount of computation needed to recalculate routing tables.

[7]

3.8. On Designing a Shortest-Path-Based Cache Replacement in a Transcoding Proxy

[8]

3.9. Optimizing Graph Algorithms for Improved Cache Performance

[9]

References

- [1] C. Chen, C. X. Lin, X. Yan, and J. Han, "On effective presentation of graph patterns: a structural representative approach," in *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*. New York, NY, USA: ACM, 2008, pp. 299–308.
- [2] B. Zheng, J. Xu, and D. L. Lee, "Cache invalidation and replacement strategies for location-dependent data in mobile environments," *IEEE Trans. Comput.*, vol. 51, no. 10, pp. 1141–1153, 2002.
- [3] S. Pandey, A. Broder, F. Chierichetti, V. Josifovski, R. Kumar, and S. Vassilvitskii, "Nearest-neighbor caching for content-match applications," in *WWW '09: Proceedings of the 18th international conference on World wide web*. New York, NY, USA: ACM, 2009, pp. 441–450.
- [4] F. Falchi, C. Lucchese, S. Orlando, R. Perego, and F. Rabitti, "Caching content-based queries for robust and efficient image retrieval," in *EDBT '09: Proceedings of the 12th International Conference on Extending Database Technology*. New York, NY, USA: ACM, 2009, pp. 780–790.
- [5] H. Hu, J. Xu, W. S. Wong, B. Zheng, D. L. Lee, and W.-C. Lee, "Proactive caching for spatial queries in mobile environments," in *ICDE '05: Proceedings of the 21st International Conference on Data Engineering*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 403–414.
- [6] G. S. Brodal, R. Fagerberg, U. Meyer, and N. Zeh, "Cache-oblivious data structures and algorithms for undirected breadth-first search and shortest paths," in *Algorithm Theory - SWAT 2004*, ser. Lecture Notes in Computer Science, T. Hagerup and J. Katajainen, Eds. Springer Berlin / Heidelberg, 2004, vol. 3111, pp. 480–492.
- [7] S. ZHANG, K. IIDA, and S. YAMAGUCHI, "Cached shortest-path tree : An approach to reduce the influence of intra-domain routing instability," *IEICE transactions on communications*, vol. 86, no. 12, pp. 3590–3599, 2003-12-01. [Online]. Available: <http://ci.nii.ac.jp/naid/110003221599/en/>
- [8] H.-P. Hung and M.-S. Chen, "On designing a shortest-path-based cache replacement in a transcoding proxy," *Multimedia Systems*, vol. 15, pp. 49–62, 2009.
- [9] J.-S. Park, M. Penner, and V. K. Prasanna, "Optimizing graph algorithms for improved cache performance," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 9, pp. 769–782, 2004.