A REPLACEMENT STRATEGY FOR A DISTRIBUTED CACHING SYSTEM BASED ON THE SPATIOTEMPORAL ACCESS PATTERN OF GEOSPATIAL DATA

Rui Li, Xinxing Wang, Xiaolong Shi

State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, Hubei, P.R.China – Ruili@whu.edu.cn

KEY WORDS: Caching; Replacement; Spatiotemporal; LRU; Networked GIS

ABSTRACT:

Cache replacement strategy is the core for a distributed high-speed caching system, and effects the cache hit rate and utilization of a limited cache space directly. Many reports show that there are temporal and spatial local changes in access patterns of geospatial data, and there are popular hot spots which change over time. Therefore, the key issue for cache replacement strategy for geospatial data is to get a combination method which considers both temporal local changes and spatial local changes in access patterns, and balance the relationship between the changes. And the cache replacement strategy should fit the distribution and changes of hotspot. This paper proposes a cache replacement strategy based on access pattern which have access spatiotemporal localities. Firstly, the strategy builds a method to express the access frequency and the time interval for geospatial data access based on a least-recently-used replacement (LRU) algorithm and its data structure; secondly, considering both the spatial correlation between geospatial data access and the caching location for geospatial data, it builds access sequences based on a LRU stack, which reflect the spatiotemporal locality changes in access pattern. Finally, for achieving the aim of balancing the temporal locality and spatial locality changes in access patterns, the strategy chooses the replacement objects based on the length of access sequences and the cost of caching resource consumption. Experimental results reveal that the proposed cache replacement strategy is able to improve the cache hit rate while achieving a good response performance and higher system throughput. Therefore, it can be applied to handle the intensity of networked GISs data access requests in a cloud-based environment.

1. INTRODUCTION

There are more and more human activities in Networked Geographic information services (NGIS), which effect the development of human-environment relationships more and more (Gong, 2006). Some reports show numerous users request the same data in NGIS, Distributed high-speed caching system (DHCS) can cache the frequently accessed geospatial data, reducing the I/O bandwidth for data resources and reducing response delay for public users, getting higher cache hit rate and data sharing (Barish, 2000). However, it is the core issues for DHCS that how to use the limited distributed caches, while ensure the cached data have a higher value to be stored and to be changed as the hotspot changing. Cache replacement strategy can address this issue.

Much useful research has been done into page cache replacement. However, they all used the traditional cache replacement algorithms, such as Least-recently-used (LRU) (Dan, 1990), Least-frequently-used (LFU) (Lee, 1999), First-in-First -out (FIFO) (Dan, 1990), Random Replacement (Dan, 1990), object-based size (Podlipnig, 2003), or some varieties of these algorithms, such as Hyper-G (Wong, 2006),GD-SIZE (Cao,1997), GD*, LRU-K (O'neil, 1993)、LRV (Cao, 1997) etc.. For geospatial data (tile) has a spatiotemporal characteristic in access pattern, these algorithms above cannot be used directly in cache replacement methods for tile. The latest trend in NGIS research is to study and mine access patterns and user interaction modes for large-scale access to big data (Liu, 2009;

Fisher, 2007). Many NGIS logs of user access show that geospatial data is accessed with spatiotemporal locality (Fisher, 2007 b; Yang, 2011; Talagala, 2000; Wang, 2010). Report (Wang, 2009a) thought over the advantage and disadvantage of typical cache replacement algorithms , such as LRU、LFU、FIFO, took into account two important factors, one is the distance between the cached tiles and the current accessed tile, the other is the current time and the user access time to a tile, and proposed a cache replacement algorithm based on the average access interval for tiles. Based on the lifetime of current cached tile and the average cached lifetime for replaced tiles, report (Wang, 2009b) proposed a cache replacement method based on calculating the degree of aging for tiles. These two algorithms all obtain good service performance. However, they still used the accumulated access time for a tile to indicate the spatial locality in tile access. It cannot reflect the nature spatial-correlation and location-correlation between tile accesses.

Therefore, the key issue to cache replacement for tile is to study a combination method which can consider and balance both temporal and spatial locality features in tile access, not only adapting to changes in access distribution of hotspot, but also reducing the frequency of cache replacement. This paper presents a distributed cache replacement method based on tile sequence with spatiotemporal feature in access pattern. The method builds a LRU stack to storage current hot tiles and their popularities based on the sequential feature when tiles are accessed, then structures tile sequence which embodies both temporal and spatial locality in hot-tile access; moreover, chooses a right tile sequence to be replaced.

2. METHOD

2.1 Sequential feature of geospatial data access

The advantage of a pyramid model for tile is to reduce the access times to hard disk, improving access efficiency (Zhang, 2004). Let a tile with coordinates (ℓ,tx,ty),where tx, ty represent the coordinates of the tile block as a unit and ℓ is the layer number in the tile pyramid model. Client calculates the coordinates of the center tile which is in the current browsing view by the location of the center and its longitude and latitude, then request tile by coordinates (ℓ, (tx,ty)) to server. When user navigating, multiple tiles are included in the current browsing view for a client, called as browser window. However, although geospatial tile navigation and Web navigation have certain similarities and the operation of browsing a map can be considered as similar to following a Web page hyperlink, the user can simultaneously browse multiple tiles rather than browsing one single Web page. Thus, tiles which have neighboring access time are sequential in the spatial access pattern. For example, when the size of the current browsing view is a*b and user requests tile (ℓ, (tx,ty)), the server will send a tile sequence whose length is a*b to the client to display. However, tiles in the sequence have spatial location correlation and their accesses time are also sequential.

Cache uses the static random-access memory (SRAM) for caching data, which is distributed in a matrix. When CPU is calling data from cache, row decoder fixes the row address firstly, then column decoder fixes the column address. Thus it fixes a unique memory address of requested cached data, and transfers it to data bus by RAM interface. Generally, SRAM reads data efficiently by a burst mode, which locks a memory row when reading data, then quickly swept out all memory in different columns, reading all bits of data on the column at the same time. Thus, the burst mode can improve the efficiency for reading memory. Since a tile sequence has the spatiotemporal locality when they are accessed, if store tiles in a sequence into consecutive memory, the tiles which have the neighboring accessed time (such as tiles in a browsing view) have the neighboring memory location in cache. It can use the reading characteristic of cache in the burst mode to minimize the response time when user is roaming in NGIS.

2.2 LRU stack

These data which will not be accessed permanently or for a longest time should be replaced in Optimal Page Replacement Algorithm (OPT) (Sun, 2004). LRU algorithm is closest to OPT in algorithm efficiency. It or its varieties are used widely, such as in Google、World Wind，Microsoft Virtual Earth. LRU usually structures a stack to input the accessed data from top to bottom. In this way, the top data always are most recently accessed, and the bottom data always are the least recently accessed.

This paper structures LRU stack to embody the spatiotemporal locality for tile sequence for cache replacement strategy. There are two key issues to be considered to structure a LRU stack. One is how to use LRU stack to express the spatiotemporal locality and spatial relationship between tiles. The other issue is how to use LRU stack to balance the temporal and spatial locality in tile access pattern to get a better replacement decision-making.

We divide the LRU stack into three portions to implement the tile sequence structure and replacement operation,. The first portion is tile receiving-pool. It receives the latest accessed tiles and filters the tiles with higher access probability, preventing them into the stack bottom. The second portion is tile sequence-pool, collecting all accessed tiles which will be structured into sequences and be replaced in the near future. The space size of the pool is floating from 0 to BUFFER_MAX. When the number of tiles in the sequence-pool is BUFFER_MAX, it triggers the process of structuring tiles into different tile sequences which have different lengths. The third portion is replacement-queue-pool, to store and sort tile sequences which will be replaced.

2.3 Replacement flow

For a strict LRU stack, when a new request to tile B arriving, it should place the tile at the top of LRU stack. It brings overhead cost and disadvantage to the manager for tile sequences in our strategy. A tile, which is hit in a tile sequence, should be moved to the stack top and resort all tile sequences. It will increase difficulty for tile sequence generation and sort in the sequence-pool and replacement-queue-pool. This paper proposes a new method to solve the problem. When a tile in the stack is hit, its access-flag is marked as NEW, instead of moving to the stack top; when a tile isn't hit, the method will check the tile on the bottom of receiving-pool. If the tile's access-flag is NEW, then it is moved to stack top and its access-flag is marked as OLD; if its access-flag is OLD, then the tile is moved to sequence-pool, to empty a space for tile B in the receiving-pool. When the size of sequence-pool is BUFFER-MAX, then begin structuring the tiles in the receiving-pool into different sequences, move these sequences to replacement-pool and sort them. When the cached tiles exceed the replacement threshold value, executes the cache replacement algorithm to replace the tile sequence at the bottom of replacement-pool. Thus, the method can use LRU stack well while reducing movements of tile.

2.4 Generation and sorting for tile sequence

The advantage of tile sequence is to improve the reading efficiency for cache. It tries to put the spatiotemporal-related tiles into neighboring cache, to read data by only one CPU instruction and respond the user's request more quickly. Thus, the generation of tile sequence and the sequence replacement method are all based on this idea. It tries to replace the tiles which have neighboring cached location meanwhile, to get successive caching space for subsequent hot accessed tiles. Actually, the replaced tiles also have an access-spatiotemporal-locality. However, how to structure tile sequence which has above characters and how to sort tile sequences in the replacement queue by their storage values, it is the key issue of our proposed replacement method.

The accessed hotspot is changed and users have different operations when having different roaming aims at different time. Thus the lengths of different tile sequences with access-spatiotemporal-locality are different. This paper proposes a method to generate tile sequence using the length of a tile sequence as a weight to measure caching cost. It is a cache replacement algorithm based on tile sequence and its size (SS cache replacement algorithm).

When it is triggered to generate multi-tile-sequence which has different length, SS begins traversing each tile in the sequence-pool and sorts the tiles which have the same row in cache into a sequence. Thus, the caching locations of tiles which are in the same sequence have the same row and neighboring locations. The SS also assigns an H-value for each sequence, which is inversely proportional to its caching cost. According report (Young, 1994), data which has a lower H-value (that is has a higher caching cost) will be replaced firstly, and it is considered as an optimal online caching replacement method (Jin, 2000). Thus, the longer tile sequence will be replaced out cache firstly. Meanwhile, avoiding keeping the shorter sequence which hasn't been accessed for a long time in the cache, the method will add the value of caching cost by a time weighting factor T, to replace the shorter sequence at a proper time. We use the H-value of the latest replaced tile sequence to mark T-value. When a new sequence Seq is added to replacement queue, its H-value is set as Equation1.

$$\mathrm{H(Seq)} = \mathrm{T} + 1/\mathrm{size(Seq)} \qquad (1)$$

$\mathrm{size(Seq)}$ is a function to get the number of tiles in tile sequence Seq, that is to get the size of Seq. The tile sequences which are generated at the same time in the sequence-pool have the same T-value to calculate their H-values, that is they have the same sequence-generation-time. According to the H-value, new generated sequences are sorted with the sequences already in replacement-queue-pool. The sequence with lower H-value moves to the stack bottom, to be replaced firstly. Thus the latest accessed longer sequence will have a higher H-value by the continuous expanded T-value, and avoid to be replaced before the shorter sequence which have a lower H-value and haven't been accessed for a long time. Thus, function H(Seq) balances the temporal locality and spatial locality for sequences in the replacement queue.

3. SIMULATION AND RESULTS

The proposed cache replacement strategy for DHCS was simulated and compared with the classic cache replacement strategy which are used widely, such as FIFO，LFU，LRU，TAIL. The simulation was performed in a networking simulation environment. Six distributed high-speed cache servers were connected using a 1,000-Mbps switch to form a fast Ethernet. The DHCS cache capability is the important factor in cache replacement strategy. The relative size of the cache (RSC) is the ratio of the cache size to the total size for the tiles requested. Therefore, the RSC was varied in simulations carried out to assess the performance in terms of cache hit rate, the average request response time.

3.1 Cache hit rate

The cache hit rate gives an indication of validity of cache replacement. The cache hit rate is the ratio of the tiles which cache response to the total size for the tiles requested. Figure 1 shows a comparison chart of the tile-request cache hit rate of the five strategies for different relative cache sizes. From figure 1, it shows the cache hit rate of any strategy increases with the cache size. FIFO is the one which always has the worst cache hit rate; LRU (which embodies temporal locality to tile access) is better than LFU (which embodies spatial locality to tile access); TAIL (which embodies both temporal and spatial locality to tile access) is better than FIFO, LFU and LRU. However, SS is a bit better than TAIL, although they all consider both temporal and spatial locality to tile access. It's because they have different methods to embody access-spatial-locality. SS uses tile sequence, in which tiles have space- relativity, to embody access-spatial- locality, while TAIL uses the accumulated access time. The method in SS follows spatial characteristic of tile.

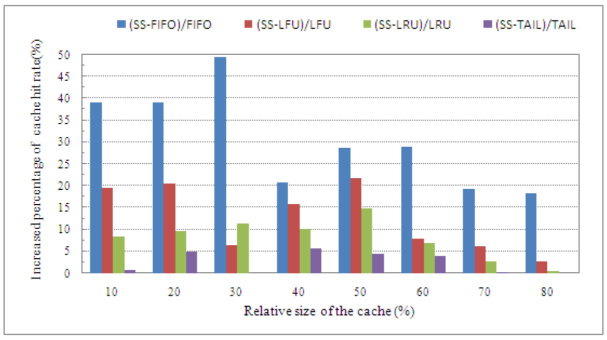

Figure1. Increased percentage of the tile-request cache hit rate

3.2 Average request response time

Figure 2 shows the average request response time of any strategy decreases with the cache size. As the simulation of cache hit rate, FIFO has the worst performance while SS has the best one. TAIL is the second one and LRU is similar as LFU. Figure 2 is the decreased percentage that SS is compared with FIFO，LFU，LRU and TAIL. SS is 6% to 14% lower than TAIL, is 7% to 19% lower than LRU, is 10% to 21% lower than LFU, and is 12% to 23% lower than FIFO. It shows SS has bigger advantage than other strategies in average request response time. It indicates that SS using the physical characteristic of cache can accelerate the access response for users.

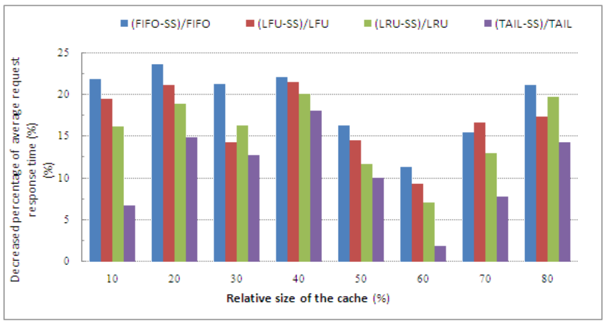

Figure2. Decreased percentage of the tile-request average request response time

4. CONCLUSION

This paper considers the fixed characteristics of tile access pattern, uses the physical characteristics of cache burst mode in reading data, generates tile sequence which has spatiotemporal locality based on a LRU stack, balances the temporal and spatial locality in tile access pattern by a weighted method, to sort the tile sequences in replacement queue to make effective cache replacement strategy. Future work is to study the dynamic change between the sequence-pool and replacement-queue-pool, to adopt the hotspot change in geospatial data access.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 41371370).

References

A. Dan, and T. Don, 1990. An approximate analysis of the LRU and FIFO buffer replacement schemes, SIGMOD '93 Proceedings of the 1993 ACM SIGMOD international conference on Management of data, Vol. 18. No. 1. ACM.

C. W. Yang. M. Goodchild, Q. Y. Huang, D. Nebertc, R. Raskind, Y. Xue, M. Bambacusf, and D. Faye, 2011. "Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing?," International Journal of Digital Earth 4.4, 305-329.

D. Lee, J. Choi, J.H. Kim, S.H. Noh, S. L .Min, Y. Cho, and C. S. Kim, 1999.On the existence of a spectrum of policies that subsumes the least recently used (LRU) and least frequently used (LFU) policies. ACM.SIGMETRICS. P.E. R. 27,134-143.

D. Fisher, 2007a. "How we watch the city: Popularity and online maps," Workshop on Imaging the City, ACM CHI 2007 Conference.

D. Fisher, 2007b. "Hotmap: Looking at geographic attention." Visualization and Computer Graphics, Proceedings of IEEE Symposium on Information Visualization, 13(6), 1184-1191.

E.J. O'neil, P. E. O'neil, and G. Weikum, 1993. "The LRU-K page replacement algorithm for database disk buffering," ACM SIGMOD Record, 22(2).

G. Barish, and O. Katia, 2000. "World wide web caching: Trends and techniques," IEEE Communications magazine, 38(5),178-184.

H. Wang, S. M. Pan, M. Peng, and R. Li, 2010. "Zipf-like distribution and its application analysis for image data tile request in digital earth," Geomatics and Information Science of Wuhan University, 35(3), 356–359.

H. Wang, Z. W. Yu, W. Zeng, and S. M. Pan, 2009a. "The research on the algorithm of spatial data cache in network geographic information service," Acta Geodaetica et Cartographica Sinica, 38(4), 348-355.

H.WANG, Z. W. YU, and W. Zeng, 2009b. "Massive spatial data cache replacement policy based on tile l ifetime and popularity," Geomatics and Information Science of Wuhan University,34(6), 667-670.

J.H. Gong, 2006. “Man-Earth relationships based on virtual geographic environments,” 6th Nat. Conf. Cartography GIS Conf., Wuhan, Hubei, China, Oct. 30.

K-Y. Wong, 2006. "Web cache replacement policies: a pragmatic approach," Network, IEEE, 20(1), 28-34.

N. Talagala, S. Asami, D. Patterson, B. Futernick , and D. Hart,2000."The art of massive storage: a web image archive," Computer, 33(11), 22-28 .

N. Young, 1994. "The K-Server dual and loose competitiveness for paging," Algorithmica, 11,525-541.

P. Cao, and I. Sandy, 1997. "Cost-Aware WWW proxy caching algorithms," Use nix symposium on internet technologies and systems, 12(97).

S. Jin and A. Bestavros, 2000. "Popularity-aware greedy dual-size web proxy caching algorithms," Distributed computing systems, 2000. Proceedings. 20th international conference on. IEEE.

S. Podlipnig and B. Laszlo, 2003. "A survey of web cache replacement strategies," ACM Computing Surveys (CSUR), 35(4), 374-398.

X. H. Liu, J. Z. Han, Y. Q. Zhong, C. D. Han, and X. B. He, 2009. "Implementing WebGIS on Hadoop: A case study of improving small file I/O performance on HDFS," Cluster Computing and Workshops, CLUSTER'09. IEEE International Conference on. IEEE.

Y. S. Zhang, D.C. Gong and J. Liu, 2004. High-resolution remote sensing satellite applications: Imaging model, processing algorithms and application technology. Science Press.

Z. X. Sun, 2004. Operating system tutorial, Higher Education Press.