Vertical Caching: Web Caching for Challenged Networks

Jay Chen
New York University
jchen@cs.nyu.edu

Lakshminarayanan Subramanian
New York University
lakshmi@cs.nyu.edu

ABSTRACT

Many network links in developing regions are low-bandwidth or intermittent. In addition to the inherent connection constraints, web browsing over these networks is an extremely painful experience due to inappropriate cache design. A single cache miss during browsing may cause a stall on the order of minutes or even hours. To improve web cache performance, we propose a new model of web caching particularly tailored for low-bandwidth and intermittent environments called vertical caching. Vertical caching extends existing caching mechanisms based on URLs to aggregates of cached pages across *topics*. We show that vertical caching has the potential to dramatically improve performance in these settings.

1. INTRODUCTION

In most of the developing world, low-bandwidth or intermittent networks are common, and web browsing over these networks is extremely poor. While improving rural connectivity is the most direct approach [5], improving web cache performance is also crucial [1, 3]. To improve cache performance, we introduce a *vertical caching* model that organizes a cache around *topics* rather than solely URLs. In conventional caching, unless there is an exact URL match to a cached content, a new page must be downloaded. However, in practice the same information may reside in the cache aliased under a different URL or similar equivalent content that satisfies a user's request may be available in the cache.

Coda introduced idea of disconnected hoarding for often used content for offline availability and synchronization when connection is re-established [4]. Since then, many descendants expanded upon the idea of caching and prefetching for disconnected operation. In contrast to existing techniques, vertical caching proposes a new type of locality to exploit when deciding deciding which pages to cache or prefetch. In vertical caching, pages are clustered based on content topic rather than by access over time windows or explicit references to each other via hyperlinks. Collaborative caching [3] and HashCache [1] have been proposed as a developing region specific solutions, but are otherwise unrelated ideas. In collaborative caching, common interest across users is exploited to improve caching across nodes with often limited resources. In vertical caching, we assume that there is common content across pages that may be leveraged. HashCache introduced techniques for scaling up caching for cheap commodity laptops with limited memory. Our work is the most similar to value-based caching [6], vertical caching allows matching based on the information between in pages rather than exact matching between pieces of data.

One of the primary challenges in vertical caching is identifying topics within a cache. A topic is simply a set of related pages relevant to an interest of users. Once the topics are identified, the vertical cache can organize the cache contents based on topics (or verticals), and potentially expose the topics to the end-user through local search [2, 7]. In this work, we describe one instantiation of vertical caching, and demonstrate that it is possible to quickly identify useful topics of interest that recur regularly. To accommodate vertical caching, we redefine the notion of a cache hit and introduce a new cost metric to more appropriately capture the value of pages to users in challenged network environments.

2. TOPIC IDENTIFICATION

Our system attempts to automatically identify topics from three sources of information. These sources are all based on information in the request log of the cache: URLs, search queries, and page contents. Due space constraints we only outline URL and search query patterns in this paper.

In our system we define topics to be a description of a preference for a particular set of web pages. Each topic consists of: (a) description of the topic, (b) a set of pages belonging to a topic, (c) a classifier for deciding whether a page belongs to the topic, and (d) cost or value of the topic. The description of the topic, (a), is simply a label used for navigating across topics in the cache and used for finding more documents belonging to the topic. This description may be automatically extracted from the pages belonging to the topic or manually defined by the user. A topic description may be a URL pattern (e.g. a.com/b/*) or a set of correlated terms (e.g. search terms, or terms appearing in a set of documents). The set of pages that belong to a topic, (b), are also constructed automatically. These pages are used by the cache to display pages belonging to a topic during browsing. The classifier, (c), is trained on the set of pages in an existing topic and used to classify new pages into topics. The cost, (d), is used by the cache to evict the least valuable topics or pages.

Domain Topics - Domain topics are extracted from the URL patterns that occur in an access log. Domains and sub-domains are identified if requests appear frequently enough in the cache such that future requests have a high probability of producing a significant number of hits. The topic extraction engine groups web page requests into a hierarchal tree of buckets. At the top are the second level domains. Branching out from these are the sub-domains, and sub-directories within these domains. Associated with each bucket is a cost C_b. Where C_b is the aggregate user time required to fetch content belonging solely to this sub-domain. This cost function explicitly takes into account the reality of the web browsing experience for users in environments where the cost of fetching N bytes may vary radically vary. A domain topic is considered "identified"

ACM X-XXXXX-XX-X/XX/XX.

once a threshold $T = 3$ of requests are made that belong to the topic. Figure 1 (a) shows how cost trees relate to domain topics.

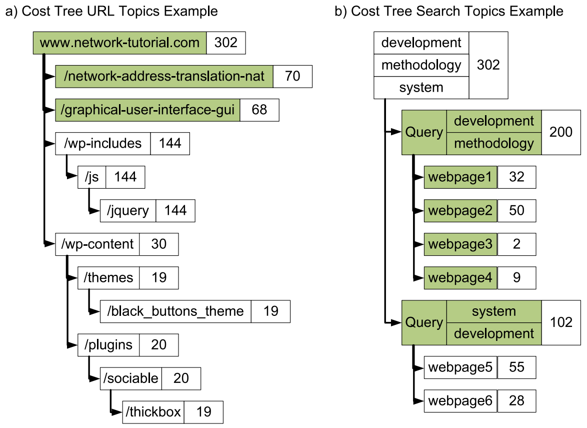
a) Cost Tree URL Topics Example
www.network-tutorial.com 302
/network-address-translation-nat 70
/graphical-user-interface-gui 68
/wp-includes 144
/js 144
/jquery 144
/wp-content 30
/themes 19
/black_buttons_theme 19
/plugins 20
/sociable 20
/thickbox 19
b) Cost Tree Search Topics Example
development methodology system 302
Query development methodology 200
webpage1 32
webpage2 50
webpage3 2
webpage4 9
Query system development 102
webpage5 55
webpage6 28

Figure 1: Example cost trees for (a) domain topics and (b) query topics. Costs are hierarchical and cumulative. Items highlighted green are used for navigation or prefetching.

Query Topics - The query analysis method of topic identification consists of identifying and extracting 'query-like' phrases from the user requests to find a smaller set of more general topics. There are many ways to extract useful query topics. The simple method we use is to split requests into search sessions based on query overlap and time of request. On a per-client basis, if a query is made that overlaps with a previous query made within the last time threshold $S = 120$ seconds then the query is considered as part of the same search session. Otherwise, a new search session is created. All requests made while a search session is active belong to the current search session. The union of each query terms of each search session then form the query topic. An example of how cost trees relate to query topics is shown in Figure 1 (b).

3. PRELIMINARY RESULTS

Evaluating a vertical cache based on cache hit rate is not particularly appropriate because vertical caching expands the functionality of the cache and relaxes the binary concept of a cache hit. We define a new metric "topic hit rate" to capture the idea of having a page in the cache that satisfies a request to some degree. A request results in a topic hit if there exists a topic identified by our system for that request.

Using log data from an intermittently connected secondary school classroom in Kenya, we show that we can extract useful topics. Our log contains over $100,000$ requests and over $1,400$ search queries gathered over a period of four months. The usage of the cache is bursty due to multiple scheduling and classroom constraints, only 41 days out of 160 have any activity.

Topic Hit Rate - Figure 2 illustrates the percentage of topic hit rate of user requests as the number of domain topics chosen varies. The top domain topics in this figure are chosen are chosen by the highest coverage. Topics are defined in this case based on a threshold T requests belonging to a particular subdomain. We make three observations: First, relatively few topics are required to cover over 75% of the topics in our 4 month long log. Second, the resultant URL topic hit rate is insensitive to choice of the threshold T other than the tail of the graph is cut off. Third, the coverage is higher than the existing cache hit rate (66%) after approximately 40 domain topics. Topic coverage for query topics is defined in the same way. We constructed these topics from $1,432$ search queries, and find that after 10 query topics the topic hit rate is greater than the cache hit rate of 66%. As few as 50 query topics are sufficient to cover 75% of requests. The analog to byte hit rate of traditional caching is the percentage of cost of requests covered by the top topics, or the "topic cost hit rate". We find similar results for topic cost hit rate.

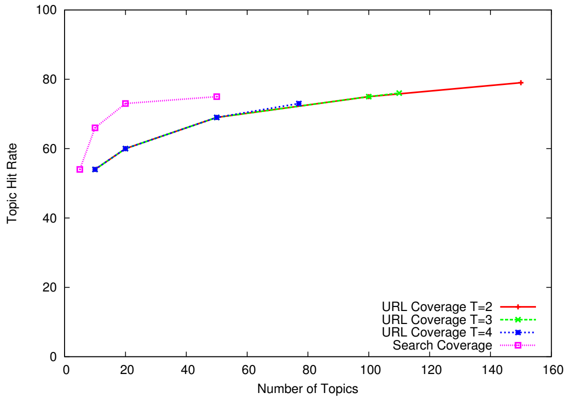

Figure 2: Topic hit rate of domain and query topics.

Topic Recurrence and Stability - We next attempt to understand the utility of topics in another way: "how long does it take to detect a topic, and how useful is the topic over time?" From our preliminary results, we observed that, for threshold $T = 3$, most of the domain topics are identified within the first 24 hours. This is sufficient for an intermittent network with a mechanical backhaul, and may be reduced by lowering the threshold, T. We also found that the identified topics recurred regularly over time. Finally, we found that while requests are bursty over short timescales of hours, topic stability is fairly regular considering that the overall usage of the system is bursty.

Our experiments confirm that in small rural community contexts there is a commonly used core of web pages that is even more cohesive than what has been observed for the web in general. It is precisely this increased topic cohesion that vertical caching capitalizes upon. Our evaluation complements existing results in this space that suggest potential benefits to collaborative caching in small rural communities (where collaborative caching is for weaker distributed nodes) [3].

4. REFERENCES

[1] BADAM, A., PARK, K., PAI, V., AND PETERSON, L. Hashcache: Cache storage for the next billion. In *Proceedings of the Sixth USENIX symposium on Networked Systems Design and Implementation* (2009).

[2] CHEN, J., SUBRAMANIAN, L., AND LI, J. Ruralcafe: web search in the rural developing world. *Proceedings of the 18th International Conference on World Wide Web* (2009).

[3] ISAACMAN, S., AND MARTONOSI, M. Potential for collaborative caching and prefetching in largely-disconnected villages. *Proceedings of the ACM Workshop on Wireless Networks and Systems for Developing Regions* (2008).

[4] KISTLER, J., AND SATYANARAYANAN, M. Disconnected operation in the coda file system. *ACM Transactions on Computer Systems (TOCS)* (1992).

[5] PATRA, R., NEDEVSCHI, S., SURANA, S., SHETH, A., SUBRAMANIAN, L., AND BREWER, E. Wildnet: Design and implementation of high performance wifi based long distance networks. In *Proceedings of the Fifth USENIX Symposium on Networked Systems Design and Implementation* (2007).

[6] RHEA, S., LIANG, K., AND BREWER, E. Value-based web caching. *Proceedings of the 12th International Conference on World Wide Web* (2003).

[7] THIES, W., PREVOST, J., MAHTAB, T., CUEVAS, G., SHAKHSHIR, S., ARTOLA, A., VO, B., LITVAK, Y., CHAN, S., HENDERSON, S., ET AL. Searching the world Wide Web in low-connectivity communities. *Proceedings of the 11th International Conference on World Wide Web* (2002).