A Hybrid Strategy for Caching Web Search Engine Results

Fabrizio Silvestri
Dipartimento di Informatica
Università di Pisa - Italy
fasilves@di.unipi.it

Tiziano Fagni
ISTI - CNR
Pisa, Italy
t.fagni@guest.cnuce.cnr.it

Salvatore Orlando
Dipartimento di Informatica
Università Ca' Foscari - Italy
orlando@dsi.unive.it

Paolo Palmerini
ISTI - CNR
Pisa, Italy
p.palmerini@cnuce.cnr.it

Raffaele Perego
ISTI - CNR
Pisa, Italy
r.perego@cnuce.cnr.it

ABSTRACT

This work discusses the design and implementation of an efficient caching system aimed to exploit the locality present in the queries submitted to a Web Search Engine (WSE). We enhance previous proposals in several directions. First we propose the adoption of a hybrid strategy for caching, and then we experimentally demonstrate the superiority of our hybrid strategy. Further we show how to take advantage of the spatial locality present in WSE query logs by exploiting a sort of adaptive prefetching strategy.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search process*; D.2.8 [**Software Engineering**]: Metrics—*performance measures*

Keywords

Caching, search engines, query log analysis

1. INTRODUCTION

In this work, we are interested in studying effective policies and implementations for server-side WSE caches. In particular, we will analyze the behavior of a one-level cache in terms of miss and hit rate.

We propose a novel hybrid replacement policy to adopt in the design of our cache. According to our hybrid caching strategy, the results of the most frequently accessed queries are maintained in a static cache of fixed size, which is completely rebuilt at fixed time intervals. Only the queries that cannot be satisfied by the static cache compete for the use of a dynamic cache. Our hybrid cache represents an effective and fast way to address both *recency*, and *frequency* of occurrences criteria. While the static cache maintains results of queries that are globally frequent, a simple and fast policy like LRU, which only takes into account query reference recency, could be adopted for the dynamic cache.

2. ANALYSIS OF THE QUERY LOGS

In order to evaluate the behavior of different caching strategies we used query logs from the Tiscali and EXCITE search engines. In particular we used *Tiscali1*, and *Tiscali2*, two different traces of the queries submitted to the Tiscali WSE engine (www.janas.it), and *Excite*, a publicly available trace of the queries submitted to the EXCITE WSE (www.excite.com) (Table 1). Our analysis shows that a) there are a certain number of queries which are frequently requested, and b) given a request for the i-th page of results, the probability of having a request for page $(i + 1)$ in the future is about 0.1 for $i = 1$, but becomes approximately 0.5 or greater for $i > 1$.

Table 1: Main characteristics of the query logs used.

Query log	queries	distinct queries	date
Tiscali1	1,352,079	626,885	March 2001
Tiscali2	3,278,211	1,538,934	April 2002
Excite	2,475,684	1,598,908	September 1997

3. OUR HYBRID CACHE

Several works have been proposed in these last years about caching of WSE results [3, 1, 2]. Markatos already studied pure static caching policies for WSE results, and compared them with dynamic caching ones [1]. The rationale of adopting a static policy, where the entries to include in the cache are statically decided, relies on the observation the most popular queries submitted to WSEs do not change very frequently. On the other hand, several queries are popular only within relatively short time intervals, or may become suddenly popular due to, for example, un-forecasted events (e.g. the 11^{th} September 2001 attack). These considerations suggested us to adopt a hybrid (static+dynamic) strategy for caching query results, where the results of most popular queries are maintained in a static cache, and only the queries that cannot be satisfied with the content of the static cache compete for dynamic cache entries.

Our caching system processes a query of the form *(keywords, page_no)*, and if the corresponding page of results is not found in the cache, it forwards to the core query service WSE an expanded query, requesting k consecutive pages starting from page *page_no*, where $k \geq 1$ is the *prefetching factor*.

Prefetching clearly involves additional load on the WSE, and might negatively affect the replacement policy adopted, since a miss causes the replacement of several pages with new pages till not accessed. In order to maximize the benefits of prefetching, and, at the same time, reduce the additional load on the WSE, an effective heuristic to adopt is to prefetch additional pages only when

WWW2003, May 20–24, 2003, Budapest, Hungary.
ACM xxx.

the cache miss has been caused by a request for a page different from the first one. In this way, since the prefetched pages will be actually accessed with sufficiently high probability, we avoid to fill the cache with pages that are accessed only rarely and, at the same time, we reduce the additional load on the core query service of the WSE.

4. EXPERIMENTAL RESULTS

Since our hybrid caching strategy requires the blocks of the static section of the cache to be preventively filled, we partitioned each query log into two parts: a *training set* which contains $2/3$ of the queries of the log, and a *test set* containing the remaining queries used in the experiments. The N most frequent queries of the training set were then used to fill the cache blocks: the first $f_{static} \cdot N$ most frequent queries (and corresponding results) were used to fill the static portion of the cache, while the following $(1 - f_{static}) \cdot N$ queries to fill the dynamic one. Note that, according to the scheme above, before starting the tests not only the static blocks but also the dynamic ones are filled, and this holds even when a pure dynamic cache ($f_{static} = 0$) is adopted. In this way we always starts from the same initial state to test and compare the various possible configuration of our hybrid cache, obtained by varying the factor f_{static}.

Figure 1 shows the hit rates achieved on the *Tiscali2* query log by varying the percentage of cache used as static f_{static}, and the prefetching factor k, whose tested values were 1 (no prefetching), 3, 5 and 7. The replacement policy adopted for the dynamic portion of the cache was always LRU. We chose the LRU policy because our tests demonstrated that while for cache organizations where the number of dynamic cache blocks predominates (i.e., small values of f_{static}), the *LRU-2S*, *FBR*, or *LRU/2* replacement policies outperform *LRU* and *2Q*, the opposite often holds when the percentage of static cache blocks is increased.

Figure 1.(a) refers to tests conducted by maintaining the prefetching factor k *constant* for all the pages requested. Conversely, the curves reported in the plot of Figure 1.(b) are relative to tests conducted by prefetching additional pages only when a cache miss is caused by a request regarding a page of results different from the first one, i.e., by employing the *adaptive* heuristic discussed above.

From both the plots we can see that prefetching is able to effectively exploit spatial locality present in the logs: we obtained remarkable improvements in the cache hit rate. On the other hand, by comparing the curves plotted in Figure 1.(a) and 1.(b), we can see that while the constant prefetching slightly outperforms the adaptive prefetching heuristic when our hybrid strategy is used, the opposite holds for a pure dynamic cache. However, this higher hit rate achieved by exploiting constant prefetching must be paid with an huge increase in the load on the core WSE query service. By profiling execution, we measured that for $f_{static} = 0.8$, in the case of a constant prefetching factor 3, only 5.7% of the prefetched pages are actually referred by the following queries, while when the adaptive prefetching heuristic is adopted (with $K_{max} = 3$), this percentage increases up to 46%. We can thus conclude that the proposed adaptive heuristic constitutes a good trade-off between the maximization of the benefits of prefetching, and the reduction of the additional load on the WSE.

5. CONCLUSIONS

In this work we presented a new hybrid policy for caching the query results of a WSE. The main enhancement over previous proposals regards the exploitation of past knowledge about queries submitted to the WSE to make more effective the management of our cache. In particular, since we noted that the most popular queries that are submitted to a WSE do not change very frequently, we maintain these queries and associated results in a read-only static section of our cache. Only the queries that cannot be satisfied by the static cache section compete for the use of a dynamic cache. The benefits in adopting our hybrid caching strategy were experimentally shown on the basis of tests conducted with three large query logs. In all the cases our strategy remarkably outperformed purely static or dynamic caching policies. We evaluated the hit-rate achieved by varying the percentage of static blocks over the total, the size of the cache, as well as the replacement policy adopted for the dynamic section of our cache. Moreover, we showed that WSE query logs also exhibit spatial locality. Users, in fact, often require subsequent pages of results for the same query. Our caching system takes advantage of this locality by exploiting a sort of adaptive prefetching strategy.

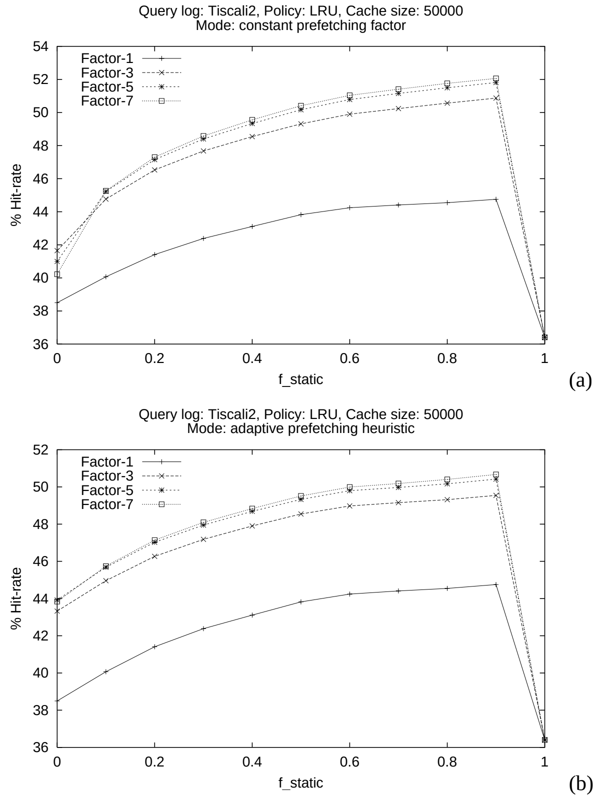

Figure 1: Hit rate for different prefetching factors as a function of the ratio between static and dynamic cache entries: (a) with constant prefetching factor, and, (b) with adaptive prefetching factor.

6. REFERENCES

[1] Evangelos P. Markatos. On caching search engine results. In *Proc. of the 5th Int. Web Caching and Content Delivery Workshop*, 2000.

[2] P.C. Saraiva, E. Silva de Moura, N. Ziviani, W. Meira, R. Fonseca, and B. Ribeiro-Neto. Rank-preserving two-level caching for scalable search engine. In *SIGIR'01*, 2001.

[3] Y. Xie and D. O'Hallaron. Locality in search engine queries and its implications for caching. In *Proceedings of IEEE INFOCOM 2002, The 21^{st} Annual Joint Conference of the IEEE Computer and Communications Societies*, 2002.