

Cooperation schemes between a Web server and a Web search engine

Carlos Castillo
Center for Web Research
Computer Science Department, Universidad de Chile
ccastill@dcc.uchile.cl

Abstract

Search engines provide search results based on a large repository of pages downloaded by a web crawler from several servers. To provide best results, this repository must be kept as fresh as possible, but this can be difficult due to the large volume of pages involved and to the fact that polling is the only method for detecting changes.

In this paper, we explore and compare several alternatives for keeping fresh repositories that involve some degree of cooperation from servers.

1. Introduction

Search engines keep an index based on pages downloaded from the web by a robot or crawler. As the number of publicly available web pages increases, the problem of keeping this index up-to-date with changes is more difficult [18], and usually an important proportion of the index is outdated [25].

Web crawlers can use an important amount of network and processor resources from web servers, specially if they don't follow existing rules of "good behavior" [15]. Web crawlers tend to visit many more pages than humans and are believed to account for at least 16% of the requests [20]. Many of the requests are to unmodified resources, and can be avoided if the server cooperates with the crawler.

There are several things that a webmaster can do to improve the representation of a web site on the search engine's index and to prevent unnecessary visits from crawlers. In this paper, we show some existing techniques and propose new ones.

The next section outlines previous work in this area, Section 3 presents the cooperation schemes and Section 4 is about how to evaluate them in terms of cost. Section 5 presents some concluding remarks.

2. Previous Work

In this paper, we only study the cooperation between web servers and crawlers, not between crawlers: this issue is studied in [19] and [11].

There are several methods for keeping mirrors (replicas) of information services they include RSYNC [27], that generates a series of fingerprints for "chunks" of data, and CTM [13] that is a method for sending differences via e-mail.

A specific proposal for pushing last-modification data to web crawlers is presented in [12]. A more general Internet notification system was presented in [7]. A proposal that uses a series of files containing descriptions of web pages, was presented in [6].

Extensions to HTTP that use differences of content and compression to significantly reduce the total transfer time of web pages when using proxies were analyzed in [21].

DASL [24], the WebDAV searching and locating protocol, is a proposed extension that will allow searching the web server using an HTTP query with certain extensions, but neither the query syntax nor the query semantics are specified by the protocol.

3. Cooperation schemes

The standard HTTP transaction in which a web crawler fetches a page from a web server is shown in Figure 1. Note that meta data (information about the page) is downloaded along with the data.

The cooperation schemes we are going to study in this paper can be divided in two main groups: *interrupt* and *polling*. A crawler may use one of them or a combination of different schemas. In the *interrupt* (or *push*) schemes, the web server begins a transaction with the search engine whenever it is necessary. This is similar to the relationship between the main processor and a hardware device (network card, scanner, etc.) in a modern computer. In the *polling* (or *pull*) schemes, the search engine periodically requests data from the web server, based on search engine policies.

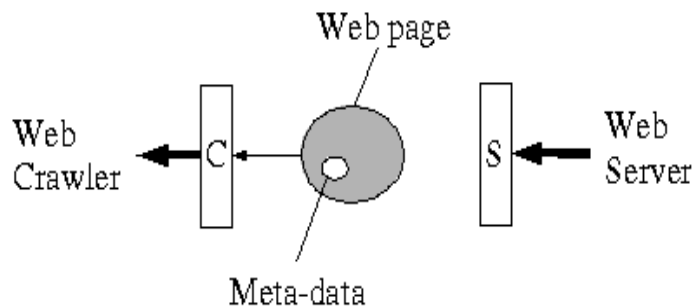


Figure 1. Without cooperation, the web crawler issues a request and then downloads the web page and the meta-data together.

The schemes studied in this paper are summarized in Table 1.

| Transferred data | Interrupt version | Polling version |
|------------------|-------------------|---------------------|
| Meta-data | Notify updates | Serve meta-data |
| Differences | Send differences | Serve differences |
| Pages | Send changed | Serve if-modified |
| Batches | Send batch | Serve multi-pages |
| Site | Send entire site | Serve entire site |
| Mixed | Remote agent | Filtering interface |

Table 1. Summary of the cooperation schemes analyzed in this paper. All of them have two versions: interrupt (push) and polling (pull).

Before we get into the details of each scheme, there are some issues we must mention that are, in some sense, orthogonal to the scheme used:

Compression can be used for decreasing transmission cost at the expense of using more processing power on the server side.

Privacy issues. In practice when using a web crawler it is not uncommon to download files that were linked by mistake; we have even found complete plain-text password files while crawling!.

Index update capabilities are very reduced in global-scale web search engines: constraints in terms of disk space are the most important limitation.

Search engine “spamming” occurs whenever web server administrators try to get undeserved high ratings in the search engines.

Structure and HTML markup used in a website affects the visibility of its pages by search engine crawlers [23].

3.1. Interruption-based cooperation

The search engine must subscribe with the web server to start receiving these notifications.

Send meta-data of updates: The update notification is sent whenever there is one or several updates on the web server. Examples: the Keryx notification service, developed in the apogee of push-based content delivery in 1997 [7], Fresh Flow [12].

Send differences of content: Whenever an event happens, the web server sends a file containing the difference between the last version and the current one. Examples: CTM [13], based on electronic mail and the patch [29] program.

Send changed pages. The web server sends the complete text of all updated pages or new pages when modifications are made. Examples: this was typical in push technologies [14].

Send multi-pages updates. The web server sends batches of modified pages according to a schedule defined by the web server. This can be useful if the updates are regular and involve several pages; for example, in the website of a daily or weekly newspaper.

Send entire site. The web server sends the entire site. This is useful, for instance, for uploading an entire web site when the site is publicly available for the first time.

Remote agent. The web server executes software provided by the search engine, that includes instructions to identify important pages for the search engine, and to detect changes in those pages that are relevant for the search engine. This is a typical application for a mobile agent [16]. Also, this software can help the search engine to fetch data from “near” servers, as proposed in [26].

3.2. Polling-based cooperation

Serve meta-data of updates. A file containing last-modification data -and probably file size- is served. This file should contain a description of many pages on the web site. In the case of single files, the HTTP protocol provides HEAD requests, and multiple HEAD requests can be pipelined, but this is not as good as serving a single, concise file. Examples: the distribution and replication protocol (DRP) [28]. In [6], files containing information about changes are served. In RDF [17], there is the possibility of informing time-related data about the resources. In Web-DAV [30], the PROPFIND method allows to query for properties of documents, and the proposed BPROPFIND method allows to query for properties of groups of documents.

Serve differences of content. The web server provides a series of differences between a base version and newer versions. Examples: HTTP Delta responses proposed in [21] that use the content-encoding field of HTTP re-

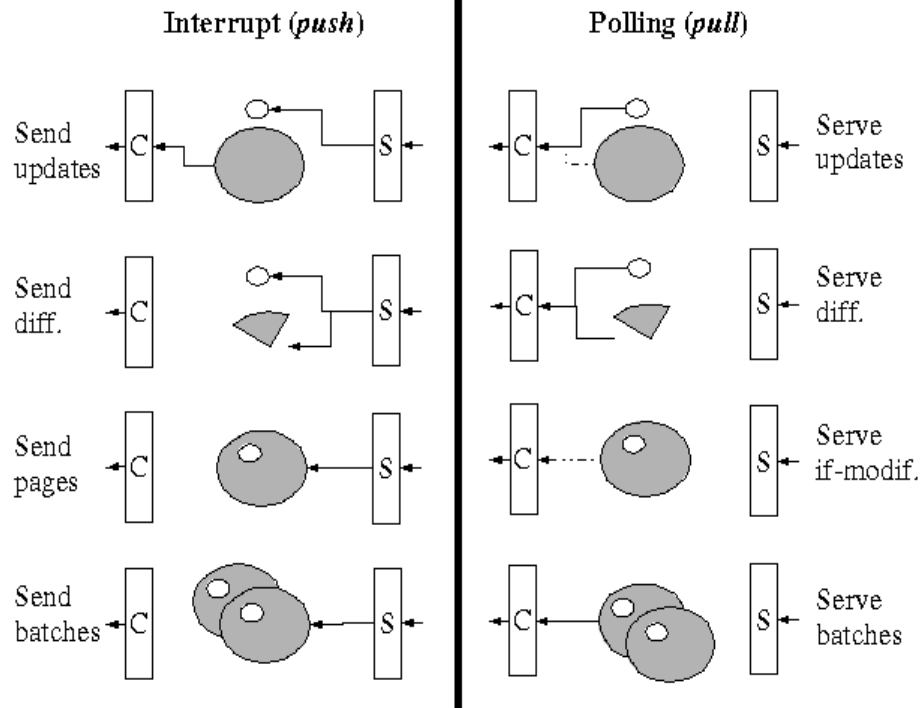


Figure 2. Diagrams of several schemes of cooperation. The arrow between the Web crawler “C” and the Web server “S” represents who initiates the connection. The small, white circle represents meta-data and the large, gray circle represents the contents of the page.

sponses; also in many cases, source code for popular free software can be updated using the `patch` [29] program. Differences in terms of structural changes in the links, can be encoded using tables as in *WHOWEDA* [5].

Serve pages only if modified. The web crawler can avoid downloading a file if it has not been modified. This can be done providing the date or the last visit, or an *entity-tag* (E-Tag): a *fingerprint* identifying the text of the document. Examples: HTTP/1.1 [10] If-Modified-Since headers, used by a minority of crawlers [6], but supported by most web servers.

Serve multiple pages on one request. The overhead associated with multiple requests can be avoided by requesting a series of pages in one request. Examples: this is default in HTTP/1.1 [10] “Keep-Alive”; in this case, pipelining of the requests can also be used.

Serve entire site. As latency is a very important component in web page transfers, specially in the case of many small files with many changes. Examples: Linux distributions are usually distributed in whole CD-sized images.

Filtering interfaces. The web servers provide a standard method of answering queries from the crawler, such as “give me all the pages that have changed since this date”. A pow-

erful filtering interface could also include requests for differences, page sizes or local importance. Examples: DASL for searching web servers [24], RSYNC [27] for mirroring content, CIP (Common Indexing Protocol) [3, 4, 2]; a general framework for this are Web Services [9].

Some of the methods we have shown can be described with diagrams as in Figure 2.

4. Cost analysis

4.1. Costs for the web server

We will consider per-page costs and benefits.

- b : Benefit for the web server from a page-view.
- c_n : Network cost for serving one page: bandwidth cost.
- c_p : Processing cost for serving one page: servers cost.

We should have $b \geq c_n + c_p$, otherwise, the server would not be able to pay the operation costs, however, some web sites are financed by revenues from other sources. Also, in general $c_n > c_p$.

| Strategy | Network cost | Processing (server) | Processing (crawler) | Freshness improvement |
|---------------------------------|--------------|---------------------|----------------------|-----------------------|
| Send meta-data of updates | + | + | | High |
| Send differences of content | -- | ++ | + | High |
| Send changed pages | - | + | | High |
| Send batch update | + | + | | High |
| Send entire site in one file | ++ | + | | High |
| Remote agent | -- | ++ | - | High |
| Serve meta-data of updates | + | + | | Normal |
| Serve differences of content | -- | ++ | + | Normal |
| Serve pages only if modified | - | | | Normal |
| Serve many pages in one request | - | | | Normal |
| Serve entire site in one file | ++ | + | | Normal |
| Filtering interface | -- | ++ | - | High |

Table 2. Relative costs of server cooperation schemes analyzed in this paper, against the base case where no cooperation exists: + means more cost, - means less cost. The right column is the expected benefit in terms of freshness.

Estimates: We cannot measure these quantities, but we can make some estimates: as of June 2003, the cost-per-click of an advertising campaign on the Web is about US\$ 0.05, so probably $b \geq 0.05$. On the other end having a web server costs about US\$ 10 for 5 gigabits of traffic, or 625Mb; if each page is 40Kb on average, it is enough for 15,000 page-views; notice that network bandwidth is usually “overbooked” in popular virtual servers, probably by a factor of 2 or 3, so a rough estimate of the cost is: $c_n + c_p \leq 0.002$.

This means that if we only account for web server usage, serving a web page costs at most $1/25$ of the benefit, and this is probably the main explanation for the huge success of the world wide web as a platform for publishing information. The main source of cost when keeping a large website is not the web hosting, but rather the cost of producing and maintaining the contents.

In Table 2 we provide a rough estimation of relative costs associated with these cooperation schemes. Network bandwidth savings are the product of not dealing with unnecessary requests from the crawlers, and costs, from sending more than is necessary. Processing costs involve keeping meta-data, calculating differences, etc. Benefits arise from increased freshness on the web search engine.

Which is the best strategy for the web server ? This will depend on the price the web server is willing to pay; if this is minor, then using server software that correctly implements HTTP/1.1 is the best option. If the price is moderate, then serving and sending meta-data of updates is the best option. If the server wishes to invest more resources, it can benefit from providing content differences and/or a filtering interface for the crawlers.

4.2. Costs for the crawler

The main costs for the crawler for each page are: polling or receiving and interrupt, downloading and processing pages.

The freshness of the repository is higher in interrupt-based strategies. The costs for the crawler are summarized in Table 2. Network cost for the crawler is the same as for the server.

Which is the best strategy for the crawler ? A remote agent or filtering interface can help to distribute the workload of the search engine, specially if servers cooperate in pre-processing documents or in generating partial indexes.

The extreme case of using an agent is when the web server generates the (partial) inverted index and then sends it to the search engine, that only needs to perform a merge. In this case, the crawling problem is simplified, and turns into polling or pushing of indexes.

4.3. Overall cost

It is very important to consider that not all web servers are equal, and the distribution of “quality” on the web is, by all measures, very skewed, so the search engine should try to get the most important and larger web servers to cooperate first.

Note that by inspecting Table 2, it is clear that all the schemes that do not require extra cost for the web server are already implemented in HTTP.

Difference-based schemes are useful if normal clients can benefit from them -not only the web crawlers, but also the general public using enabled browsers or cache services.

If the request-response paradigm is enforced strictly, then the only scheme that can provide high benefits in terms of freshness is a filtering interface.

5. Concluding remarks

How probable is the wide adoption of cooperation strategies ? Web site administrators can provide cooperation if it is not so costly and means an important benefit for them. This benefit should come mainly in terms of exposition on the web search engines. We consider that the reductions on load for the web server are probably not enough by themselves to justify the adaption of a cooperation strategy.

With the emergence of web services, filtering strategies could be an interesting possibility for the near future, as they can help crawlers and other autonomous agents to interact with web servers at a more meaningful level, bringing the networked nature of the web one step further.

References

- [1] Search engine referrals nearly double worldwide. www.websidestory.com/pressroom/pressreleases.html?id=181, 2003.
- [2] J. Allen, P. Leach, and R. Hedberg. RFC 2653: CIP transport protocols. www.ietf.org/rfc/rfc2653.txt, 1999.
- [3] J. Allen and M. Mealling. RFC 2651: The architecture of the Common Indexing Protocol. www.ietf.org/rfc/rfc2651.txt, 1999.
- [4] J. Allen and M. Mealling. RFC 2652: MIME objects definitions for the Common Indexing Protocol. www.ietf.org/rfc/rfc2652.txt, 1999.
- [5] S. Bhowmick, S. K. Madria, and W. K. Ng. Detecting and representing relevant web deltas in WHOWEDA. *IEEE Transactions on Knowledge and Data Engineering*, (2):423–441, 2003.
- [6] O. Brandman, J. Cho, H. Garcia-Molina, and N. Shivakumar. Crawler-friendly web servers. In *Proceedings of the Workshop on Performance and Architecture of Web Servers (PAWS)*, Santa Clara, California, 2000.
- [7] S. Brandt and A. Kristensen. Web push as an Internet Notification Service. In *W3C workshop on push technology*, 1997.
- [8] B. Charny. Wireless web embraces “push”, 2002.
- [9] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. WSDL: Web services description language. www.w3.org/TR/wsdl, 2001.
- [10] J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. HTTP/1.1 - Hypertext Transfer Protocol. w3.org/Protocols/rfc2616/rfc2616.html, 1999.
- [11] L. Gravano, K. C.-C. Chang, H. Garcia-Molina, and A. Paepcke. STARTS: Stanford proposal for internet meta-searching. In J. Peckham, editor, *Proceedings of International Conference on Management of Data (SIGMOD)*, pages 207–218. ACM Press, 1997.
- [12] V. Gupta and R. H. Campbell. Internet search engine freshness by web server help. In *Proceedings of the Symposium on Internet Applications (SAINT)*, pages 113–119, 2001.
- [13] P.-H. Kamp. OpenBSD CTM. www.openbsd.org/ctm.html, 2003.
- [14] T. Kapyla, I. Niemi, , and A. Lehtola. Towards an accessible web by applying push technology. In *4th ERCIM Workshop on “User Interfaces for All”*, 1998.
- [15] M. Koster. Robots in the web: threat or treat ? In *ConneXions*, 4(4), 1999.
- [16] D. B. Lange and M. Oshima. Seven good reasons for mobile agents. *Communications of the ACM*, 42(3):88–89, 1999.
- [17] O. Lassila and R. Swick. World wide web consortium - rdf. www.w3.org/TR/REC-rdf-syntax, 1999.
- [18] S. Lawrence and C. L. Giles. Accessibility of information on the web. *Nature*, 400(6740):107–109, 1999.
- [19] G. L. McLearn. Autonomous cooperating web crawlers, 2002.
- [20] D. Menasce, V. Almeida, R. Riedi, F. Pelegrinelli, R. Fonseca, and W. M. Jr. In search of invariants for e-business workloads. In *Second ACM Conference on Electronic Commerce*, 2000.
- [21] J. C. Mogul, F. Douglis, A. Feldmann, and B. Krishnamurthy. Potential benefits of delta encoding and data compression for HTTP. In *SIGCOMM*, pages 181–194, 1997.
- [22] J. Nielsen. Statistics for traffic referred by search engines and navigation directories to useit. www.useit.com/about/searchreferrals.html, 2003.
- [23] S. Raghavan and H. Garcia-Molina. Crawling the hidden web. In *Proceedings of the Twenty-seventh International Conference on Very Large Databases*, 2001.
- [24] J. Reschke, S. Reddy, J. Davis, and A. Babich. DASL - dav searching and locating protocol. www.webdav.org/dasl/, 2002.
- [25] D. Sullivan and C. Sherman. Search Engine Watch. www.searchenginewatch.com/reports/, 2002.
- [26] W. Theilmann and K. Rothermel. Maintaining specialized search engines through mobile filter agents. In M. Klusch, O. Shehory, and G. Weiß, editors, *Proc. 3rd International Workshop on Cooperative Information Agents (CIA’99)*, pages 197–208, Uppsala, Sweden, 1999. Springer-Verlag: Heidelberg, Germany.
- [27] A. Tridgell and M. Pool. RSYNC: fast incremental file transfer. samba.anu.edu.au/rsync/, 2003.
- [28] A. van Hoff, J. Giannandrea, M. Hapner, S. Carter, and M. Medin. DRP - distribution and replication protocol. www.w3.org/TR/NOTE-drp, 1997.
- [29] L. Wall, P. Eggert, W. Davison, and D. MacKenzie. GNU patch. www.gnu.org/software/patch/patch.html, 2000.
- [30] E. J. Whitehead, Jr. and Y. Y. Golland. WebDAV: A network protocol for remote collaborative authoring on the web. In *Proc. of the Sixth European Conf. on Computer Supported Cooperative Work (ECSCW’99)*, Copenhagen, Denmark, September 12-16, 1999, pages 291–310.