Wikipedia Workload Analysis for Decentralized Hosting

Guido Urdaneta Guillaume Pierre
Maarten van Steen
Dept. of Computer Science, Vrije Universiteit, Amsterdam
Email: {guidou,gpierre,steen}@cs.vu.nl

Technical report IR-CS-041, Vrije Universiteit, September 2007.
Revised: June 2008.

Abstract

We study an access trace containing a sample of Wikipedia's traffic over a 108-day period aiming to identify appropriate replication and distribution strategies in a fully decentralized hosting environment. We perform a global analysis of the whole trace, and a detailed analysis of the requests directed to the English edition of Wikipedia. In our study, we classify client requests and examine aspects such as the number of read and save operations, significant load variations and requests for nonexisting pages. We conclude that differentiation is important, but that replica management may be problematic.

Keywords: Workload analysis, Wikipedia, Decentralized hosting.

1 Introduction

Despite numerous pessimistic predictions, Wikipedia is a blatant success. As of December 2007, it contains approximately 9.25 million articles in 253 languages, and is considered one of the ten most visited web sites on the Internet [1]. Its uninterrupted popularity growth has forced its operators to upgrade the hosting architecture from a single server to a distributed architecture with more than 250 servers at three locations on different continents. The current architecture, however, is still subject to scalability issues since it has centralized components such as a database for each language edition.

For Wikipedia it is particularly important to find economical ways to make their system more scalable, since its operation depends essentially on donations and the work of volunteers. There are basically three techniques that can be used to improve the scalability of any distributed system: distribution, replication and caching [15]. In contrast with the current architecture which relies on full database replication and Web page caching, we recently proposed an alternative architecture based on data distribution [20]. In this architecture, each server is responsible for hosting only a small fraction of the whole Wikipedia content. We believe that this would allow the system capacity to scale linearly with the number of hosting resources. However, the extent to which such an architecture may work in practice greatly depends on the characteristics of the workload addressed to it.

To gain a better understanding of the Wikipedia workload, we obtained and studied an access trace containing a 10% sample of the total Wikipedia traffic over a 107-day period. Studying the workload as a whole and classifying the different request types handled by the site allows us to validate assumptions on the parts of the load that are critical for overall performance. We also analyzed the workload of the English-language Wikipedia on a per-document basis. This analysis allows us to understand the properties of individual documents and the interaction between the web front end and the storage system. To the best of our knowledge, our study is unique compared to all other studies of Wikipedia's content in that it studies the actual request pattern of Wikipedia rather than just snapshots of its database.

Unlike previous workload analyses of systems such as e-commerce sites [3], P2P file sharing systems [8], static Web sites [5, 4, 2] and multimedia delivery systems [9, 19, 6], our study gives insight on the functioning of a collaborative web site, where most of the content is created and updated by external users and not the operators. In addition, we outline strategies to extend Wikipedia's collaborative model to include not only edition of content, but also hosting.

The rest of the paper is organized as follows. Section 2 gives an overview of the Wikipedia operation. Section 3 reviews related work. Section 4 describes the information available in the trace. Section 5 presents an general analysis of Wikipedia as a whole. Section 6 presents a detailed analysis of the English Wikipedia, and Section 7 serves as our conclusion.

2 Wikipedia Operation

Wikipedia is composed of a number of wikis [12]. Each wiki is typically associated with a different language edition and has a separate DNS name. For example, `en.wikipedia.org` refers to the English-language edition of Wikipedia, and `fr.wikipedia.org` to the French-language edition. In addition to Wikipedia, the Wikimedia Foundation, which is responsible for the hosting of Wikipedia, uses the same infrastructure to host other related wiki projects, such as Wiktionary (a dictionary) and WikiNews (a news site).

As is shown in Figure 1, the functionality of Wikipedia can be divided into three parts: page management, control and search. The page management part is the most important since most of the information provided by Wikipedia such as encyclopedic articles, user information, and discussions is in the form of wiki pages. Each page has a unique identifier

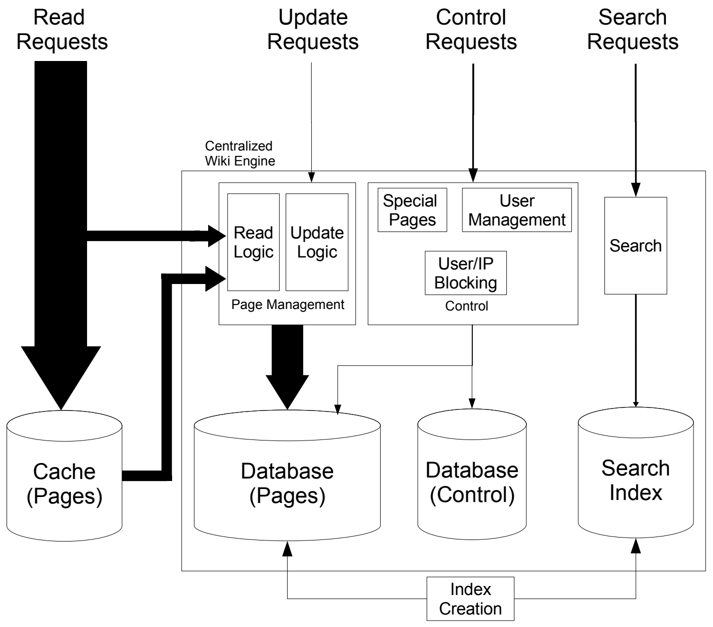


Figure 1: Wikipedia Architecture. The width of arrows roughly denotes the fraction of requests of each type.

consisting of a character string and an integer representing a name space. Pages can be created, read, and modified by any user. However, a page update does not result in the modification of an existing database record, but in the creation of a new record next to the previous version. It is therefore straightforward for a user to get a list of all editions of a page, read old versions as well as reverting a page to a previous state. Privileged users have the option to rename, delete, and protect pages from being edited. Part of the load generated by page read operations issued by anonymous (not logged-in) users is handled by a group of external cache servers, with a reported hit-rate of 78% [13].

Pages are written using a markup language called "wikitext." One important aspect of this language is that it allows for the creation of parameterized pages called templates, and the inclusion of one page into another. For example, there is a template with an information box for soccer players that takes parameters such as name, nationality and current club. This template is rarely requested by end users directly, but it is included in the articles of many soccer players and is thus frequently requested in an indirect way. A page can also be configured to redirect all its read requests to another page, similarly to a symbolic link. Redirection is implemented by rendering the content of the target page when the master page is requested. One consequence of these features is that there is not a one-to-one correspondence between the HTML pages that users typically read and the wikitext pages stored in the database.

The search part allows users to enter keywords and receive lists of links to related wiki pages as a result. This part of the system is isolated from the rest of the application in that it does not access the centralized database, but instead accesses a separate index file generated periodically from the text of the pages.

Finally, the control part groups the rest of the functionalities. It encompasses features such as user management, which allows users to authenticate to the system and have their user names stored in public page history logs instead of their IP addresses; user/IP address blocking, which allows administrators to prevent page updates from certain IP addresses or user accounts; and special pages, which are not created by users, but generated by the execution of server-side logic and provide information about the database or specific functions such as uploading static files to be referenced in wiki pages.

3 Related Work

Many previous studies have characterized different types types of web workloads. However, to the best of our knowledge our study is the first one to study the workload of a major collaborative Website such as Wikipedia. Almeida [2] analyzes the workload of four web sites and studies temporal and spatial locality. He shows that temporal locality can be characterized using the stack distance metric, while spatial locality can be modeled using the notion of self-similarity. Arlitt and Williamson [4] study the workload of six web sites and identify a number of invariants that apply to all the studied data sets such as lookup success rate, mean transfer size, size distribution, inter-reference times, and concentration of reference, among others. Bent et al. [5] study the properties of a large number of web sites hosted by a major ISP. They find that the workload contains a high degree of uncacheable requests, related to the widespread use of cookies; that most sites do not use HTTP 1.1 cache-control features, and that most sites would benefit from the use of a content delivery network. Arlitt et al. [3] analyze a five-day workload from a large e-commerce site. They characterize user requests and sessions and determine their impact on scalability. They find that horizontal scalability is not always an adequate mechanism for scaling the system, but that system-level and application-level QoS mechanisms are required in overload conditions. Note that these workloads differ from Wikipedia's in that almost all Wikipedia content is dynamic and updated by end-users.

Various analyses have been conducted to study Wikipedia from publically available database dumps. Voß [21] studied four language editions of Wikipedia and measured numerous statistics such as size distribution of articles, number of distinct authors per article, number of articles per author, distribution of links among pages, and growth in several variables such as database size and number of articles. Ortega et al. [16] analyzed user contributions in the English Wikipedia and found that a small percentage of authors are responsible for most of the contributions. Wilkinson and Huberman [22] studied the relationship between quality and cooperation in Wikipedia articles and found that high-quality articles (denoted as "featured" in Wikipedia) are distinguished from the rest by a larger number of edits and distinct editors, following a pattern where edits begat edits. Hu et al. [10] propose three models for automatically deriving article quality in Wikipedia. The models are based on interaction data between articles and their contributors. While these studies are important to understand the update behavior of Wikipedia users, one cannot ignore the fact that most Wikipedia users simply read the encyclopedia without editing it. Our study differs from these in that we use a unique trace that contains a sample of the full Wikipedia traffic including read requests in addition to the information available in database dumps.

Decentralized collaborative designs for Wikipedia or wikis in general have also been proposed. Morris proposes a distributed wiki based on the JXTA P2P framework [14]. This design drops the web interface in favor of a specialized client that allows every user to store redundant copies of wiki pages. We recently proposed a decentralized architecture for collaborative Wikipedia hosting [20]. In this design, pages are distributed over a network of collaborators that contribute their computing and networking resources to help host Wikipedia. A load balancing algorithm tries to place pages on the most appropriate nodes and a distributed hash table is used to locate pages by name. Regular Wikipedia users use a normal web browser and do not necessarily participate in the collaborative hosting system. Vikram et al. [18] propose a new model for Wikipedia hosting and moderation. In this system, Wikipedia hosting is distributed over a variety of sites that include full hosting sites holding a full replica of the Wikipedia content, subset hosting sites that host articles relevant to a particular community, niche hosting sites that host articles outside the realm of the current Wikipedia, and archiving sites that store full replicas for archival purposes. The system proposes a hierarchical article moderation architecture where updates are quarantined in a local hosting site until a moderator approves further propagation to other sites. We however observe that the potential performance of Wikipedia collaborative hosting largely depends on the traffic that should be handled. The purpose of the present article is precisely to highlight the important characteristics of Wikipedia's traffic from the point of view of collaborative content distribution infrastructures.

4 Wikipedia Traces

To conduct our study of the Wikipedia workload, we were provided by the Wikimedia Foundation with a sample of 10% of all requests directed to all the wiki projects they operate. The sample used in our study was generated by Wikipedia's front-end proxy caches, and contains 20.6 billion HTTP requests corresponding to the period from September 19th, 2007 to January 2nd, 2008. Each request in our trace is characterized by a unique ID, a timestamp, the requested URL, and a field that indicates if the request resulted in a save operation. Each Wikipedia request in the studied period has a 10% probability of being included in our trace. As a consequence, no bias is introduced at the request level. However, aggregate information related to specific pages may be inaccurate due to the sampling and user actions such as page removal and creation. We quantify the impact of this innaccuracy in Section 6, where we analyze the English Wikipedia at the page level.

For privacy reasons, the trace given to us by the Wikimedia Foundation does not contain any direct or indirect means to identify users, such as client IP address or session cookie. Our study therefore focuses on server-side information. The only client-side data available is update information present in public snapshots of the Wikipedia database. For some of our analyses, we used a snapshot of the English Wikipedia database, dated January 3rd, 2008 [7].

5 Global Analysis

From the URL included in the trace it is possible in most cases to determine the targeted wiki project and the type of operation issued by the client. Table 1 shows the different types of requests addressed to Wikipedia, and their relative frequency.

We can see that most of the traffic is generated by the action of end users issuing read operations to wiki pages. Since it is common for pages to reference multiple uploaded images and static files, these two types of requests account for more than 64% of all requests. We can also see that page editions (at 0.03%) are very infrequent compared to page reads, and image uploads are even less frequent (0.002%). It is thus clear that a high degree of caching or replication can be used to improve performance. It should also be noted that a nontrivial number of page requests are for formats different from the default HTML, which suggests that in some cases replicating the wikitext instead of, or in addition to the final HTML, would produce further performance improvements.

Not all wikis in Wikipedia are equally popular or equally used. Table 2 shows the distribution of request load by the wiki projects as well as the ratio of HTML read requests to save requests. Although the trace has references to more than 800 wikis with more than 2000 requests in our sample (many of them nonexisting), almost half of the total traffic is directed to the English Wikipedia, and about 90% of the traffic is concentrated in the 10 most popular wikis. This shows that a strategy of using a separate database for each wiki cannot efficiently solve the scalability issues since there is a large imbalance in the load. This also justifies a more comprehensive study of the English Wikipedia in order to

Table 1: Wikipedia request types, and their frequencies expressed in fractions of the total number of requests.

Description	Frequency
Requests for the current version of a wiki page using the default HTML rendering.	13.15%
Requests for the current version of a wiki page using a different format, such as printer-friendly versions or raw wikitext.	8.50%
Requests that result in a page update or creation.	0.03%
Requests for the edition history of a page.	0.06%
Requests for a specific version of a page. It may be a diff operation, in which case two versions are compared.	0.06%
Requests in which a page name is specified, but the information retrieved is independent from the page's wikitext. For example, Wikipedia allows obtaining the Javascript or CSS used in the rendering of a specified page without obtaining the page itself.	4.49%
Requests for user-uploaded binary files, typically images.	21.88%
Requests for thumbnails of user-uploaded images.	18.70%
Requests for possible uploads of binary files. They can refer to new files, updates of existing files or retrieval of a form that allows the user to send an actual upload request. It is impossible to determine the name of the uploaded file from our traces nor if it is an actual upload.	0.002%
Keyword search requests handled by the search component of the wiki engine.	0.81%
Requests related to cache maintenance.	5.18%
Requests for special pages other than full text search or upload. Some of these special pages result in the retrieval of wiki pages. However, the names of the wiki pages involved cannot be obtained from the trace.	0.88%
Requests for static files. These are usually files used in the rendering of wiki pages, such as CSS and Javascript files as well as generic images such as bullets.	24.04%
Requests directed to a web service API. Most of these requests result in the retrieval of wiki pages, but in many cases it is not possible to determine the names of the pages involved.	0.14%
Requests for a static HTML version of Wikipedia that is available as an alternative to the normal one. The static version is updated periodically but is never guaranteed to have the most current version of a page.	0.01%
Requests that produce search results in a standardized format known as OpenSearch.	1.31%
Other types of request.	0.758%

Table 2: Distribution of load and read/save ratios across different wiki projects

Wiki	Frequency	HTML Read/ Save Ratio
English Wikipedia	45.05%	480.0
Wikipedia Commons	14.33%	N/A
German Wikipedia	7.07%	504.7
Japanese Wikipedia	6.19%	1081.2
Spanish Wikipedia	4.87%	458.8
French Wikipedia	3.42%	228.1
Mathematical formulas	2.45%	N/A
Italian Wikipedia	2.03%	216.2
Portuguese Wikipedia	1.84%	346.5
Polish Wikipedia	1.70%	363.5
Dutch Wikipedia	1.1%	258.7
Others ($<$ 1% each)	9.95%	Unknown

gain a deeper understanding of the issues that affect global Wikipedia performance. We can also see that the read/save ratio varies significantly for the different language editions of Wikipedia. This shows that factors such as culture, size, and geographical distribution of the user base influence the workload and thus have an effect on how strategies should be selected to improve performance.

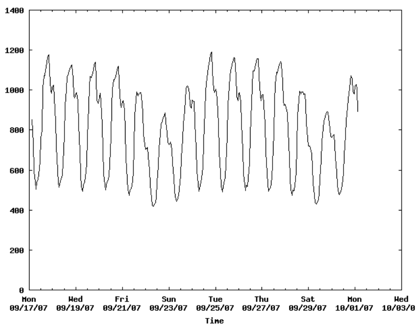
Our next analysis examines the usage of the most popular wikis. Figure 2 shows the request rate for the four most popular Wikipedias during a two-week period. The workload follows typical time-of-day and day-of-week patterns. However, the load variations differ for each wiki. For example, within a single day the request rate is expected to change by a factor of about 2.3 in the English Wikipedia and by a factor that can be as high as 19 in the German Wikipedia. On the other hand, we did not observe any flash crowds that may affect the normal daily behavior.

6 English Wikipedia

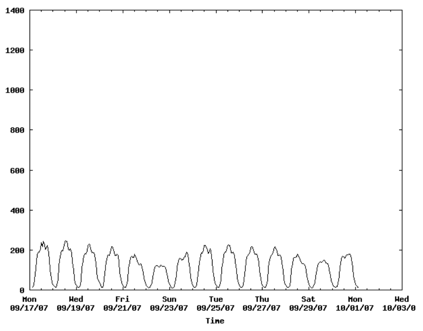
We now focus on the English edition of Wikipedia to conduct a more in-depth workload analysis. The data we consider in this section includes all the requests in the trace directed to a wiki page in the English Wikipedia. In this study requests for uploaded images, special pages, static files, or other types of objects are not included. Requests to pages that specify an invalid page name are included and analyzed in Section 6.4.

There are two main reasons why we focus on wiki pages. First, they can be updated by ordinary users at any time, so they introduce a nontrivial consistency problem in a decentralized replicated scenario. Second, they are directly or indirectly responsible for the vast majority of the Wikipedia traffic. As we have seen in Table 1, static files and uploaded images are requested more frequently than pages, but this is explained by the fact that wiki pages often reference several images and static files. Static files are rarely updated, if ever, so they do not represent a special difficulty in a decentralized environment. Uploaded binary files can be updated by users, but in practice this occurs very rarely, so they can in general be regarded as static files or as read-only wiki pages.

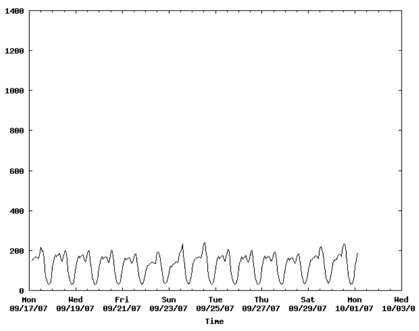
In this section, unless explicitly stated, when we refer to the total number of pages, we mean all the requested page names, including invalid ones. When we refer to existing pages, we refer to pages that were registered as existing in the database at some point during the studied period. This includes pages that were deleted or renamed. When we denote a number as coming from the database (e.g. number of updates in the database), we mean that we are using data from the database snapshot and not from the trace.



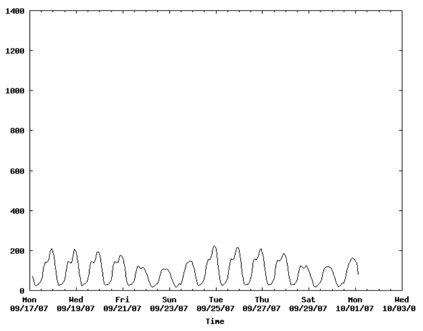
(a) English



(b) German



(c) Japanese



(d) Spanish

Figure 2: Usage of the four most popular Wikipedias over a two-week period.

6.1 Trace Validation

Our first analysis studies the validity of our trace. We claim that the trace is unbiased at the request level, since all requests have the same probability to appear in the trace. However, when requests are aggregated to produce information about specific pages, the sampling and certain events during the study period may produce inaccuracies. For example, low traffic pages may not be represented at all in the trace, while pages deleted during the study period may appear as having a certain popularity that no longer applies.

Fortunately, the trace is not our only source of information. We also use a publicly available snapshot of the English Wikipedia database with accurate information about events such as page creation, deletion and renaming, as well as the full history of updates for all the valid pages at the time the snapshot was created[1]. Since the snapshot was taken just one day after the end of the study period, the information we derive from it is very accurate with respect to the trace.

Table 3 summarizes the classes of pages that may deserve special attention, and their importance in terms of the fraction of pages they represent with respect to the total number of existing pages, and the fraction of requests they represent with respect to the total number of page requests in the trace. It should be noted that we identify pages by their name, so our page-level analyses are actually at the page-name level.

We can see that the impact of these events on our page-level analyses is limited. Note that, despite their specificities, these pages do provide accurate information for many of our analyses. Our analyses measure some variables that characterize the set of all pages, and allow us to classify them. The most important variables we study at the page level are the distribution of popularity in terms of number of requests and number of save operations, the format in which pages are read, and the ratio between save and read operations.

[1]The snapshot does not include the update history for deleted pages.

Table 3: Pages with events during the study period that may influence the accuracy of results

Category	Fraction of pages	Fraction of Requests
Pages that were created during the study period. The popularity rank derived from the number of requests during the study period cannot be used as an estimate of the future popularity rank for these pages.	10.94%	0.87%
Pages that were deleted during the study period. Popularity rank in the studied period cannot be used to predict future popularity, which is expected to be very low or null.	5.03%	2.98%
Pages that were renamed during the study period. In this case, we consider the page as two different pages.	0.62%	0.52%
Pages that were renamed during the study period, with a new redirect page created with the old name. In this case, we consider the old page name a single page, and the new name as a new page. This is inaccurate, because the history of save operations stays with the new name, and the nature of the load for the old name may change significantly. For example, if a frequently updated page is renamed with a redirect, the new redirect page will probably have fewer updates. The result is that the save/read ratio for the old name will be calculated using two different workloads.	0.11% (old name) 0.15% (new name)	0.21% (old name) 0.52% (new name)
Total	16.85%	5.1%

Deleted and newly created pages present no problem regarding any of these variables apart from the fact that individual page popularity rankings cannot be used as a predictor for future popularity.

Renamed pages represent a negligible portion of the dataset. However, the little information they provide can still be useful. First, they contribute to correctly determine a popularity distribution of page names, which is relevant in a decentralized setting, where a distributed lookup service is needed. Second, what could be perceived as inaccuracies in the determination of save/read ratios can be considered as part of the load variations that pages are subject to. For example, renaming a page with a redirect is not very different from simply modifying it to redirect requests to another page.

Another potential source of inaccuracies is the sampling. Pages with very low traffic may be misrepresented, mainly because they may not be reported at all in the trace, but also because they may be overrepresented due to chance. We can get an indication of how representative the sample is by comparing the save operations reported in the trace, with the save operations stored in the database snapshot.

Figure 3 shows the correlation between the number of save operations per page reported in the trace and the save operations registered in the database snapshot during the studied period. Each point in the graph represents a page, the thick line represents the median number of saves in the database for a given number of save requests in the trace, and the bars represent the 10th and 90th percentiles. The correlation coefficient is 0.81. As expected, the median is a good approximation to a line with slope 10. We can see that the variability is reduced as the number of saves increases. Since read requests are considerably more frequent than save requests, we can expect the sampling to be even more accurate in that case. We conclude that the sample is representative of the real traffic for virtually all pages, and that only pages whose influence on the load is negligible may have a significant probability of being misrepresented.

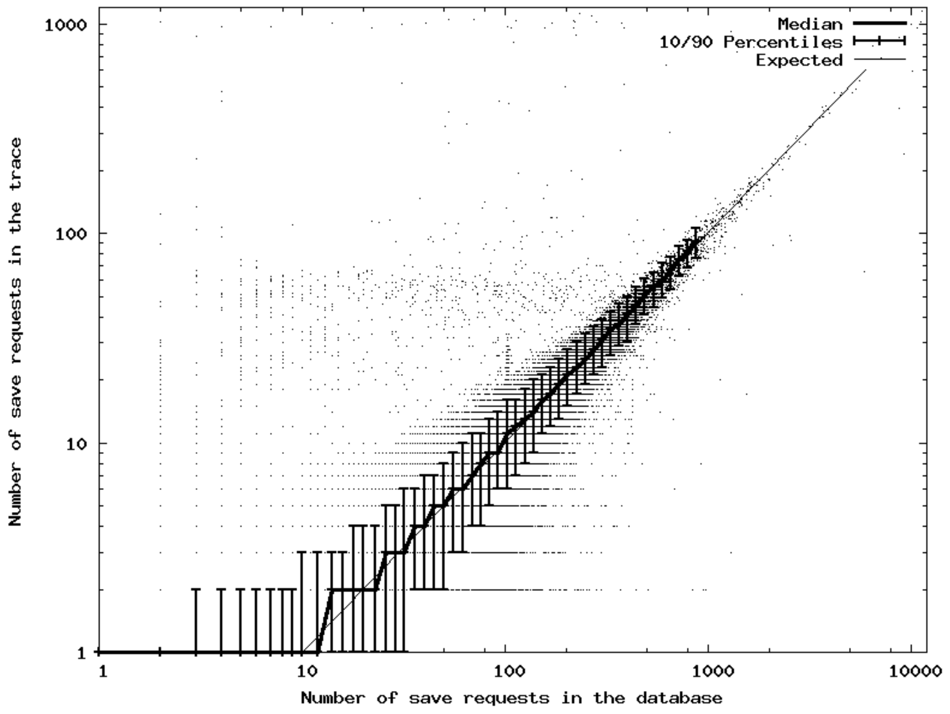


Figure 3: Correlation between the number of save operations per page reported in the trace and in the database snapshot

6.2 Page Popularity

Figure 4 shows the popularity distribution of all referenced pages in the trace. Pages are ordered by decreasing number of requests in the studied period. Unlike other types of websites, where the page popularity distribution closely follows a Zipf distribution [3, 4, 11], the popularity distribution of Wikipedia pages can be described as having three zones. First, we see that the four most popular pages show a popularity orders of magnitude higher than any other page. This is an artifact of the structure of Wikipedia pages: three of these pages contain CSS or Javascript code included in many other HTML-rendered Wikipedia pages and the other is the main page, which is the most popular HTML-rendered page for obvious reasons. A second part is constituted by approximately the next 20,000 pages in popularity, which roughly follow a Zipf distribution (with an estimated $\beta = 0.53$), indicated in the graph. A third part is constituted by the remaining 28.7 million pages, which deviate from the model by having lower frequencies than predicted by the Zipf distribution. The more rapid decline in pages with 50 or less requests might be due to the sampling, as we saw with pages with few save requests in Figure 3.

In Wikipedia, most requests performed by ordinary users are read operations that normally result in a default HTML rendering of the requested page. Wikipedia implements caches for these HTML renderings to reduce the centralized database load. However, a page can be rendered in many different formats. Figure 5 shows the correlation between the number of reads per page in any format, and reads that result in the default HTML rendering. Each point represents a page, and pages where all reads are in the default HTML format appear on the diagonal. For readability reasons the graph shows a random sample of 10% of all pages. It can be seen that for a significant number of pages, the number of read operations in a non-default format is considerable. More specifically, 8.1% of the existing pages with at least one read operation in the trace have more than 25% of their read

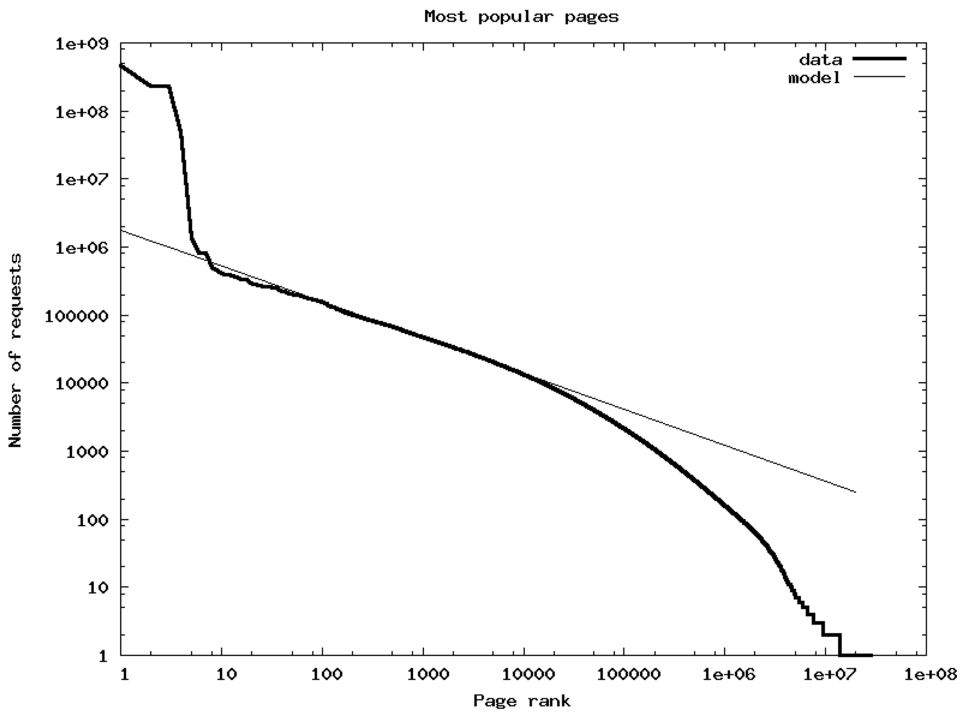


Figure 4: Page popularity distribution

operations in a non-default format. For these pages it would be useful to replicate or cache the wikitext since it can be used to generate all possible formats.

Save Operations

Now we turn our attention to save operations. Figure 6 shows pages ranked by the number of save operations in the trace. We can see that the popularity of save operations approximately follows a Zipf distribution where we have computed β to be 0.64.

We note that 44% of the existing pages have at least one real (not sampled) save operation during the studied period, and they represent 91% of all page requests in the trace. This forces us to make a distinction between read-only pages, which are easy to host, and updated pages, which are more difficult to host in a scalable way and represent the vast majority of page requests. Moreover, pages with only a handful of updates in the study period can be considered unmaintained, and treated similarly to read-only pages.

Figure 7 shows the correlation between the number of read and save operations. Each point represents a page. All pages with at least one read and one save operation in the trace are represented in the graph. The line represents the median number of save operations for pages with a given read popularity. We observe that read and save operations per page are correlated, so popular pages are more likely to be frequently updated than unpopular pages. Moreover, we can see that for pages with at least 1000 read requests in the trace, the median read/save ratio is essentially constant and approximately equal to 1000, with a slight tendency to grow for the most popular pages. The variability is significant and there are many exceptions, but the order of magnitude of the variability is also constant.

These results suggest that less popular pages would benefit more from distribution than replication, and that replication is, in principle, a good strategy for the more popular pages. Moreover, the essentially constant median read/save ratio could be used to establish initial

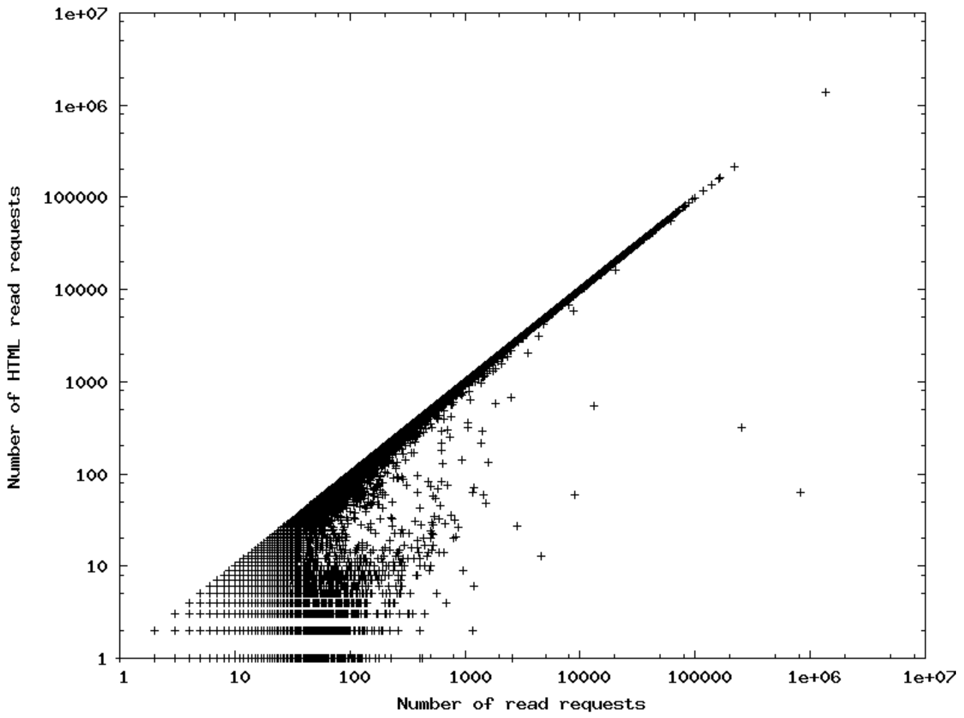


Figure 5: Correlation between the number of all reads per page and page reads in the default HTML format

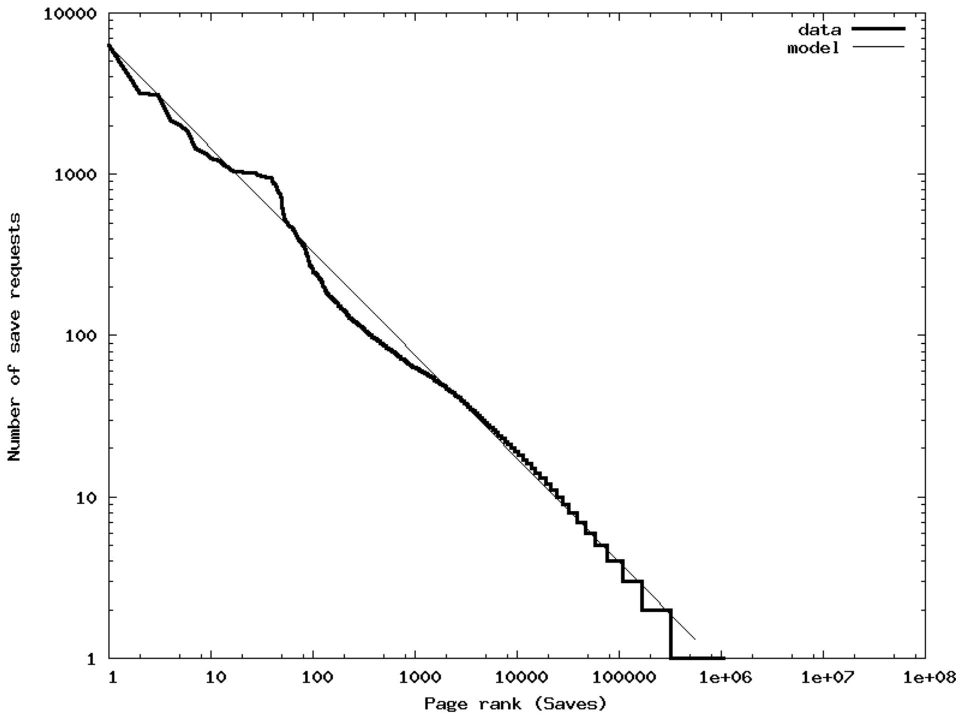


Figure 6: Distribution of page save operations in the trace

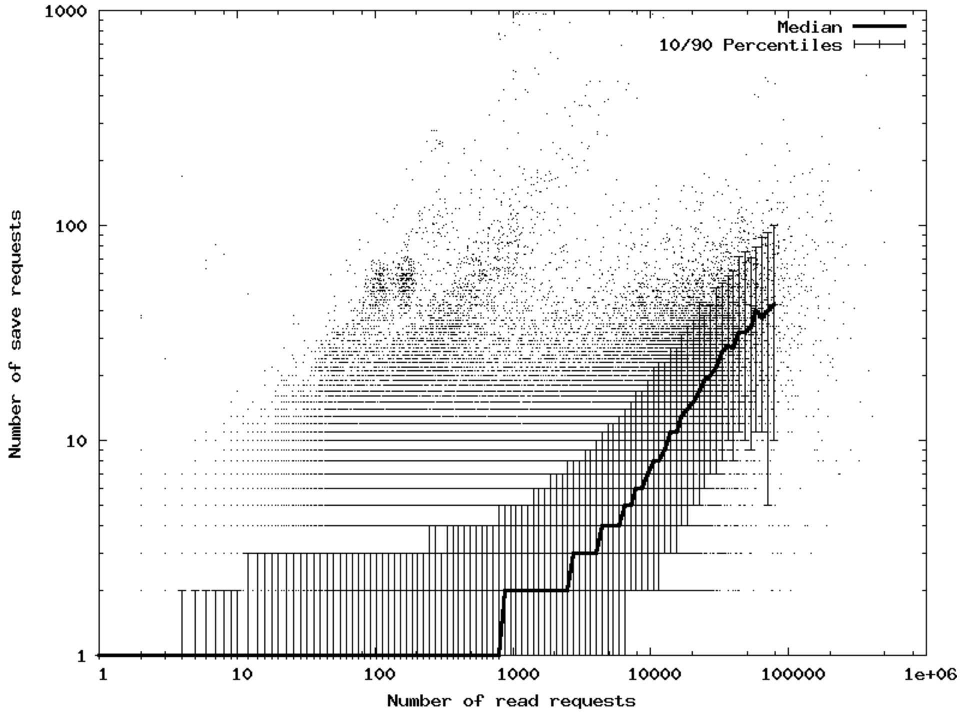


Figure 7: Correlation between the numbers of page read and save operations

Table 4: Number of increases and decreases in daily number of requests per page by a factor of at least 10

Request Type	Number of events	Number of Pages
Increases	14404	13649
Decreases	8177	7847

default policies for newly created pages. However, as we will see in Section 6.5, these considerations do not necessarily apply to replication of HTML-rendered pages.

6.3 Load Variations

As we saw previously, Wikipedia as a whole does not seem to exhibit significant load variations apart from normal time-of-day and day-of-week patterns. However, individual pages may present spikes of popularity. Such events can potentially have an important impact on a decentralized hosting infrastructure, since each server may host only a handful of documents and thus be affected by such local variation.

We analyzed occurrences where the number of requests that a page receives in a whole day represents at least a 10-fold increase or decrease with respect to the previous day. In a decentralized system, such pages would most probably need to adjust their replication factor or migrate to more appropriate servers. We ignored cases where the daily number of requests is less than 100 in our trace for both the previous and actual day of the event. Table 4 shows the number of significant load variations that we observed.

As we can see, significant variations of the popularity of pages are relatively common, as we noticed on average 133.6 increases per day (14404 increases in 108 days). Furthermore, it was uncommon for a page to experience multiple events of this type in the studied period.

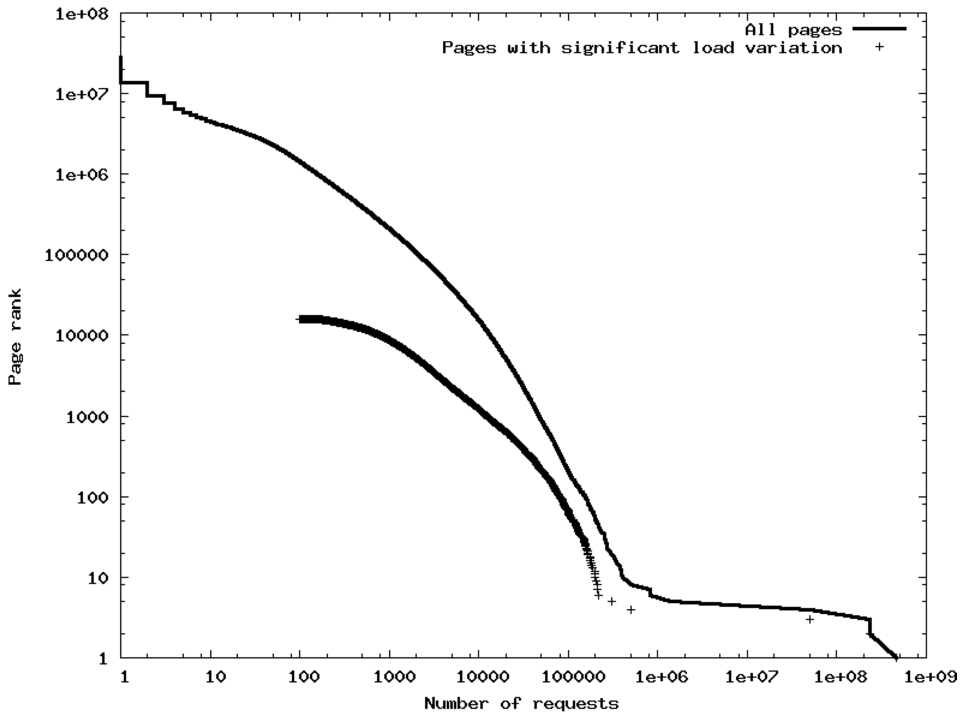


Figure 8: Distribution of load variations vs. page popularity

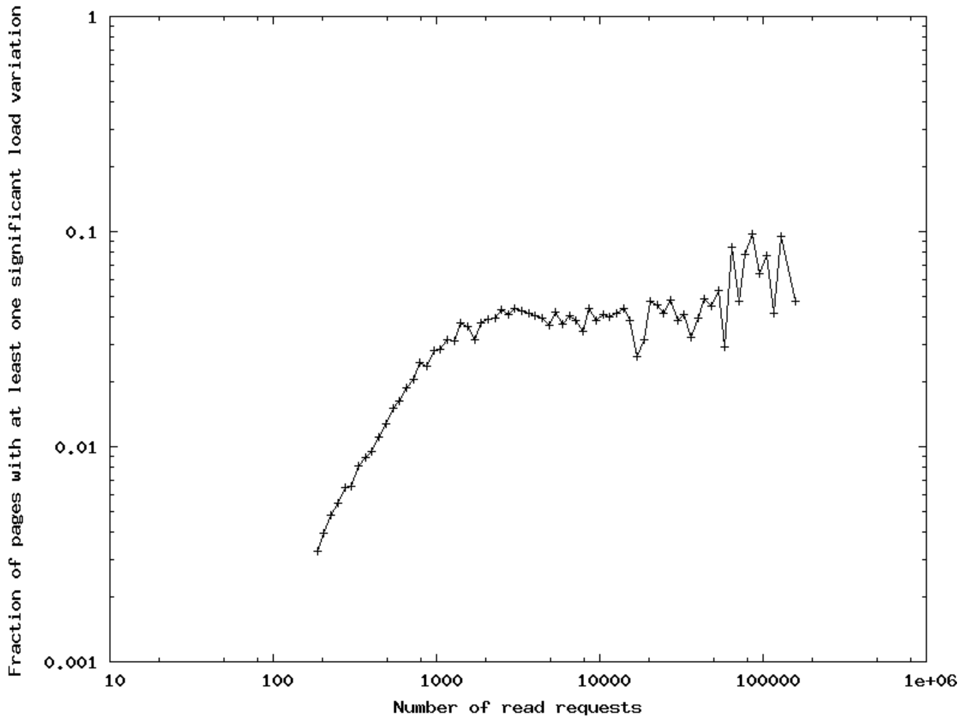


Figure 9: Fraction of pages with load variations ranked by popularity

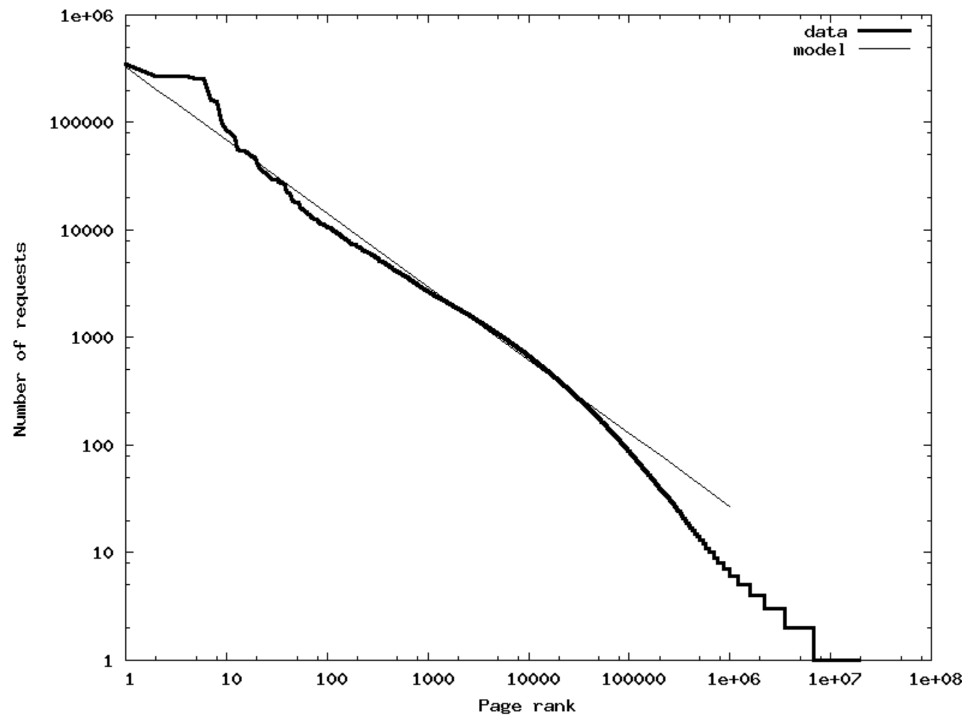


Figure 10: “Popularity” distribution of nonexisting pages.

Figures 8 and 9 relate page-level load variations to popularity. In Figure 8 the top curve shows the number of page read requests for all pages ranked by popularity, while the bottom curve ranks only pages that exhibit at least one significant load variation event in its daily request rate. We see that pages with a wide range of popularity can experience significant load variations. Figure 9 shows the fraction of pages with load variations given the popularity. We can see that the fraction of pages that experience load variations is essentially independent of the initial page popularity. Therefore, current popularity cannot be used to predict future load variations.

6.4 Nonexisting Pages

About 3.5% of page requests are addressed to pages that do not exist in the database. Such requests may result from simple typing mistakes by end users. This could, however, be problematic when using a decentralized hosting infrastructure, as each request may potentially cause a distributed lookup operation.

Figure 10 shows the “popularity” of requested nonexisting pages. This popularity again approximately follows a Zipf distribution (with an estimated $\beta = 0.68$).

The effects of requests to nonexisting pages can therefore be largely reduced by using negative caching techniques where front-end servers cache the information that a page does not exist.

We however also note that certain nonexisting pages are requested too frequently to be attributed to typing mistakes alone. One of the most requested nonexisting page names is the following (truncated to 255 characters during the analysis):

```
    Skins-1.5/common/skins-1.5/common/skins-1.5/common/skins-1.5/common/
skins-1.5/common/skins-1.5/common/skins-1.5/common/skins-1.5/common/
skins-1.5/common/skins-1.5/common/skins-1.5/common/skins-1.5/common/
skins-1.5/common/skins-1.5/common/skins-1.5/common/
```

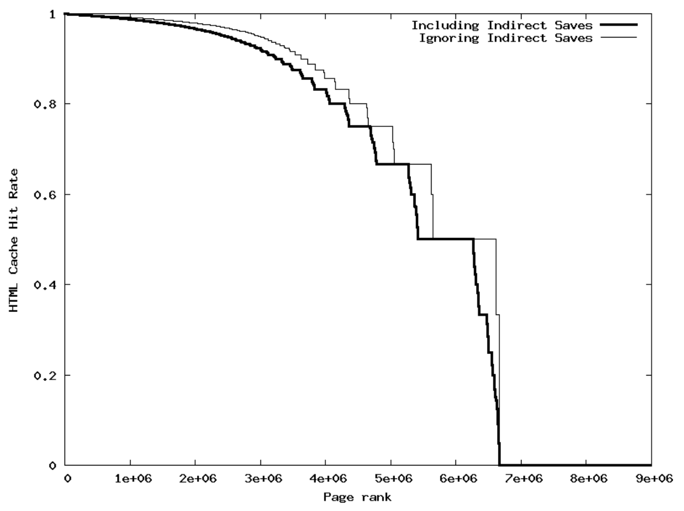


Figure 11: Distribution of HTML cache hit rate

This type of page name is clearly not produced by a normal user. It is possible that requests of this type are generated by faulty processes or malicious users trying to disrupt the normal operation of Wikipedia.

6.5 Indirect Save Operations

Our next analysis concerns the effect of pages that are included in other pages. As we explained above, Wikipedia allows one page to include the content of another page with the inclusion and redirection features. One important implication of these features is that save operations to included or redirected pages affect the consistency of replicated or cached HTML renderings of their master pages. This means that an otherwise read-only page can become a frequently updated page if it includes a frequently updated template page. For example, in an extreme case, the user discussion page "User_talk:Hu12" has a read/save ratio of 8.5 if we consider only direct save operations. However, if we take into account indirect save operations, this ratio drops to just 0.14, meaning that it is updated more often than read.

We try to determine the impact of indirect update operations on the ability of pages to be cached or replicated. To do this, we examine the inclusion and redirection maps available in the database snapshot and determine the indirect number of saves on the master pages. This method is not fully accurate, since the inclusion and redirection maps can change as the result of save operations. However, inclusion and redirection maps are not expected to change significantly during a 3.5-month period.

To quantify the impact of indirect saves, we compute the cache hit rate of an infinite size cache for HTML versions of pages. The cache hit rate is defined as the quotient between the number of requests that successfully read the cache versus the number of requests that either use the cache successfully or cause it to be reloaded. In our case, we use the expression $\frac{H-1}{H+D+I}$, where H is the number of read operations in HTML format, D is the number of direct updates and I the number of indirect updates.

Figure 11 shows pages ranked by their HTML cache hit rate both taking and without taking into account the effect of indirect saves. We see that if indirect saves are ignored, approximately 41% of pages have an HTML cache hit rate of 90% or more. Indirect updates make this fraction drop to approximately 37%.

Our conclusion is that indirect save operations can have an impact on caching and replication of master pages. Thus, it is important to minimize this impact by implementing policies such as protecting popular templates from being updated by unprivileged users, and by adopting separate policies for replicating wikitext and rendered versions of pages.

6.6 Old Page Versions

From the point of view of data storage, a page consists of the current version of its content, the content of all previous versions, and some metadata such as update restrictions. From the database snapshot, we estimate that old versions account for approximately 95% of the total storage space. However, they are rarely accessed, as shown by Table 1, and are read-only by definition.

This definition of a page is not convenient for load balancing, since the cost of moving a page would increase every time the page receives an update, to the point that after a certain number of updates, load balancing would stop being a viable solution.

A possible solution to this problem would be to assume that old versions are separate documents from the current version of the page, and that every time a page is updated, a new unpopular read-only document is introduced into the system. Thus, old versions of pages are easy to host on computers with low bandwidth and abundant disk space, such as ADSL-connected personal computers.

7 Conclusions

Our study of the Wikipedia workload provides important insights relevant for hosting Wikipedia in a decentralized and collaborative environment. We have centered our detailed analysis on a number of essential page characteristics: frequency of requests, the format in which pages are read, the frequency of direct save operations, the frequency of indirect save operations, and relevant ratios between these variables. These variables together should determine the policies for distributing and replicating pages, which includes both the original wikitext source format and rendered HTML formats.

There are several factors that make it difficult to automatically determine policies. First, one must decide whether to replicate pages, or to try to place them on the most appropriate nodes. Second, one may need to select separate policies for the wikitext and rendered versions of a page, since direct save operations affect the consistency of both versions, while indirect updates affect only rendered versions. Third, all variables should be considered in combination to decide the policies, which may result in difficult tradeoffs. For example, an unmaintained page that is very popular in HTML mode and receives many indirect updates is in the situation where it should benefit from HTML replication, but this replication may introduce scalability problems due to the high indirect update rate. At the same time, the wikitext source could be easily replicated, and this could be used to generate the HTML, but at a significant cost in terms of performance with respect to a previously rendered replica.

In addition, a decentralized system must be ready to take emergency action under unexpected load variations on specific pages that may result from real-world events external to the system, and should efficiently handle invalid requests, including some that might try to disrupt the normal operation of the system.

In Table 5 we attempt to classify Wikipedia pages according to their workload, and give some guidelines for setting policies for each type of page. We classify pages according to three metrics: HTML cache hit rate, direct save/read ratio, and fraction of reads in HTML format. Together with page popularity, we believe that these metrics are the most relevant to select appropriate hosting policies at the page level. For simplicity, we classify pages as having a 'high' or 'low' values for each metric. Although the cutoff values are chosen rather arbitrarily, we believe our classification is sufficient for providing general guidelines.

We can distinguish two important sets of pages. The most important is by far the set of cacheable pages, which represent 43.1% of all pages and 95.7% of requests. These pages

Table 5: Wikipedia page characterization according to the workload. The cutoff values we use are: 85% or more is considered a high cache hit ratio, 15% or more is a high save/read ratio, and 75% or more reads in HTML is a high HTML fraction. Popularity is used to determine the strength of the policies.

HTML Cache Hit Rate	S/R Ratio	HTML fraction	%Pages	% Requests	Commentary
Low	Low	Low	6.5%	0.2%	Replication of wikitext should be the preferred policy for these pages. Popularity should determine the replication degree.
Low	Low	High	47.3%	0.8%	This type of page benefits mainly from load balancing of HTML replicas. Popularity should determine how agressive the system should try to find appropriate nodes for these pages.
Low	High	Low	1.5%	0.0%	These pages benefit mainly from load balancing. Replication of wikitext should be only for fault tolerance, and HTML replicas do should not be used. Popularity should indicate how agressively the system should try to find appropriate nodes for these pages.
Low	High	High	1.7%	0.0%	These pages benefit mainly from load balancing. Replication should be only for fault tolerance. HTML caches could be colocated with the wikitext replicas to make consistency easier. Popularity should indicate how agressively the system should try to find appropriate nodes.
High	Low	Low	0.3%	20.2%	These pages benefit from a high degree of replication of wikitext or alternate formats. There may be little need for HTML replicas, but they may exist and be colocated with wikitext replicas to make consistency easier. Popularity should determine the replication degree.
High	Low	High	42.8%	75.2%	These pages benefit from a high degree of HTML caching. Popularity should determine the replication degree for rendered replicas. Wikitext replication may be used, but mostly for fault tolerance.

are relatively easy to host since they can be easily replicated in the appropriate format. Popularity can be used to determine the appropriate replication degree. The second important group of pages consists of pages for which caching makes no sense due to a low number of read operations with respect to the number of updates. In these cases, the system should rely on a load balancing algorithm to place each page in the most appropriate node, with replication only for fault tolerance. These pages represent almost half of all pages, but only a small fraction of the requests. Therefore, they are good candidates for hosting in personal computers with ADSL connections.

Determining how much to rely on replication and how much on load balancing is complicated by the fact that the studied variables can take many more than just two values. A decentralized solution should solve this kind of tradeoff and automatically determine the appropriate parameters for each page. Classifying pages and determining per-page strategies has already been shown to be a solvable, yet nontrivial problem [17].

A fully decentralized solution must also satisfy a number of functional requirements such as efficiently detecting if a page exists or not, and implementing relationships among pages such as categories, which allow encyclopedic articles to specify topics they cover, and edition protection, which allows an administrator to protect a page and its included pages from being updated. In addition, it should deal with extra-functional issues such as security and privacy.

The problem of detecting if a page exists or not, given its name, is crucial for two reasons. First, Wikipedia functionality requires links to nonexisting pages to be rendered in a different color than links to valid pages. Second, this is, with little modification, the same problem as locating the node where a page resides in order to forward requests to it.

The problem of implementing relationships among pages in a decentralized way is complicated by the fact that the relationships must be kept in a consistent state in the presence of updates and partial failures. Solving this problem in a decentralized environment is similar to implementing consistency for replicas, which is a problem that must be solved both to improve performance and to achieve fault tolerance.

However, the most difficult challenge faced by a decentralized and collaborative system for hosting Wikipedia is solving all the aforementioned problems in an environment where mutually untrusted parties participate, while at the same time guaranteeing fair resource usage for participants, and privacy for regular Wikipedia users who have nothing to do with the hosting of the system.

Acknowledgments

We wish to thank the Wikimedia Foundation, especially Gerard Meijssen and Tim Starling, for making their access trace available for our study.

References

[1] Alexa Internet. Alexa web search - top 500, 2007. `http://www.alexa.com/site/ds/top_sites?ts_mode=global`.

[2] Virgílio Almeida, Azer Bestavros, Mark Crovella, and Adriana de Oliveira. Characterizing reference locality in the WWW. In *Proceedings of the IEEE Conference on Parallel and Distributed Information Systems (PDIS)*, Miami Beach, FL, 1996.

[3] Martin F. Arlitt, Diwakar Krishnamurthy, and Jerry Rolia. Characterizing the Scalability of a Large Web-Based Shopping System. *ACM Transactions on Internet Technology*, 1(1):44–69, 2001.

[4] Martin F. Arlitt and Carey L. Williamson. Web Server Workload Characterization: the Search for Invariants. In *SIGMETRICS '96: Proceedings of the 1996 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 126–137, New York, NY, USA, 1996. ACM Press.

[5] L. Bent, M. Rabinovich, G. M. Voelker, and Z. Xiao. Characterization of a Large Web Site Population with Implications for Content Delivery. In *WWW '04: Proceedings of the 13th International Conference on World Wide Web*, pages 522–533, New York, NY, USA, 2004. ACM Press.

[6] Ludmila Cherkasova and Minaxi Gupta. Analysis of Enterprise Media Server Workloads: Access Patterns, Locality, Content Evolution, and Rates of Change. *IEEE/ACM Transactions on Networking*, 12(5):781–794, 2004.

[7] Wikimedia Foundation. Wikimedia dump service, 2007. `http://download.wikimedia.org/`.

[8] Krishna P. Gummadi, Richard J. Dunn, Stefan Saroiu, Steven D. Gribble, Henry M. Levy, and John Zahorjan. Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload. In *SOSP '03: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, pages 314–329, New York, NY, USA, 2003. ACM Press.

[9] Lei Guo, Songqing Chen, Zhen Xiao, and Xiaodong Zhang. Analysis of Multimedia Workloads with Implications for Internet Streaming. In *WWW '05: Proceedings of the 14th International Conference on World Wide Web*.

[10] Meiqun Hu, Ee-Peng Lim, Aixin Sun, Hady Wirawan Lauw, and Ba-Quy Vuong. Measuring Article Quality in Wikipedia: Models and Evaluation. In *CIKM '07: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, pages 243–252, New York, NY, USA, 2007. ACM.

[11] Frank T. Johnsen, Trude Hafsoe, and Carsten Griwodz. Analysis of Server Workload and Client Interactions in a News-on-Demand Streaming System. In *ISM '06: Proceedings of the Eighth IEEE International Symposium on Multimedia*, pages 724–727, Washington, DC, USA, 2006. IEEE Computer Society.

[12] Bo Leuf and Ward Cunningham. *The Wiki Way: Collaboration and Sharing on the Internet*. Addison-Wesley Professional, April 2001.

[13] Wikimedia Meta-Wiki. Cache strategy, 2007. `http://meta.wikimedia.org/w/index.php?title=Cache_strategy&oldid=720259`.

[14] Joseph C. Morris. DistriWiki:: a Distributed Peer-to-Peer Wiki Network. In *WikiSym '07: Proceedings of the 2007 International Symposium on Wikis*, pages 69–74, New York, NY, USA, 2007. ACM.

[15] B. Clifford Neuman. Scale in Distributed Systems. In T. L. Casavant and M. Singhal, editors, *Readings in Distributed Computing Systems*, pages 463–489, Los Alamitos, CA, 1994. IEEE Computer Society.

[16] Felipe Ortega and Jesus M. Gonzalez Barahona. Quantitative Analysis of the Wikipedia Community of Users. In *WikiSym '07: Proceedings of the 2007 International Symposium on Wikis*, pages 75–86, New York, NY, USA, 2007. ACM.

[17] Guillaume Pierre, Maarten van Steen, and Andrew S. Tanenbaum. Dynamically Selecting Optimal Distribution Strategies for Web Documents. *IEEE Transactions on Computers*, 51(6):637–651, June 2002.

[18] Vikram Sharma, Carson Black, and Brent Hoon Kang. Towards a Distributed Peer Encyclopedia Model. Technical Report TR-SIS-06-ISR-001, University of North Carolina at Charlotte, Charlotte, NC, USA, November 2006.

[19] Kunwadee Sripanidkulchai, Bruce Maggs, and Hui Zhang. An Analysis of Live Streaming Workloads on the Internet. In *IMC '04: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, pages 41–54, New York, NY, USA, 2004. ACM Press.

[20] Guido Urdaneta, Guillaume Pierre, and Maarten van Steen. A Decentralized Wiki Engine for Collaborative Wikipedia Hosting. In *WEBIST '07: Proceedings of the 3rd International Conference on Web Information Systems and Technologies*, pages 156–163, March 2007.

[21] Jakob Voß. Measuring Wikipedia. In *Proceedings 10th International Conference of the International Society for Scientometrics and Informetrics*, 2005.

[22] Dennis M. Wilkinson and Bernardo A. Huberman. Cooperation and Quality in Wikipedia. In *WikiSym '07: Proceedings of the 2007 International Symposium on Wikis*, pages 157–164, New York, NY, USA, 2007. ACM.