About the heterogeneity of web prefetching performance key metrics

Josep Domènech, Julio Sahuquillo, José A. Gil, and Ana Pont

Department of Computer Engineering (DISCA)
Polytechnic University of Valencia. Spain
jododе@doctor.upv.es, {jsahuqui,jagil,apont}@disca.upv.es

Abstract. Web prefetching techniques have pointed to be especially important to reduce web latencies and, consequently, an important set of works can be found in the open literature. But, in general, it is not possible to do a fair comparison among the proposed prefetching techniques due to three main reasons: i) the underlying baseline system where prefetching is applied differs widely among the studies; ii) the workload used in the presented experiments is not the same; iii) different performance key metrics are used to evaluate their benefits.

This paper focuses on the third reason. Our main concern is to identify which the main meaningful indexes are when studying the performance of different prefetching techniques. For this purpose, we propose a taxonomy based in three categories, which permits us to identify analogies and differences among the indexes commonly used. In order to check, in a more formal way, the relation between them, we run experiments and estimate statistically the correlation among a representative subset of those metrics. The statistical results help us to suggest which indexes should be selected when performing evaluation studies depending on the different elements in the considered web architecture.

The choice of the appropriate key metric is of paramount importance for a correct and representative study. As our experimental results show, depending on the metric used to check the system performance, results can not only widely vary but also reach opposite conclusions.

1 Introduction

The international and global nature of Internet makes arduous (or sometimes really impossible) to increase the system performance working in the networks and their interconnection elements because of the existing gap of time between technological advances in infrastructure and its use in the real life. As a consequence, research efforts have been concentrated on the web architecture and organization using and exporting techniques already learned and widely used in computer architecture to improve the performance.

Those techniques take advantage of the locality properties inherent to the web objects accesses. In the Web, the locality has three different characteristics: temporal, spatial and geographical, which permits to implement efficiently caching, prefetching and replication techniques in order to increase performance. As a result, many efforts

(commercial and research) applying those techniques in the Web architecture have been carried out to increase performance.

In this paper we focus on prefetching techniques, although some of our studies and conclusions presented in this paper can also be extended to the general web performance analysis.

Many research studies concentrate on the proposals of new prefetching techniques. Performance comparison studies among them can not be fairly done because the proposed approaches are applied and tested in different baseline systems using also different workloads and conditions. In addition, the studies present different performance key metrics to evaluate their benefits.

In order to do fair performance comparison studies we need to tackle the mentioned drawbacks. To deal with the first drawback in a previous work we proposed [1] a general experimental framework which permits the implementation of prefetching techniques in a flexible and easy way, under the same platform and real workloads; therefore, the same experimental conditions can be considered.

In this paper, we address our work to tackle with the second drawback. To this end, we analyze a large subset of key metrics and propose a taxonomy based in three main categories, which permits us to identify analogies and differences among the metrics commonly used and check experimentally their relation.

The remainder of this paper is organized as follows. Section 2 describes a generic web prefetching system in order to set the basic glossary of terms that will be used in the remaining sections. Section 3 presents the proposed taxonomy for prefetching performance metrics. Section 4 illustrates the usefulness of the different metrics with some experimental examples and finally, section 5 includes some concluding remarks.

2 Generic Prefetch Architecture and Basic Glossary

We assume a generic web architecture composed by three main elements: clients, servers and proxies. Note that proxies act both as a client for the server and as a server for the client.

It is important to remark the difference between the user and the client. The user is the person in front of a computer (or a similar device) demanding information, whereas the client is the software (i.e., the browser) with which the user interacts, that manages the search and request of the demanded information to the appropriate server.

The main aim of prefetching techniques is to reduce the average latency perceived by the user. Several prefetching related techniques have been proposed focusing on different ideas to exploit the benefits of the prefetch; for instance: some research studies propose clients to download objects prior to be requested by the user [2],[3], [4],[5]; other studies propose to preprocess dynamic content [6]; some others concentrate on how to make pre-connections to server [7]; and so on. All prefetching related techniques start predicting or trying to guess the next objects the client will access to. This part of prefetch is usually referred as the prediction engine. Then, the prediction results are submitted to the prefetching engine, which decides whether to

prefetch or not such results, depending on other parameters; e.g., available bandwidth or server load. Notice that both the prediction engine and the prefetching engine can be found in the same element (client, proxy or server). We define below some basic variables that will be used in section 3:

- *Predictions*: amount of predicted objects by the prediction engine.
- *Prefetchs*: amount of prefetched objects by the prefetching engine.
- *GoodPredictions*: amount of predicted objects that are subsequently demanded by the user.
- *BadPredictions*: those predictions that not result in good predictions.
- *PrefetchHits*: amount of the prefetched objects that are subsequently demanded by the user.
- *ObjectsNotUsed*: amount of prefetched objects never demanded by the user.
- *UserRequests*: amount of objects the user demands.

Analogously, we can define byte related variables ($Predictions_B$, $Prefetchs_B$, and so on) by replacing the objects by the corresponding size in bytes in their definition.

3 Web Performance Indexes Taxonomy

This section surveys the web performance indexes appeared in the open literature focusing on prefetch aspects. To the better understanding of the meaning of those indexes, we classify them into three main categories (see Figure 1), attending to the system feature they evaluate:

- Category 1: prediction related indexes.
- Category 2: resource usage indexes.
- Category 3: end-to-end perceived latencies indexes.

The first category is the main one when comparing prediction algorithms performance and includes those indexes which quantify both the efficiency and the efficacy of the algorithm (e.g., precision). The second category quantifies the additional cost that prefetching incurs (e.g., traffic increase or processor time). This cost may become really high; thus, it must be taken into account when comparing prefetching techniques, thus those indexes can be seen as complementary measures. Finally, the third category summarizes the performance achieved by the system from the user point of view. Notice that prefetching techniques must take care of the cost increase because they can negatively impact on the overall system performance (traffic increase, user perceived latencies). Therefore, the three categories are closely related since in order to achieve a good overall performance (category 3) prefetching systems must trade off the aggressiveness of the algorithm (category 1) and the cost increase due to prefetching (category 2).

Different definitions for the same index can be found in the literature (e.g., precision) and this fact increases the heterogeneity of the research efforts. In order to make more readable this survey, we only include the definition we consider more precise and appropriate for evaluation purposes. In the cases where several names match the same definition, we select the most appropriate index name our point of view. The goal of this section is not only to help the understanding of the indexes but also to discuss their usefulness, distinguishing those used for comparison purposes in

any prefetching systems from those applicable to a particular prefetching technique (i.e. specific). Specific indexes are only found in the Category 1 (Prediction), as shown in Figure 1.

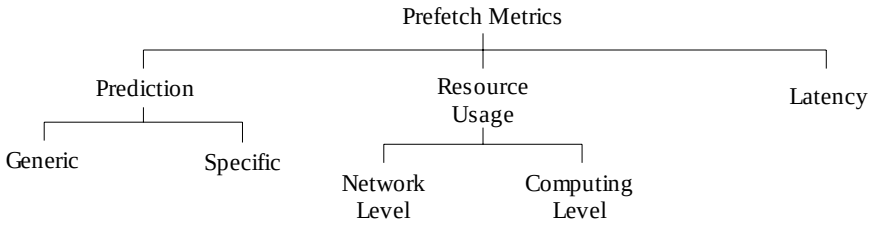

Fig. 1. Prefetching metrics taxonomy

3.1 Prediction Related Indexes

This group includes those indexes aimed at quantifying the performance that the prediction algorithm provides. Prediction performance can be measured at different moments, for instance when the algorithm makes the prediction and when prefetching is applied in the real system. Thus, each index in this category has a *dual* index; e.g., we can refer to precision of the algorithm and precision of prefetch. Notice that those indexes measured when the prediction list is given do not take into account system latencies because the prediction algorithm works independently of the underlying network and the user restrictions.

3.1.1 Generic Indexes

Precision (Pc). *Precision* measures the ratio of good predictions to the number of predictions [8],[9],[10],[11] (see equation 1). Precision, defined in this way, just evaluates the algorithm without considering physical system restrictions; e.g., cache, network or time restrictions; therefore, it can be seen as a theoretical index.

Other research studies refer to this index as *Accuracy* [4],[12],[13],[14],[15],[16], [17] while some others use a probabilistic notation; e.g., some Markov chains based models like [18] *Pr(hit|match).*

Some research works measure the impact on performance of the precision [4],[14]. In these cases, the number of prefetched objects and prefetch hits are used instead of the number of predictions and good predictions respectively (see equation 2).

$$Pc = \frac{GoodPredictions}{Predictions} \quad (1)$$

$$Pc = \frac{PrefetchHits}{Prefetchs} \quad (2)$$

Recall (Rc). *Recall* measures the percentage of requested objects that were previously prefetched [9],[10],[11]. The *recall* quantifies the weight of the predicted (see equation 3) or prefetched objects (see equation 4) over the amount of objects requested by the user. Notice that this index only deals with the number of good predictions made but not with the total amount of predictions.

Some research works refer to this metric as *usefulness* [16],[17] or *hit ratio* [19]. *Predictability* has been employed in [6] to refer to the upper limit of the *recall*.

$$Rc = \frac{GoodPredictions}{UserRequests} \quad (3)$$

$$Rc = \frac{PrefetchHits}{UserRequests} \quad (4)$$

Applicability. Bonino et al [8] define *applicability* as the ratio of the number of predictions to the number of requests. This is the only research work using this index; nevertheless, authors fail when stating this index as synonymous of *recall*. Notice that this index can be obtained from the previous ones (i.e., dividing *recall* by *precision*); therefore no additional information is given. This is the main reason because no other works use this index.

$$Applicability = \frac{Predictions}{UserRequests} \quad (5)$$

Precision alone or together with *recall* has been the most widely used index to evaluate the goodness of prediction algorithms. Each response request time consists of four main time components; i.e., connection establishment, request submitting, request processing, and response transference. Both *precision* and *recall* are closely related on the three first components. Therefore, authors feel that a complete comparison study about prediction algorithms should also include byte related indexes to quantify the last component. In this sense, analogously to the web proxy caching indexes (e.g., the *byte hit ratio*) [20], we propose the use of *byte precision* and *byte recall* as indexes to estimate the impact of prefetching on the time that the user wastes when waiting for the bytes of the requested objects.

Byte Precision (Pc_B). *Byte precision* measures the percentage of prefetched bytes that are subsequently requested. It can be calculated by replacing the number of predicted objects by their size in bytes in equation 1.

Remark that, like *precision* does, *byte precision* quantifies how good the predictions are, but measured in bytes instead of objects. An earlier approach to this index is the *Miss rate ratio* [21], described below.

$$Pc_B = \frac{GoodPredictions_B}{Predictions_B} \quad (6)$$

Byte Recall (Rc_B). *Byte recall* measures the percentage of demanded bytes that were previously prefetched. As mentioned above, this index quantifies how many accurate predictions are made, measured in transferred bytes.

This index becomes more helpful than the previously mentioned *recall*, when the transfer time is an important component of the overall user perceived latency.

$$Rc_B = \frac{GoodPredictions_B}{UserRequests_B} \quad (7)$$

3.1.2 Specific Indexes

Request savings. *Request savings* measures the number of times that a user request hits in the browser cache or the requested object is being prefetched, in percentage of the total number of user requests [2]. Furthermore, the request savings can be broken down into three groups depending on if they were previously prefetched; have been partially prefetched or cached as normal objects.

Notice that when prefetching is user-initiated the number of requests increases; therefore, this index makes sense when prefetching is proxy or server initiated. Fan *et al* use this index in a prefetch system where the proxy pushes objects to the client. Nevertheless, this index could be also used when prefetching pushes objects from the server to the proxy or from the server to the client (with no intermediate proxies).

Miss rate ratio. Bestavros defines the *miss rate ratio* [21] as the ratio between the byte miss rate when the prefetch is employed to the byte miss rate when the prefetch is not employed, where the byte miss rate for a given client is the ratio of bytes not found in the client's cache to the total number of bytes accessed by that client.

As one can see, this index quantifies in which percentage the miss rate in the client cache drops due to the prefetching system. This is a specific index since it is only applicable to those systems storing the prefetched objects in the client's cache.

Probability of making a prediction. Pitkow et al [18] quantify the probability that the last accesses match the pattern prediction; in such case the prefetch system computes the prediction outcomes. This index can be applied, for example, to those systems based in Markov models, but can not be applied to a large subset of prefetching systems; e.g., the top-10 approaches [19], so that it is classified as specific.

3.2 Resource usage

The benefits of prefetching are achieved at the expense of using additional resources. This overhead, as mentioned above, must be quantified because they can negatively impact on performance.

Although some prediction algorithms may require huge memory or processor time (e.g., high order Markov models), it is not the current general trend, where the main

prefetching bottleneck is the network traffic. Therefore, we break down indexes in this category into two subgroups: network level and computing level.

3.2.1 Network level

Traffic increase (ΔTr_B). *Traffic increase* quantifies the increase in traffic (in bytes) due to unsuccessfully prefetched documents [19]. It is also called *wasted bandwidth* [2], *extra bytes* [10], *network traffic* [17], and *bandwidth ratio* [21].

When using prefetch, network traffic usually increases due to two side-effects of the prefetch: bad predictions and overhead. Bad predictions waste network bandwidth because these objects are never requested by the user. On the other hand, the network traffic increases due to the prefetch related information interchange, called *Network overhead* by [4]).

Several research studies fail in not taking into account that overhead [16],[19],[22]; therefore, their results can not accurately estimate prefetching costs.

$$\Delta Tr_B = \frac{ObjectsNotUsed_B + NetworkOverhead_B + UserRequests_B}{UserRequests_B} \quad (\mathbf{8})$$

Object traffic increase (ΔTr_{ob}). *Object traffic increase* quantifies in which percentage increases the number of documents that clients get when using prefetching. Nanopoulus et al [16] refer to this index as *network traffic* and Rabinovich [10] as *extra requests*.

As equation 9 shows, this index estimates the ratio of prefetched objects never used with respect to the total user's requests. It is analogous to the *traffic increase*, but it measures the overhead in number of objects.

$$\Delta Tr_{ob} = \frac{BadPredictions + UserRequests}{UserRequests} \quad (\mathbf{9})$$

3.2.2 Computing level

Server load ratio. *Server load ratio* is defined as the ratio between the number of requests for service when speculation is employed to the number of requests for service when speculation is not employed [21].

Space and Time overhead. In addition to the server load, some research works discuss how the overhead impact on performance. For instance, Duchamp [4] discusses the memory and processor time the prefetch would need.

3.3 Latency related indexes

Indexes belonging to this category include those aimed at quantifying the end-to-end latencies; e.g., user or proxy related latencies. The main drawback of these indexes is

that they include several time components, some of them difficult to quantify. Many times researchers do not detail what components their measures include, although they use a typical index name; e.g., latency. This situation is not the best for the research community, due to the fact that different proposals can not be fairly compared among them.

Through the different research works, latencies are measured both per page and per object. The *Latency per page* (L_p) is calculated by comparing the time between browser's initiation of an HTML page GET and browser's reception of the last byte of the last embedded image for that page [4]. Analogously, the *Latency per object* (L_{ob}) can be defined as the elapsed time since the browser requests an object until it receives the last byte of that object. In order to illustrate the benefits of prefetching, researchers calculate the ratio of the latency prefetching achieves (either per page [2],[4],[14] or per object [23]) to the latency with no prefetching.

Unfortunately, most proposals that use *latency* when measuring performance do not specify which latency they are referring to. This fact can be misleading because both indexes do not perform in the same way. In order to illustrate this fact we present the following example: a user requests an HTML object embedding two images (IMG1 and IMG2). As figure 2 shows, the transference of the HTML file starts at t_0 and ends in t_5. At times t_1 and t_3 the browser reads and processes the IMG tags of the embedded images then starts the transferences, which end at t_2 and t_5, respectively. In this case, the cumulative L_{ob} is the sum of the time taken by the three transferences ($L_{ob} = t_4-t_0 + t_2-t_1 + t_5-t_3$) where the *Latency per page* (L_p) is t_5-t_0. If it is assumed that IMG1 was previously prefetched, no waiting time for such object will be plotted; i.e. t_1 will match t_2, so it reduces L_{ob} but not L_p which will remain the same value.

To observe this feature into a real environment for a given client is necessary that the object retrieving times are each other independent. Nevertheless, although times are not independent, both indexes can measure different values, as experimental results will show.

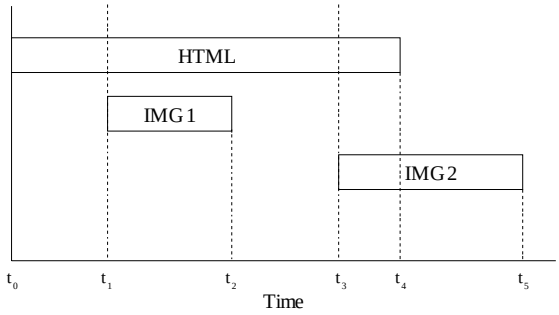

Fig. 2. Example where latency metrics behave different when the prefetch hits on one of the images

Furthermore, the *Latency per object* measured when an object is downloaded by the web browser has two main components: i) the queue time, since the browser has a limited number of connections, and ii) request time, including the time of connection establishment (if needed) and the object transference. The first component is often

ignored whereas other research studies [5],[21] do not specify whether the latency measured includes queue time. The first component should not be ignored because prefetch hits do not compete for a free connection so the queue time of the remaining objects decreases and, consequently, their latency.

On the other hand, several names have been used instead of *latency*; for instance, *access time* [3], *service time* [21] and *responsiveness* [22],[24].

3.4 Summary: synonymous and experimental index category used

Through the proposed taxonomy, we discussed the large variety of index names (synonymous) used to refer to the same metric. This heterogeneity can be observed through web performance studies appeared in the literature, adding extra difficulty to obtain a general view of the subject. Table 1 offers a scheme of the current situation. In addition, we also found that the same index name has been used when measuring different variables (e.g., *network traffic* appears both for *traffic increase* and for *object traffic increase*). As one can observe, the widest heterogeneity appears in the first category due to the large diversity of prediction algorithms appeared in the literature and its importance in the prefetching systems, which are the main focus of this work.

Table 2 relates the research works found in the open literature with the categories of indexes used in such works. As discussed above, a complete research work should include indexes belonging to the three categories. However, we only found this fact in 15% of the explored works, just three of twenty works (row 1).

Notice that the first column shows that 75% of the studied research works have considered at least one of the indexes belonging to the prediction category. Moreover, 40% of the studies only measure prediction metrics to evaluate the performance of the system. The second column shows that 60% of the papers do not measure resource usage metrics. Finally, in the third column we can observe that 55% of the research works do not quantify the end perceived latencies.

4 Experimental Examples

This section has three main goals. The first one is to illustrate the usefulness of the proposed indexes (i.e., *byte precision* and *byte recall*) as well as how they differ from the analogous classical ones. The second one is to study and show how indexes are related among them. Finally, the third goal is to identify the most meaningfulness indexes when evaluating the overall system performance.

In order to reach the second goal, we choose a subset of representative pairs of indexes that could be potentially related. Those pairs are selected from the most widely used indexes as discussed in Section 3: *precision*, *byte precision*, *recall*, *byte recall*, *traffic increase*, *object traffic increase*, *latency per page* and *latency per object*. A graphic is plotted for each pair in order to detect the possible relations. Then, in those graphics where some sign of linear relation appears, we quantify it statistically.

Table 1. Relation between the selected index name in this paper and those appeared in the literature

Category	Selected name	Literature	
		Name	**References**
1. Prediction	*Precision*	*Precision*	[8],[9],[10],[11]
		Accuracy	[4],[12],[13],[14],[15],[16],[17]
		Pr(hit|match)	[18]
	Recall	*Recall*	[9],[10],[11]
		Usefulness	[16],[17]
		Hit Ratio	[19]
		Predictability	[6]
	Applicability	*Applicability*	[8]
2. Resource usage	*Traffic increase*	*Traffic increase*	[3],[4],[19]
		Wasted Bandwidth	[2]
		Bandwidth ratio	[21]
		Extra bytes	[10]
		Data transfer	[22]
		Network Traffic	[17]
	Object traffic increase	*Network Traffic*	[16]
		Extra requests	[10]
3. Latency	*Latency per page*	*Latency*	[2],[4],[14]
		Responsiveness	[22],[24]
	Latency per object	*Latency*	[5],[23]
		Access time	[3]
		Service time ratio	[21]

In order to obtain the experimental results presented in this section, we use the experimental framework described in [1]. To illustrate how the different indexes behave, we need both a prefetching system and a non-prefetching system. In order to provide a prefetching system, we implement the prefetching algorithm proposed by Padmanabhan and Mogul [3]. This algorithm uses a threshold value so that objects with less probability than such value to be requested in the following k accesses are not prefetched (in our experiments we take k=4). In order to observe how the indexes behave in a wide range of situations, the experiments were run ranging the threshold from 0.1 to 0.9.

Table 2. Relation between research studies and indexes used grouped by category

References	***Category***			
	1. Prediction	***2. Resource***	***3.Latency***	***%***
[2],[4],[21]	X	X	X	15
[3],[22]		X	X	10
[14]	X		X	5
[5],[23],[24]			X	15
[16],[17],[19]	X	X		15
[6],[8],[9],[11],[12],[13],[15],[18]	X			40

The system was configured to simulate users accessing to *Marca*, which is a Spanish popular news web server (www.marca.es). A trace collected during one week (about 145,000 accesses) was used to train the prediction engine while the logs of the following day (about 35,000 accesses) were used to obtain simulation results. Each simulation included about 250 clients. Plotted points in the figures that this section includes refer to the performance indexes measured for every client in each experiment (each plot consists of about 2,500 points). On the other hand, the measured values of the prediction indexes showed in this section refer to the prefetched objects, not to the predicted objects, as they are close to the real system performance. Available bandwidth per each simulated user was ranged from 48 kbps to 400 kbps but, due to space limitations, only 200 kbps users are presented and analyzed in this paper. In the whole set of experiments we saw that the correlation among the indexes increases as the available bandwidth does.

Figure 3 plots the *Latency per object* ratio to the *Latency per page* ratio, both referring to different alternatives about how latency can be measured (as discussed in Section 3). Note that depending on the way the latency is measured, it is possible to reach not only different but also opposite conclusions. Suppose that we take a point in the upper left-side quarter and we consider the *Latency per object;* in such case, the prefetching system outperforms the non-prefetching system. However, if we consider the *Latency per page*, we would conclude the opposite. The correlation coefficient between both latency ratios corresponding to the points plotted in Figure 3 is 0.60, i.e., the indexes present a certain linear correlation but it is far from being strong; so that *Latency per object* and *Latency per page* ratios are not directly comparable.

Consequently, we suggest that studies should differentiate the use of both latency ratios because they are addressed to explore the performance from different points of view. The *Latency per page* ratio evaluates the system performance from the user's point of view (since it measures the latency as perceived by the user) while the *latency per object* ratio measures the performance from the http protocol point of view. Therefore, it should be used when the meaning of a page is not so clear; e.g. in a proxy.

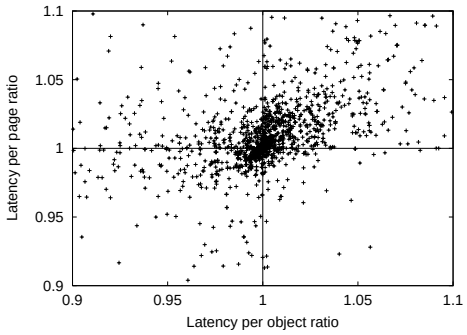

Fig. 3. *Latency per object* ratio as a function of *latency per page* ratio

Figure 4 presents an almost horizontal curve showing no apparent relation between precision and latency related indexes. The correlation coefficients showed in table 3 quantify this negligible linear correlation, which confirms the visual appreciation.

Table 3. Linear correlation coefficient between prediction and latency indexes

	Precision		**Recall**	
Latency	**Pc**	**Pc_B**	**Rc**	**Rc_B**
L_p	-0.294	-0.403	-0.198	-0.526
L_{ob}	-0.291	-0.228	-0.625	-0.363

Figure 5 shows the relation between precision and traffic related indexes. One can appreciate that points define an area whose superior slope has certain resemblance to an exponential function. The figure can be interpreted as the higher the (*byte*) *precision* is the lower probability of having high (*object*) *traffic increase*. The upper row (Figure 5.a, and Figure 5.b) shows that *precision* has better relation (defines better the exponential area) to the *object traffic increase* than to the *traffic increas*e, since points in this plot appear more widely dispersed. On the other hand, the right column (Figure 5.b, and Figure 5.d) shows the relation between *traffic increase* and both *precision* and *byte precision*. As expected, *byte precision* is closer related to *traffic increase* than *precision*. Despite this, to the best of our knowledge, no other research work has used *byte precision* as a performance index.

As the *traffic increase* can not be considered the main aim of the prefetching, precision indexes can not be used as main indexes to measure the system performance.

Figure 6 shows the relation between the *Recall* and both latency indexes. One can observe that there is a linear relation between the *recall* index and the *latency per object* ratio, but not to *latency per page* object. Due to fact that *recall* increases with the good predictions count, this index has a significant impact on the *latency per object*, but they are not directly comparable (the correlation coefficient is 0.62, as shown in table 3). Some extra elements not gathered by the *recall*, like queued time and object size heterogeneity justify why the relation is not stronger. On the other hand, the *latency per page* is not explained by the *recall* index since it involves more elements that affect it, like the simultaneous transference of objects (as explained in

section 3.3). These results corroborate the differences between both latencies observed in the figure 3.

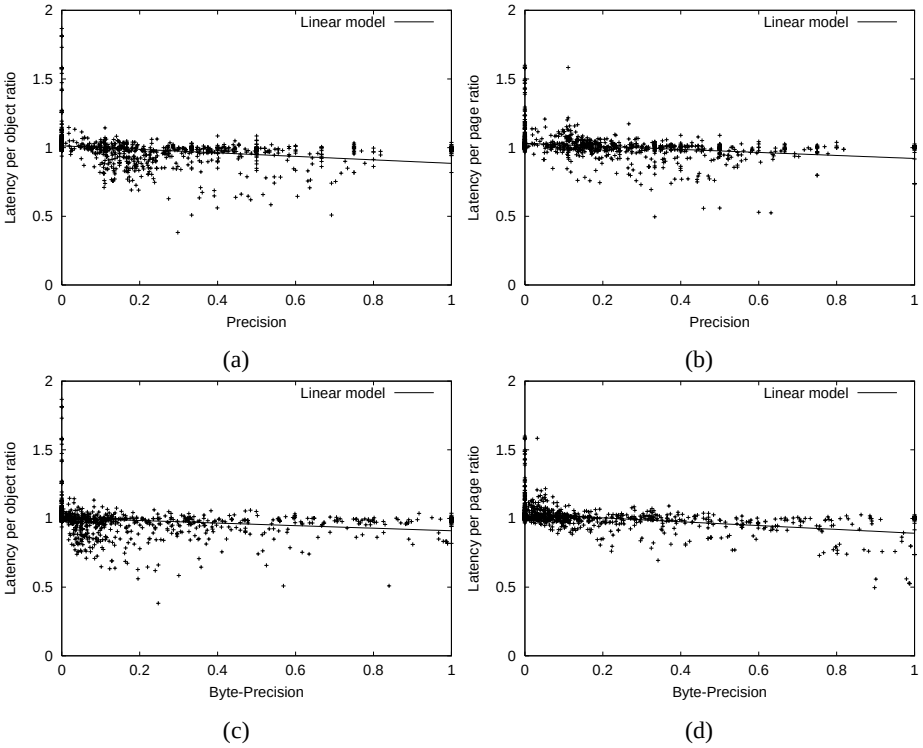

Fig. 4. Relation between precision and latency related indexes

5 Conclusions

A large variety of key metrics have appeared in the open literature in order to evaluate the performance of web prefetching systems. This paper has analyzed this wide heterogeneity trying i) to clarify both index definitions and how they are related, and ii) to help when deciding which metric or index should be selected to evaluate the system performance.

We have proposed a taxonomy, classifying the indexes related to prefetch in three main categories according to the part of the system they are addressed to evaluate: prediction, resource usage and latency. The goal of this taxonomy is not only to help the understanding of the indexes definition but also to analyze analogies and differences between them, in order to identify which the main useful metrics are when carrying out performance studies.

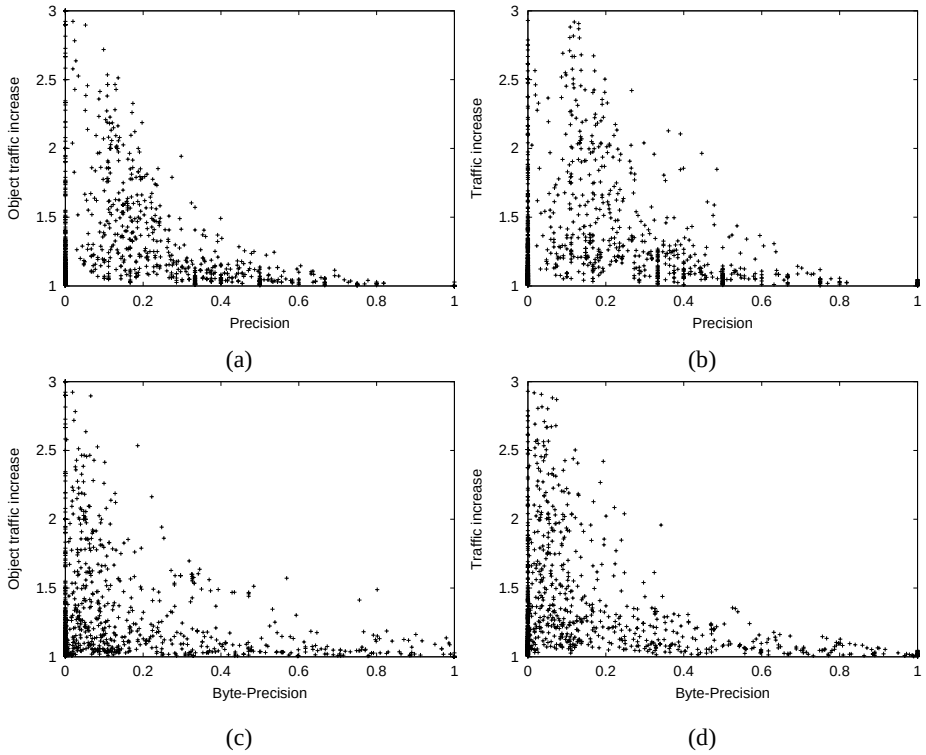

Fig. 5. Relation between precision and network level related indexes

For each metric or index we have provided a definition (the one we considered more precise) among the large variety appeared in the literature. Then, looking for these definitions we observed certain analogies. To check possible relations among those indexes in a more formal way, we ran experiments and calculated the statistical correlation among a representative set of them. From these experiments we can extract the main following conclusions:

- Performance studies should include at least latency related metrics. Depending on the goal of the performed study *latency per page* or *per object* is preferred. For instance, if the goal is to analyze the user's point of view, the latency per page must be included; whereas when evaluating from the point of view of a proxy server, the *latency per page* makes no sense due to the lack of page concept. Consequently, the *latency per object* can be the most useful metric.
- Latencies alone can not be used as metrics to check performance. Studies must analyze how the latencies reduction has been achieved for a given proposal. In this sense, resource usage indexes should be taken into account. *Traffic increase* is the one that provides more information; therefore, we suggest that performance studies should include at least this index.
- Our discussion has shown that is not recommendable to perform studies about the behavior of prefetching techniques just focusing on the algorithm point of view. Nevertheless, if some studies focus on this part, they should include *recall* and *byte*

recall as performance metrics because they are the most correlated to the *latency per object* and *latency per page* respectively.

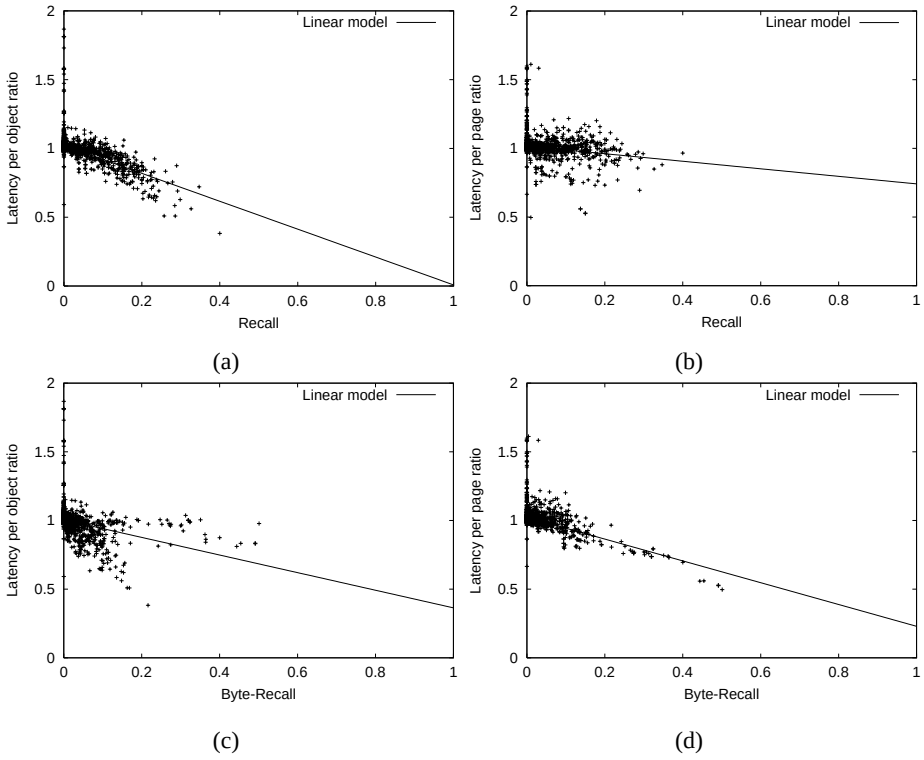

Fig. 6. Relation between recall and latency related indexes

References

1. Domènech, J., Pont, A., Sahuquillo, J., and Gil, J. A.: An experimental framework for testing web prefetching techniques. Submitted (2004)
2. Fan, L., Cao, P., and Jacobson, Q.: Web prefetching between low-bandwidth clients and proxies: potential and performance. Proceedings of the ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems. Atlanta, USA (1999)
3. Padmanabhan, V., and J. C. Mogul.: Using predictive prefetching to improve World Wide Web latency. Proceedings of the ACM SIGCOMM '96 Conference. Palo Alto, USA (1996)
4. Duchamp, D.: Prefetching hyperlinks. Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems. Boulder, USA (1999)
5. Kokku, R., Yalagandula, P., Venkataramani, A., and Dahlin, M.: NPS: A non-interfering deployable web prefetching system. Proceedings of the USENIX Symposium on Internet Technologies and Systems. San Antonio, USA (2003)

6. Schechter, S., Krishnan, M., and Smith, M. D.: Using path profiles to predict HTTP requests. Proceedings of the 7th International World Wide Web Conference. Brisbane, Australia (1998)
7. Cohen, E., and Kaplan, H.: Prefetching the means for document transfer: A new approach for reducing web latency. Proceedings of the IEEE INFOCOM. Tel Aviv, Israel (2000)
8. Bonino, D., Corno, F., and Squillero, G.: A real-time evolutionary algorithm for web prediction. Proceedings of the International Conference on Web Intelligence. Halifax, Canada, (2003)
9. Albrecht, D., Zukerman, I., and Nicholson, A.: Pre-Sending documents on the WWW: A comparative study. Proceedings of the 16th International Joint Conference on Artificial Intelligence. Stockholm, Sweden (1999)
10. Rabinovich, M. and Spatscheck, O.: Web caching and replication. Addison-Wesley. Boston, USA (2002)
11. Cohen, E., Krishnamurthy, B., and Rexford, J.: Efficient algorithms for predicting requests to web servers. Proceedings of the Conference on Computer and Communications. New York, USA (1999)
12. Cunha, C. R., and Jaccoud, C. F. B.: Determining WWW user's next access and its applications to pre-fetching. Proceedings of the 2nd International Symposium on Computers and Communication. Alexandria, Egypt (1997)
13. Zukerman, I., Albrecht, D., and Nicholson, A.: Predicting users' requests on the WWW. Proceedings of the 7th International Conference on User Modeling. Banff, Canada (1999)
14. Loon, T. S., and Bharghavan, V.: Alleviating the latency and bandwidth problems in WWW browsing. Proceedings of the USENIX Symposium on Internet Technologies and Systems. Monterey, USA (1997)
15. Davison, B. D.: Predicting web actions from HTML content. Proceedings of the 13th ACM Conference on Hypertext and Hypermedia. College Park, USA (2002)
16. Nanopoulos, A., Katsaros, D., and Manopoulos, Y.: Effective prediction of web-user accesses: A data mining approach. Proceedings of Workshop on Mining Log Data across All Customer Touchpoints. San Francisco, USA (2001)
17. Palpanas, T., and Mendelzon, A.: Web prefetching using partial match prediction. Proceedings of the 4th International Web Caching Workshop. San Diego, USA (1999)
18. Pitkow, J. and Pirolli, P.: Mining longest repeating subsequences to predict World Wide Web surfing. Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems. Boulder, USA (1999)
19. Markatos, E. P., and Chronaki, C. E.: A top-10 approach to prefetching the web. Proceedings of the INET' 98. Geneva, Switzerland (1998)
20. Cherkasova, L., and Ciardo, G.: Characterizing temporal locality and its impact on web server performance. Proceedings of the 9th International Conference on Computer Communication and Networks. Las Vegas, USA (2000)
21. Bestavros, A.: Using speculation to reduce server load and service time on the WWW. Proceedings of the 4th ACM International Conference on Information and Knowledge Management. Baltimore, USA (1995)
22. Khan, J. I., and Tao, Q.: Exploiting webspace organization for accelerating web prefetching. Proceedings of the International Conference on Web Intelligence. Halifax, Canada (2003)
23. Kroeger, T. M., Long, D. D. E., and Mogul, J. C.: Exploring the bounds of web latency reduction from caching and prefetching. Proceedings of the USENIX Symposium on Internet Technologies and Systems. Monterey, USA (1997)
24. Tao, Q.: Impact of Webspace Organization and User Interaction Behavior on a Prefetching Proxy. Ph. D. Thesis. Kent State University, USA (2002)