On Learning Strategies for Topic Specific Web Crawling

Charu C. Aggarwal
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598
charu@us.ibm.com

Abstract

Crawling has been a topic of considerable interest in recent years because of the rapid growth of the world wide web. In many cases, it is possible to design more effective crawlers which can find web pages belonging to specific topics. In this paper, we will discuss some recent techniques for crawling web pages belonging to specific topics. We discuss the following classes of techniques: (1) Intelligent Crawling Methods: These methods learn the relationship between the hyper-link structure/web page content and the topic of the web page. This learned information is utilized in order to guide the direction of the crawl. (2) Collaborative Crawling Methods: These methods utilize the pattern of world wide web accesses by individual users in order to build the learning information. In many cases, user access patterns contain valuable statistical patterns which cannot be inferred from purely linkage information. We will also discusses some creative ways of combining different kinds of linkage- and user-centered methods in order to improve the effectiveness of the crawl. We will discuss some of the recent algorithms proposed in each topic along with some discussions on the directions of future research.

1 Introduction

In recent years the world wide web has grown at a rapid pace. Currently, there are more than a billion documents on the web, and it continues to grow at a pace of more than a million documents a day. In many cases, it is desirable to be able to find documents belonging to specific topics. In order to achieve this goal, a number of search engine technologies such as $Yahoo!$, *Lycos* and $AltaVista$ have recently surfaced [30, 28, 29]. Such technologies provided a limited query capability using specific keywords or phrases in the document. In many cases, it may be desirable to find documents on the web which satisfy particular kind of conditions such as the overall topic of the document.

Recently, a technique called *focussed crawling* [9] has been proposed for automated resource discovery. The focussed crawling technique utilizes topical locality on the web in order to perform resource discovery. The idea is to start at a few well chosen points and maintain the crawler within the range of these topics. Starting with this pioneering work, there has been some recent work [13, 22] in order to improve the efficiency of the crawl. We refer the reader to [1, 5, 7, 8, 14, 18] for some closely related work.

In order to define the relevance of a web page, a *predicate* may be utilized. This predicate may be in the form of a particular set of keywords present in the document, a particular category which is implicitly defined by a classifier, or a combination of the above. Some examples of useful predicates are as follows:

- All web pages which a classifier defines in the "SPORTS" category, and which contain the keyword "hockey".
- All web pages which contain the keyword "NBA", and whose URL ends with the extension ".uk".

The utilization of a predicate to crawl web pages of interest is a very flexible model which is often not supported by some of the well known methods such as focussed crawling [9]. Instead, the focussed crawling method uses a pre-defined taxonomy of classes in order to perform the crawl. Such a procedure comes with its disadvantages, since the effectiveness of the crawl is sensitive to the underlying nature of the taxonomy. This greatly restricts the nature of the predicate which are used as queries for the crawling process. On the other hand, the learning methods discussed in the paper can handle arbitrary predicates rather than those which are drawn from a hierarchical taxonomy.

Methods such as focussed crawling are essentially ad-hoc heuristics in picking a particular strategy for the crawling process.[1] A broader goal would be relate particular attributes of the web page to the predicate of interest. Such methods are referred to as learning strategies. We will discuss two recent such methods in some detail.

The first class of methods [4] uses linkage based information in order to improve the effectiveness of the crawling. In these techniques, various features of the web page such as its content, URL extensions, and hyperlink structure may contribute to the classification of a page as a more likely candidate for belonging to a particular topic. Such methods are however essentially linkage based since the URL pointers play the key role in the resource discovery process.

The second class of methods [1] is user-based, since it utilizes patterns of user behavior in order to determine topics of interest. In the former case, the linkage based information is used directly under the assumption that web pages show topical locality. The advantages of the user-based system is that it can be effective in cases where it is not possible to reliably use linkage structure in order to search for topical resources. This is partially because of the increasing noisy nature of the links on the web [5, 11, 16]. Such links may correspond to banners, advertisements and other content which do not carry specific information about resource discovery. This increased noisy behavior is partially a result of the rapid commercialization of the world wide web. In user-centered methods, information from world wide web traces is utilized in order to determine the candidate pages of interest.

[1] We note that the focussed crawling technique [9] also uses a classifier in the resource discovery process. However, this classifier is only utilized for defining the relevance of the web page to a given topic. In other words, the classifier acts as the predicate definition for the web page. This does not mean that the crawler is itself a learning process in terms of the strategy it uses for finding candidate web pages of interest.

In user-centered systems, we make use of logs of user access behavior on public domain proxies. An example of such a *public-domain proxy* is the Squid [24]. The access patterns (web logs) of such proxies are publically available on the world wide web. These logs can be utilized in order to determine the topical connection between users and web pages. Such topical connections can then be leveraged in order to compute the relevance of a given web page to the user-specified predicate. We note that in the second case, the linkage between web pages is *indirect*, since instead of using URL pointers, we are using the commonality in browsing behavior in order to find new and relevant web pages. We note that such a strategy shares some aspects of collaborative filtering techniques [2, 26], though the aim and scope of our system is different in many critical respects. Whereas collaborative filtering is only useful in recommending web pages which are interesting to a user, our system is designed to find web pages belonging to particular topical classes or predicates.

This paper is organized as follows. In the next section, we will discuss the basic framework which can be leveraged in order to create an effective crawling system. In section 3, we will discuss the use of linkage based techniques in order to perform effective crawling. In section 4, we will discuss how a user centered system can be created in order to create an effective crawling system. In section 5, we will discuss methods for combining the linkage and user-centered systems. In section 6, we will present the conclusions and summary.

2 The Learning Crawler Framework

The basic framework for learning based crawlers is illustrated in Figure 1. The technique uses a graph search technique on the world wide web pages, in which the pages are explored sequentially in order of a *structural search mechanism.* At each point of the search, a candidate list of web pages is maintained. The pages on the candidate list are examined in order of increasing priority. This priority value is calculated using a criterion which is dependent on the particular strategy being utilized.[2] When a candidate web page F is *examined*, it means that the web page has already been accessed from the world wide web. At this point, it is checked whether or not the web page satisfies the user-defined predicate. If such is indeed the case, then the web page is saved as a relevant web page. In addition, considerable learning information can be gleaned from whether or not the web page satisfies the user-defined predicate. This learning information is stored in the statistics $\mathcal{K}$. For example, in the case of a linkage based strategy, it may be more desirable to the set of candidates pointed to from a web page are utilized in order This structured search mechanism may vary according to the particular method being used.

- In linkage based mechanisms [4], the URLs contained in the web page are used as the criterion for expanding the candidate list. Specifically, we check the URLs which are contained in that web page as pointers. The set of URLs $\mathcal{N}$ are generated by the *CreateDescendentCandidates* procedure. This procedure also computes the priorities of the different web pages using the appropriate criterion. We note that the criterion for linkage based systems may be different than the criterion for user-centered systems.

[2] We will discuss more on the aspect of priority computation in later sections.

```
Subroutine CreateDescendentCandidates(WebPage: $F$, Learning Statistics: $K$);
   Find all immediate descendents of $F$ using the
   appropriate criteriuon for defining a descendent. Denote
   descendents by $\mathcal{N}$. Calculate the
   priority of the candidate pages in $\mathcal{N}$
   with the use of learning statistics in $\mathcal{K}$.
Subroutine ExpandList(CandidateList: Cand, NewCandidates: $\mathcal{N}$);
  { Add those candidates in $\mathcal{N}$ to $Cand$ which
  have priority above a user-defined threshold; }

Algorithm LearningCrawlerFramework(StartingSeeds: $S$);
begin
  $Cand = S$;
  Set priority of each element in $Cand$ to 1;
    while $Cand$ is not empty do
    begin
       Sort $Cand$ in order of decreasing priorities;
       Pick the first page $F$ from $Cand$;
       Issue a get request for URL $F$ on the world wide web;
       if $F$ satisfies the predicate then save $F$;
       Update learning statistics $\mathcal{K}$;
       $\mathcal{N}$ = CreateDescendentCandidates($F$, $\mathcal{K}$);
       ExpandList($Cand$, $\mathcal{N}$);
       Delete $F$ from $Cand$;
    end;
end
```

Figure 1: The Basic Framework for Learning Based Crawlers

- In user-centered systems [1], the behavior of users is utilized in order to guide the resource discovery process. This behavior may be obtained from publically available web traces. In this case, instead of using the set of hyperlinks emanating from a web page, we use the set of web pages which are browsed by the different users in order to guide the process of resource discovery.

3 Use of Linkage Based Techniques

We note that a linkage based crawler essentially searches for statistical aspects of the data which are relevant for the purpose of crawling. These features may include the following aspects of web pages:

- The set of words in the web page.
- URL tokens from the candidate URL. For example, if we are looking for skiing web pages, the word "ski" in the URL provides evidence of the nature of that web page.
- Statistics of the number of inlinking web pages which satisfy the predicate.
- Statistics of the number of siblings of a candidate which have already been crawled that satisfy the predicate. A web page is said to be a sibling of a candidate URL, when it is linked to by the same page as the candidate. As the crawl progresses, the importance of each of the above set of statistics is learned by the crawler.

As discussed earlier, the statistical model maintains a dynamically updated set of statistical information $\mathcal{K}$ which it has learned during the crawl, and a set of features in the given web page and computes a priority order for that web page using this data. As we shall see later, the particular priority order which we determine calculates the interest factor on the likelihood that the features for a candidate web page make it more likely that this page satisfies the predicate.

In order to calculate the priorities, we compute the ratio which signifies whether a given set of events makes it more likely for a candidate to satisfy the user defined predicate. We will develop some notations and terminology in order to explain the model a little better. Let C be the event that a crawled web page satisfies the user defined predicate. For a *candidate page* which is about to be crawled, the value of $P(C)$ is equal to the probability that the web page will indeed satisfy the user-defined predicate if it is crawled. The value of $P(C)$ can be estimated by the fraction of web pages already crawled which satisfy the user defined predicate.

Let E be a fact that we know about a candidate URL. This fact could be of several types. For example, it could be a fact about the content of the inlinking web pages into this candidate URL, it could be a fact about the set of tokens in the string representing the URL, or it could be a fact about the linkage structure of the URL. We will explore all of these options slightly later.

Our knowledge of the event E may increase the probability that the web page satisfies the predicate. For example, consider the case when the candidate URL is linked to by another web page which belongs to the same topic. In such a case, it is evident from earlier results on focused crawling [9], that the resulting web page is more likely to satisfy the predicate. Thus, in this case, we have

$P(C|E) > P(C)$. In order to evaluate $P(C|E)$, we use the following relationship:

$$P(C|E) = P(C \cap E)/P(E) \tag{1}$$

Therefore, we have:

$$P(C|E)/P(C) = P(C \cap E)/(P(C) \cdot P(E)) \tag{2}$$

The idea is that the values of $P(C \cap E)$ and $P(E)$ can be calculated using the information that has been accumulated by the crawler. This is the self-learning data $\mathcal{K}$ which is accumulated over time during the crawling process. Correspondingly, we calculate the interest ratio for the event C, given event E as $I(C, E)$. Therefore, we have:

$$I(C, E) = P(C|E)/P(C) \tag{3}$$

Note that when the event E is favorable to the probability of the candidate satisfying the predicate, then the interest ratio $I(C, E)$ is larger than 1. Correspondingly, when the event E is unfavorable, then this interest ratio will be in the range $(0, 1)$. Such a situation occurs when the event E makes the candidate less desirable to crawl.

Let $E_1 \dots E_k$ be a set of k events. Let the composite event $\mathcal{E}$ be defined by the occurrence of all of these events. In other words, we have $\mathcal{E} = E_1 \cap E_2 \dots E_k$. Then the composite interest ratio is defined as follows:

$$I(C, \mathcal{E}) = \pi_{i=1}^{k} I(C, E_i) \tag{4}$$

The composite event $\mathcal{E}$ is interesting when the corresponding interest ratio is larger than 1. We will now proceed to examine the different factors which are used for the purpose of intelligent crawling.

In order to identify the value of the content in determining the predicate satisfaction of a given candidate page, we find the set of words in the web pages which link to it (inlinking web pages). A statistical analysis is performed on this set of words. We define the event Q_i to be true when the word i is present in one of the web pages pointing to the candidate.

Let $M = \{i :\ \text{Event } Q_i \text{ is true}\}$
Now, let us consider a given word i such that $i \in M$. Therefore, the event Q_i is true. If C be the event that a candidate URL is likely to satisfy the predicate, then let us calculate the value of $I(C, Q_i)$:

$$I(C, Q_i) = P(C \cap Q_i)/(P(C) \cdot P(Q_i)) \tag{5}$$

It now remains to estimate the parameters on the right hand side of the above equation. In order to estimate these parameters, we can only rely on the experimental evidence of the web pages which we have crawled so far. The exact details of these estimations will be discussed in a later section.

In order to filter out the noisy words which do not carry much statistical significance, we calculate the level of significance at which it is more likely for them to satisfy the predicate. Let $n(C)$ be the number of pages crawled so far which satisfy the user defined predicate. Then, if N is the total number of pages which have been crawled so far, we have $n(C) = N \cdot P(C)$. The significance factor for the event C and condition Q_i is denoted by $S(C, Q_i)$ and is calculated as follows:

$$S(C, Q_i) = |(P(C|Q_i) - P(C))/(\sqrt{P(C) \cdot (1 - P(C))/n(C)})| \tag{6}$$

For some pre-defined significance threshold t, we now define the significant composite ratio to include only those terms which are in M, and for which $S(C, Q_i)$ is above this threshold. We use the process only on words which are present in M. There are two reasons for this: (1) The words which are not in M are often not statistically significant, because most words in the lexicon are not in M by default. (2) The scalability of the technique is affected by the use of an exceptionally large number of words. The interest ratio for content based learning is denoted by $I_c(C)$, and is calculated as the product of the interest ratios of the different words in any of the inlinking web pages:

$$I_c(C) = \pi_{i:i \in M, S(C,Q_i) \geq t} I(C, Q_i) \tag{7}$$

The value of t denotes the number of standard deviations by which the presence is greater than the mean for the word to be useful. Under the assumption of normally distributed data, a value of $t = 2$ results in about 95% level of statistical significance. Therefore, we chose the value of $t = 2$ consistently in all results tested.

The method discussed above can easily be extended to the case of URL token based learning. The tokens contained inside a Universal Resource Locator (URL) may carry valuable information about the predicate-satisfaction behavior of the web page. The process discussed above for the content of the URL can also be applied to the tokens in the URL. For example, a URL which contains the word “ski” in it is more likely to be a web page about skiing related information. Therefore we first apply the step of parsing the URL. In order to parse the URL into tokens, we use the “.” and “/” characters in the URL as the separators. We define the event R_i to be true when token i is present in the URL pointing to the candidate. As before, we assume that the event that the candidate satisfies the predicate is denoted by C. The interest ratio for the event C given R_i is denoted by $I(C, R_i)$. The process of actually calculating the interest ratio is exactly analogous to the case of content based learning. Therefore, we will omit the details of this aspect of the learning process. We will denote the composite

In link based learning, we exploit the short range topical locality on the web. This is somewhat similar to the focused crawler discussed in [9]. While the significance of such link based information may vary from predicate to predicate, the intelligent crawler tries to learn the significance of link based information during the crawl itself. This significance is learned by maintaining and updating statistical information about short-range topical locality during the crawl itself. Thus, if the predicate shows considerable short-range locality, the crawler would learn this and use it effectively. Consider, for example, when the crawler has collected about 10000 URLS and a fraction of $P(C) = 0.1$ of them of them satisfy a given predicate. If the linkage structure were completely random, then the expected fraction of links for which both the source and destination web page satisfy the predicate is given by 1%. In reality, because of the short range topic locality discussed in [9], this number may be much higher and is equal to $f_1 = 7\%$. The corresponding interest ratio is is given by $0.07/0.01 = 7$. Since this is greater than 1, it implies a greater degree of short range topic locality than can be justified by random behavior. In Table 1, we illustrate the different cases for a link encountered by the crawler for which both the inlinking and linked-to web page have already been crawled. The four possible cases for the pages are illustrated in the first column of the Table. The second column illustrates the expected proportion of web pages belonging to each class, if the linkage structure of the web were completely random. At the same time, we continue

Table 1: Topic Locality Learning Information

Type of link	Expected	Actual
Pred- Pred	$P(C) \cdot P(C)$	f_1
Pred- Non-Pred	$P(C) \cdot (1 - P(C))$	f_2
Non-Pred- Pred	$P(C) \cdot (1 - P(C))$	f_3
Non-Pred- Non-Pred	$(1 - P(C)) \cdot (1 - P(C))$	f_4

to collect information about the actual number of each of the four kinds of links encountered. The corresponding fractions are illustrated in Table 1.

Now, consider a web page which is pointed to by k other web pages, m of which satisfy the predicate, and $k - m$ of which do not. (We assume that these k pages have already been crawled; therefore we can use the corresponding information about their predicate satisfaction; those inlinking pages to a candidate which have not yet been crawled are ignored in the calculation.) Then, for each of the m web pages which satisfy the predicate, the corresponding interest ratio is given by $p = f_1/(P(C) * P(C)$. Similarly, for each of the $k-m$ web pages which do not satisfy the predicate, the corresponding interest ratio is given by $q = f_3/(P(C) \cdot (1 - P(C))$. Then, the final interest ration $I_l(C)$ is given by $p^m \cdot q^{k-m}$.

Finally, the sibling based interest ratio is based on the idea that a candidate is more likely to satisfy a predicate if many of its siblings also satisfy it. (As in [20], a parent that has many children which satisfy the predicate is likely a hub and therefore a good place to find relevant resources.) For instance, consider a candidate that has 15 (already) visited siblings of which 9 satisfy the predicate. If the web were random, and if $P(C) = 0.1$, the number of siblings we expect to satisfy the predicate is $15 \cdot P(C) = 1.5$. Since a higher number of siblings satisfy the predicate (i.e. 9>1.5) , this is indicative that one or more parents might be a hub, and this increases the probability of the candidate satisfying the predicate.

To compute an interest-ratio based on this observation, we used the following rule: If s is the number of siblings that satisfy the predicate, and e the expected under the random assumption, then when $s/e > 1$ we have positive evidence that the candidate will satisfy the predicate as well. (Siblings that have not yet been visited are ignored, since we don't know whether they satisfy the predicate.) In the example above, the interest ratio for the candidate is thus 9/1.5=6, which suggests that the candidate is likely to satisfy the predicate. The sibling based interest ratio is denoted by $I_s(C)$.

Once the individual interest ratios have been computed, an aggregate interest ratio can be computed as a (weighted) product of the interest ratios for each of the individual factors. Equivalently, we can combine the preferences by summing the weighted logarithms of the individual factors.

$$PriorityValue = \log(I_c(C)) + \log(I_u(C)) + \log(I_l(C)) + \log(I_s(C))$$

If desired, it is also possible to use weights in order to vary the importance of the different factors.

3.1 Utilizing User Experiences in Resource Discovery

In this section, we will discuss the statistical model which is used to connect the user behavior with the predicate satisfaction probability of the candidate web pages. As in the previous case, the learning set $\mathcal{K}$ maintains the set of probabilities which indicate the user behavior during the crawling process. We note that several kinds of information about the user behavior may be relevant in determining whether a web page is relevant to the crawl:

- **Access Frequency Behavior:** Since users that have accessed web pages belonging to a particular predicate are also more likely to access other web pages belonging to the predicate, this is an important factor in determining the probability that a given candidate page will belong to the crawl topic.
- **Signature Features:** A signature feature is described as any characteristic of a web page such as content, vocabulary, or any other characteristic of a web page. Such signatures are often useful in identifying aspects which the raw frequency counts cannot provide.
- **Temporal Patterns of Users:** A set of accesses of web pages are likely to create similar accesses in the near future.

In order to calculate the priorities with the access frequency, we compute the likelihood that the frequency distribution of user accesses makes it more likely for a candidate web page to satisfy the predicate. In order to understand this point a little better, let us consider the following case. Suppose that we are searching for web pages on online malls. Let us assume that only 0.1% of the pages on the web correspond to this particular predicate. However, it may happen that the percentage of web pages belonging to online malls accessed by a user is over 10%. In such a case, it is clear that the user is favorably disposed to accessing web pages on this topic. If a given candidate web page has been accessed by many such users that are favorably disposed to the topic of online malls, then it may be useful to crawl the corresponding web page.

In order to develop the machinery necessary for the model, we will introduce some notations and terminology. Let N be total number of web pages crawled so far. Let U be the event that a crawled web page satisfies the user defined predicate. For a *candidate page* which is about to be crawled, the value of $P(U)$ is the probability that the web page will indeed satisfy the user-defined predicate. The value of $P(U)$ can be estimated by the fraction of web pages already crawled which satisfy the user defined predicate.

We will estimate the probability that a web page belongs to a given predicate U, given the fact that the web page has been crawled by user i. We shall denote the event that the person i has accessed the web page by R_i. Therefore, the predicate satisfaction probability is given by $P(U|R_i) = P(U \cap R_i)/P(R_i)$. We note that when the person i is topically inclined towards accessing web pages that belong to the predicate, then the value of $P(U|R_i)$ is greater than $P(U)$. Correspondingly, we define the interest ratio of predicate satisfaction as follows:

$$I^B(U|R_i) = P(U|R_i)/P(U) \tag{8}$$

We note that an interest ratio larger than one indicates that the person i is significantly more interested in the predicate than the average interest level of users in the predicate. The higher the interest ratio, the greater the topical affinity of the user i to the predicate. Similarly, an interest ratio less than one indicates a negative propensity of the user for the predicate. Now, let us consider a web page which has been accessed by the users $i_1 \dots i_k$. Then, a simple definition of the cumulative interest ratio $I(U|R_{i_1}, \dots R_{i_k})$ is the product of the individual interest ratios for each of the users. Therefore, we have:

$$I^B(U|R_{i_1} \dots R_{i_k}) = \pi_{j=1}^{k} I(U|R_{i_j}) \tag{9}$$

The above definition treats all users in a uniform way in the computation of the interest ratio. However, not all interest ratios are equally valuable in determining the value of a user to the crawling process. This is because we need a way to filter out those users whose access behavior varies from average behavior only because of random variations. In order to measure the significance of an interest ratio, we use the following computation:

$$T(U, R_{i_j}) = \frac{|P(U|R_{i_j}) - P(U)|}{\sqrt{P(U) \cdot (1 - P(U))/N}} \tag{10}$$

We note that the denominator of the above expression is the standard deviation of the average of N independent identically distributed bernoulli random variables, each with success probability $P(U)$. The numerator is the difference between the conditional probability of satisfaction and the unconditional probability. The higher this value, the greater the likelihood that the event R_{i_j} is indeed relevant to the predicate. We note that this value of $T(U, R_{i_j})$ is the significance factor which indicates the number of standard deviations by which the predicate satisfaction of U is larger than the average if the user i_j has browsed that web page. In the computation of the interest ratio of a candidate page, we use only those users i_j for which $T(U, R_{i_j}) \geq t$ for some threshold[3] t.

We note that the nature of proxy traces is inherently sparse. As a result, in many cases, a single user may not access too many documents in a single trace. Therefore, a considerable amount of information in the trace can be broken up into *signatures* which are particular characteristics of different web pages. Such signatures may be chosen across the entire vocabulary of words (content), topical categories based on scans across the world wide web, or other relevant characteristics of web pages.

The use of such characteristics is of tremendous value if the signatures are highly correlated with the predicate. For example, if the document vocabulary is used as the relevant signature, then even though there many be billions of documents across the world wide web, the number[4] of relevant words is only of the order of a hundred thousand or so. Therefore, it is easier to find sufficient overlap of signatures across users in the crawling process. This overlap helps in reducing the feature space sufficiently, so that it is possible to determine interesting patterns of user behavior which cannot be discerned only by using the patterns in terms of the individual web pages. The process for

[3] For the purpose of this paper, we will use a threshold of $t = 2$ standard deviations in order to make this determination.

[4] This assumes that the documents are in English and stop-words/rare words have been removed.

finding the signature specific interest ratio $I^{SF}(U|R_{i_1} \ldots R_{i_k})$ is quite analogous to that of finding the user-specific interest ratio. More details on the method of finding the signature specific interest ratio may be found in [1].

The web pages accessed by a given user often show considerable temporal locality. This is because the browsing behavior of a user in a given session is not just random but is highly correlated in terms of the topical subject matter. Often users that browse web pages belonging to a particular topic are likely to browse similar topics in the near future. This information can be leveraged in order to improve the quality of the crawl.

In order to model this behavior, we will define the concept of *temporal locality* region of a predicate U by $\mathcal{TL}(U)$. To do so, we will first define the temporal locality of each web page access A. The temporal locality of a web page access A is denoted by $TLR(A)$ and is the n pages accessed either strictly before or strictly after A by the *same* user, but not including A. Now let us say that $A_1 \ldots A_m$ be the set of accesses which are known to belong to the predicate U. Then the temporal locality of the predicate U which is denoted by $\mathcal{TL}(U)$ is defined as follows:

$$\mathcal{TL}(U) = \cup_{i=1}^{k} TLR(A_i) \tag{11}$$

Let f_1 be the fraction of web pages belonging to $\mathcal{TL}(U)$ which also satisfy the predicate. Furthermore, let f_2 be the fraction of web pages *outside* $\mathcal{TL}(U)$ which satisfy the predicate. Then, the overall interest ratio for a web page belonging to $\mathcal{TL}(U)$, is given by:

$$I^{TL}(U) = f_1/P(U) \tag{12}$$

Similarly, the Interest Ratio for a web page which does *not* belong to the temporal locality of U is given by:

$$I^{TL}(U) = f_2/P(U) \tag{13}$$

We note that in most cases, the value of f_1 is larger than $P(U)$, whereas the value of f_2 is smaller than $P(U)$. Correspondingly, the interest ratios are larger and smaller than one respectively. For a given web page, we check whether or not it belongs to the temporal locality of a web page which satisfies the predicate. If it does, then the corresponding interest ratio is incorporated in the computation of the importance of that predicate.

The different factors discussed above can be utilized in order to create a composite interest ratio which measures the value of the different factors in the learning process. We define the composite interest ratio as the product of the interest ratios contributed by the different factors. Therefore, the combined interest ratio $I^C(U)$ for a web page which has been accessed by users $i_1 \ldots i_k$ is given by:

$$I^C(U) = I^{TL}(U) \cdot I^{SF}(U|R_{i_1} \ldots R_{i_k}) \cdot I^B(U|R_{i_1} \ldots R_{i_k}) \tag{14}$$

This composite interest ratio reflects the overall predicate satisfaction probability of a web page based on its characteristics.

4 On the Merits of Combining User and Linkage Information for Topical Resource Discovery

In the previous sections, we discussed methods for crawling with the utilization of user and linkage information. In this section, we will discuss methods for combining the two, It turns out that a method which combines user and linkage information is much more effective than one which uses only one of the two. In order to achieve this, the system can find all the web pages which are linked to by a predicate-satisfying web page and added to the candidate list. As a result, the list $Cand$ becomes a combination of candidates for which we may have either web log access information or linkage information. In addition, we need to make changes to the process of calculation of priorities. In [4], we discussed the computation of interest ratios which are analogous to those discussed in this paper using purely linkage based information. Let $I^L(U)$ be the interest ratios computed for a candidate page using the method in [4]. Let $I^C(U)$ be the corresponding user-based interest ratio. As discussed earlier, not all web pages have both linkage based and user-based information associated with them. Therefore, when such information is not available, the corresponding value for $I^L(U)$ or $I^C(U)$ is set to one. Since the URL for a candidate page is discovered either from the content of a web page or from the web log, at least one of these interest ratios can be calculated effectively. The overall interest ratio is defined by $I^C(U) \cdot I^L(U)$. We will see that the combination of the linkage and user-based systems is very powerful in practice in improving the quality of the crawl. This is because the user behavior quickly identifies the most popularly visited resources which often link to a large number of closely related pages. These pages are added to the candidate list. The addition of such candidates facilitates the discovery of some of those rarely accessed web pages which may not be found in the traces. These in turn help in the discovery of more topically inclined users and vice-versa. In other words, some resources can be more easily crawled from the user information, whereas others require linkage information. These interest ratios thus act in a complimentary way in finding promising candidates during the resource discovery process. As a result, the system works better than one which is developed using either purely user or purely linkage information.

The system was implemented on an AIX 4.1.4 system with 100 MB of main memory and 2GB SCSI drive. The results were tested using two kinds of traces:

- **Squid Proxy Traces:** These traces are available from [25], and reflect the web page access behavior across a wide spectrum of users on the internet. Each of these traces contained between 100,000 to 1000,000 user accesses.
- **IBM Proxy Traces:** These traces reflect the access behavior of IBM employees and are stored at the IBM Raleigh site. Each of these traces contain about 100,000 accesses, and are each based on a single day of access behavior.

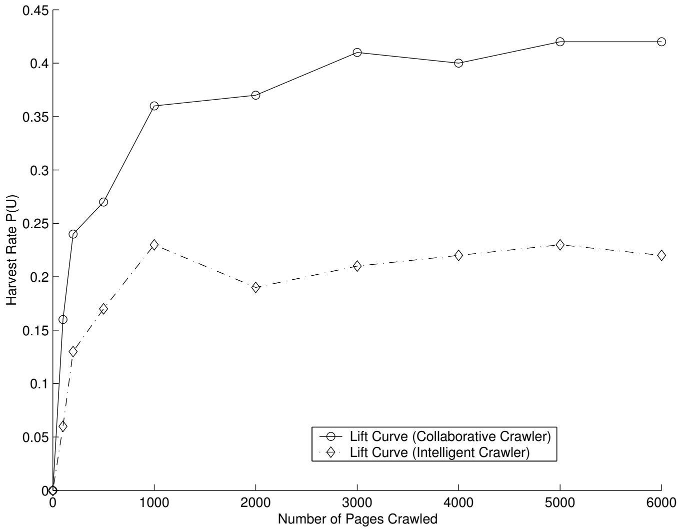

Figure 2: Performance of Collaborative and Intelligent Crawler (Predicate is category "SPORTS")

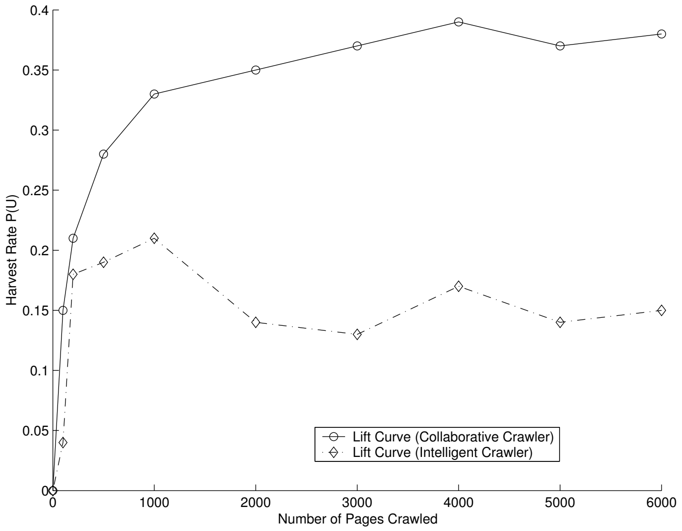

Figure 3: Performance of Collaborative and Intelligent Crawler (Predicate is category"ARTS")

The performance of the crawler was characterized by using the harvest rate $P(U)$, which was the percentage of web pages crawled that satisfy the predicate. In order to illustrate our results, we will present the *lift curve* which illustrates the gradual improvement of the harvest rate with the number of URLs crawled. Initially, the crawling system is slow to find relevant web pages, but as the crawl progresses, it gradually learns the propensities of the users in terms of their predicate satisfaction probability. This improves the computation of the interest ratios of the candidate web pages over time. Correspondingly, the percentage of the candidates crawled belonging to the predicate increases as well.

First, we present the results which illustrate the comparisons between a purely linkage based crawler (intelligent crawler) and a user-based crawler (collaborative crawler). In Figure 2, we have illustrated an example of a lift curve which shows the crawling performance of the system over time. In this case, the predicate is the web pages belonging to category "SPORTS" as predicted by the classifier discussed in [3]. The initial behavior of the crawler system is random, but as it encounters web pages belonging to the predicate, the performance quickly improves. In the same chart, we have also illustrated the performance of the intelligent crawler algorithm from [4]. The intelligent crawler algorithm was run using five different starting points and the *best* of these lift curves (as indicated by the value of $P(U)$ at the end of the crawl) was used. It is interesting to see that even a single execution of the collaborative crawler was significantly more effective than even the best of five executions of the intelligent crawler.

We have illustrated the performance of the collaborative crawler in Figure 3. In this case, the predicate is the category "ARTS". As in the previous case, we have plotted the curve for the intelligent crawler for the best of five executions of the algorithm. Again, the collaborative crawler has a much greater harvest rate than the intelligent crawler. Another interesting difference between the collaborative and intelligent crawler is that the harvest rate of the intelligent crawler varies significantly over the crawl. This tends to indicate that the information available in web page links may not be very robust in always providing considerably effective information for the crawling process.

It is useful to measure the behavior of the crawler when linkage and user information are combined. Even though the information from the web traces turns out to be valuable in finding the most popular resources, its performance can be improved further with the incorporation of linkage information. This helps in finding resources which are not quite as popular, but are often relevant to the predicate. In Figures 4 and 5, we have illustrated the harvest rate of the collaborative and intelligent crawlers, together with a crawler which combines the two pieces of information. In Figure 4, the predicate corresponds to the category "COMPUTERS" and containing the keyword "viruses".

In Figure 5, the predicate corresponds to the AUTOMOTIVE category and containing the keywords "Toyota Corolla". In both cases, the combined crawler performs better than a crawler employing purely trace-based or linkage information. This behavior seemed to be consistent over a wide variety of experiments that we performed. This behavior seems to be a little surprising, since one would normally expect that the lift curve for the combined system ought to be the average of the two individual systems.

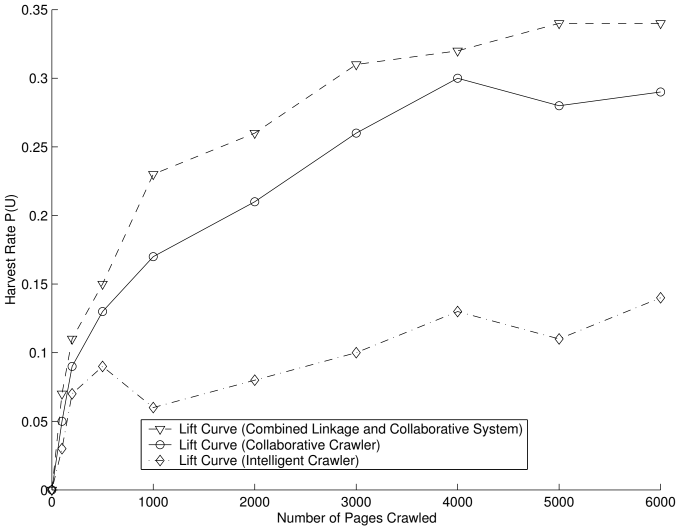

Figure 4: Combining trace-based and linkage information (Predicate is category "COMPUTERS" containing keyword "viruses")

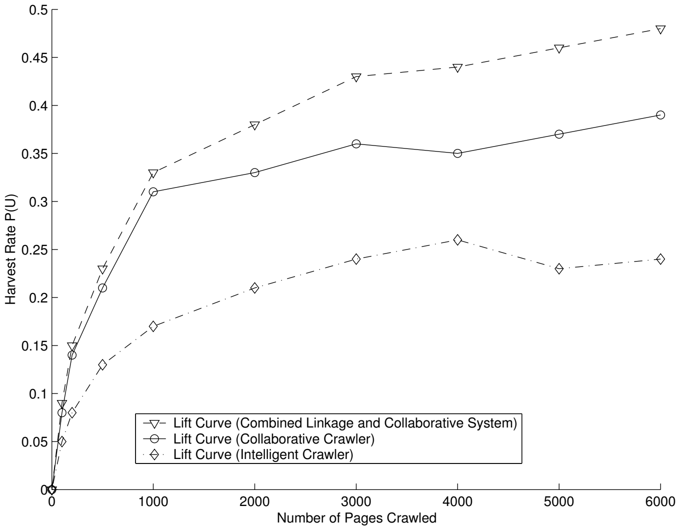

Figure 5: Combining trace-based and linkage information (Predicate is category "AUTOMOTIVE" containing keywords "Toyota Corolla")

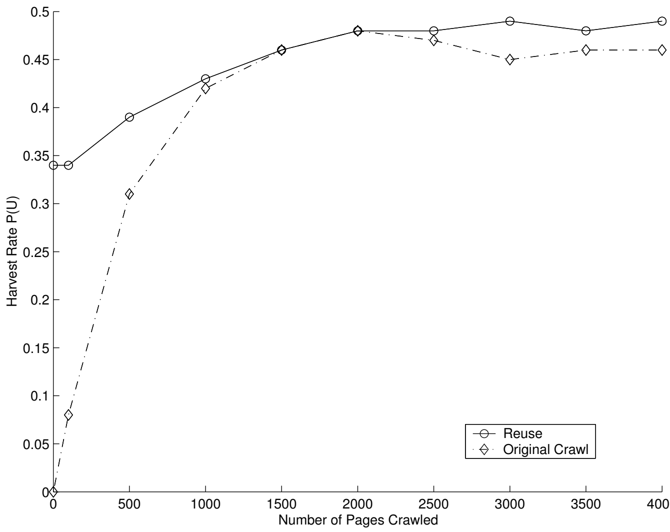

Figure 6: Reuse of crawler learning information (Predicate is category "SPORTS" containing keyword "basketball")

On examining the web pages obtained from the crawl, we discovered that both the trace-based and the linkage-based systems discovered web pages which the other could not discover. Correspondingly, many web pages had a high interest ratio using the linkage based computation, whereas others had a high interest ratio using the trace based computation. The noise-reduction methodology built into the interest ratio computation ensures that at each instant in time, the best criterion is used. In many cases, the web pages discovered by the linkage based methodology would provide information about new users that often access predicate specific web pages. In other cases, the web pages discovered using traces provided valuable hyperlinks to other predicate-relevant web pages. This complementary relationship resulted in a system which was better than either purely trace-based or linkage-based systems.

4.1 Reuse of crawler learning information

We note that the crawler system discussed in this paper learns from the user behavior in the web traces, which are usually available in large quantities on a daily basis. As the set of pages available on the web change over time, it may be desirable to perform the same query repeatedly in order to discover the most recent resources. It is possible to leverage the information gained from a given crawl in order to improve the effectiveness of subsequent crawls. This is because many of the user access patterns and interests are consistent over time, and provide valuable hints to the resource discovery process. In Figure 6, we have illustrated the performance of the collaborative crawler by using two traces from the IBM proxy server. In this case, the predicate corresponds to the category "SPORTS" containing the keyword "basketball". The traces were separated by a period of about 6 months, as a result of which many of the user domains and web pages in one trace were not present in the other. We note that the process of reusing learned information from a given crawl is not specific to the collaborative crawling method, but can also be easily extended to linkage based

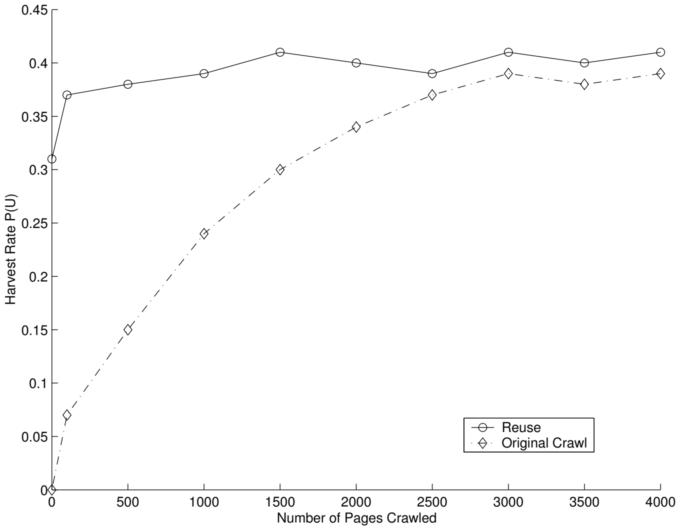

Figure 7: Reuse of crawler learning information (Predicate is category "ARTS" containing keyword "gallery")

methods.

However, even this partial relevance was sufficient to significantly improve the quality of the web pages obtained over a crawl which started from scratch. The crawl started with the affinities of the users from the previous trace in order to decide the priority of the web pages to be crawled.

In Figure 7, we have illustrated similar results for the predicate corresponding to the category "ARTS" and containing the keyword "gallery". Again, the reuse of crawler learning information significantly improves the performance of the crawl. It is clear from Figures 6 and 7, that when the information from one trace is used in the second, the performance of the crawler is improved substantially during the initial learning phase. This property is valuable in building an incremental crawler in which similar kinds of resources often have to be repeatedly crawled over different periods of time. As discussed in recent research [19], such queries are useful because of the rapidly evolving nature of the world wide web. The system of this paper provides a useful technique to resolve such queries effectively.

5 Conclusions and Summary

In this paper we discussed a number of learning techniques for topical resource discovery. Specifically, we discussed the collaborative crawler and the intelligent crawler which are techniques for utilizing learning methods for resource discovery. The results illustrate that while user-experiences provide a richer ability to perform the learning, the combination of user and linkage based methods tend to be more effective than either. This is because the combination is able to capture those kinds of web pages that either of the techniques cannot do alone. We note that the learning system discussed in this paper is also applicable to an environment in which the web pages evolve over time. In such cases, the system can leverage the information learned about user behavior in a given crawl in order to improve the effectiveness of subsequent crawls. We also showed that the collaborative crawling system can be combined with a linkage based system in order to create a crawler which is more effective than either a purely linkage or trace-based system.

References

[1] C. C. Aggarwal. Collaborative Crawling: Mining User Experiences for Topical Resource Discovery. *Proceedings of the KDD Conference*, 2002.

[2] C. C. Aggarwal, J. L. Wolf, K.-L. Wu, P. S. Yu. Horting Hatches an Egg: A New Graph Theoretical Approach to Collaborative Filtering. *Proceedings of the ACM SIGKDD Conference*, 1999.

[3] C. C. Aggarwal, S. C. Gates, P. S. Yu. On the merits of using supervised clustering for building categorization systems. *Proceedings of the ACM SIGKDD Conference*, 1999.

[4] C. C. Aggarwal, F. Al-Garawi, P. Yu. Intelligent Crawling on the World Wide Web with Arbitrary Predicates. *Proceedings of the WWW Conference*, 2001.

[5] K. Bharat, M. Henzinger. Improved Algorithms for Topic Distillation in a Hyperlinked Environment. *Proceedings of the ACM SIGIR Conference*, 1998.

[6] J. Carriere, R. Kazman. Searching and Visualizing the Web through Connectivity. *Proceedings of the World Wide Web Conference*, pages 701-711, 1997.

[7] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, S. Rajagopalan. Automatic Resource Compilation by Analyzing Hyperlink Structure and Associated Text. *Special Issue of the Seventh World Wide Web Conference*, 30(1-7), April 1998.

[8] S. Chakrabarti, B. Dom, S. Ravi Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, J. Kleinberg. Mining the Web's link structure. *IEEE Computer*, 32(8):60-67, August 1999.

[9] S. Chakrabarti, M. van den Berg, B. Dom. Focussed Crawling: A New Approach to Topic Specific Resource Discovery. *Proceedings of the Eighth World Wide Web Conference*, pages 545-562, 1999.

[10] S. Chakrabarti, M. van den Berg, B. Dom. Distributed Hypertext Resource Discovery through Examples. *Proceedings of the VLDB Conference*, 1999.

[11] S. Chakrabarti. Integrating the Document Object Model with Hyperlinks for Enhanced Topic Distillation and Information Extraction. *Proceedings of the WWW Conference*, 2001.

[12] M. S. Chen, J. S. Park, P. S. Yu. Data Mining for Path Traversal Patterns in a Web Environment. *ICDCS Conference*, 1996.

[13] M. Diligenti et al. Focused Crawling Using Context Graphs. *Proceedings of the VLDB Conference*, 2000.

[14] J. Ding, L. Gravano, N. Shivakumar. Computing Geographical Scopes of Web Resources.. *Proceedings of the VLDB Conference*, 2000.

[15] J. Edwards, K. McCurley, J. Tomlin. An Adaptive Model for Optimizing Performance of an Incremental Web Crawler. *Proceedings of the World Wide Web Conference*, 2001.

[16] R. Lempel, S. Moran. The Stochastic Approach for Link-Structure Analysis (SALSA) and the TKC effect. *WWW9 Conference*, pages 387-401, 2000.

[17] M. Najork, J. Wiener. Breadth-First Search Yields High-Quality Web Pages. *Proceedings of the World Wide Web Conference*, 2001.

[18] Z. Bar-Yossef, A. Berg, S. Chein, J. Fakcharoenphol, D. Witz. Approximating Aggregate Queries about Web Pages via Random Walks. *Proceedings of the VLDB Conference*, 2000.

[19] J. Cho, H. Garcia-Molina. The Evolution of the Web and Implications for an Incremental Crawler. *Proceedings of the VLDB Conference*, 2000.

[20] J. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Proceedings of the ACM-SIAM Symposium of Discrete Algorithms*, 1998.

[21] R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins. Trawling the Web for Emerging Cyber-communities. *Proceedings of the World Wide Web Conference*, 1999.

[22] S. Mukherjea. WTMS: A System for Collecting and Analyzing Topic-Specific Web Information. *Proceedings of the World Wide Web Conference*, 2000.

[23] S. Raghavan, H. Garcia-Molina. Crawling the hidden web. *Proceedings of the VLDB Conference*, 2001.

[24] A. Rousskov, V. Solviev. On Performance of Caching Proxies.
http://www.cs.ndsu.nodak.edu/ rousskov/research/cache/squid/profiling/papers/

[25] ftp://ircache.nlanr.net/Traces/

[26] U. Shardanand, P. Maes. Social Information Filtering: Algorithms for Automating Word of Mouth *Proceedings of CHI '95*, Denver CO, pp. 210-217, 1995.

[27] R. Srikant, Y. Yang. Mining Web Logs to Improve Website Organization. *ACM KDD Conference*, 2001.

[28] http://www.yahoo.com

[29] http://www.altavista.com

[30] http://www.lycos.com