# Collaborative Crawling: Mining User Experiences for Topical Resource Discovery

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

## ABSTRACT

The rapid growth of the world wide web had made the problem of topic specific resource discovery an important one in recent years. In this problem, it is desired to find web pages which satisfy a predicate specified by the user. Such a predicate could be a keyword query, a topical query, or some arbitrary contraint. Several techniques such as focussed crawling and intelligent crawling have recently been proposed for topic specific resource discovery. All these crawlers are *linkage based*, since they use the hyperlink behavior in order to perform resource discovery. Recent studies have shown that the topical correlations in hyperlinks are quite noisy and may not always show the consistency necessary for a reliable resource discovery process. In this paper, we will approach the problem of resource discovery from an entirely different perspective; we will mine the significant browsing patterns of world wide web *users* in order to model the likelihood of web pages belonging to a specified predicate. This user behavior can be mined from the freely available traces of large public domain proxies on the world wide web. We refer to this technique as *collaborative crawling* because it mines the collective user experiences in order to find topical resources. Such a strategy is extremely effective because the topical consistency in world wide web browsing patterns turns out to very reliable. In addition, the user-centered crawling system can be combined with linkage based systems to create an overall system which works more effectively than a system based purely on either user behavior or hyperlinks.

## 1. INTRODUCTION

With the rapid growth of the world wide web, the problem of resource collection on the world wide web has become very relevant in the past few years. Users may often wish to search or index collections of documents based on topical or keyword queries. Consequently, a number of search engine technologies such as *Lycos* and *AltaVista* have flourished in recent years.

A significant method proposed recently for automated re-source discovery is the technique of *focused crawling* [4]. The essential idea in focused crawling is that there is a short range topical locality on the web. This locality may be used in order to design effective techniques for resource discovery by starting at a few well chosen points and maintaining the crawler within the ranges of these known topics. A recent paper discusses the application of *learning* methodologies [3] in order to improve the effectiveness of the crawl. While the focussed crawling technique is designed for finding web pages belonging to one of the classes from a hierarchically arranged topical collection, the intelligent crawling technique is capable of more flexible and arbitrary *predicates* such as a combination of different kinds of topical queries, keyword queries or other constraints on the content or meta-information about the web page such as its URL domain.

Both the focussed crawling and intelligent crawling methods [3, 4] use only *linkage based information* in order to find topical resources. In the former case, the linkage based information is used directly under the assumption that web pages show topical locality, whereas in the latter case, a learning process is used to identify the most suitable web pages. It is this learning process of the intelligent crawling technique which provides it with the flexibility and robustness necessary to collect web pages belonging to arbitrary predicates.

In many cases, links are quite noisy in terms of topical locality. Such links may correspond to banners, advertisements and other content which do not carry specific information about resource discovery. Often advertisements, banners and such content dominate the resource structure of many web sites even though they are only occasionally browsed by users. In addition, since the web continues to grow rapidly in a unstructured fashion through the efforts of millions of non-collaborating users, it is difficult for a single linkage model to completely capture the complex and unstructured variation in hyperlink patterns.

This paper will therefore diverge significantly from linkage-based methodologies, and instead concentrate on utilizing the browsing patterns of users on the web in order to mine significant patterns. We will achieve this goal by utilizing the logs of user access behavior on public domain proxies. An example of such a *public-domain proxy* is the Squid [5] proxy cache, which is a set of large hierarchical caches on the web which is capable of improving the latency performances of different geographical regions. The access patterns of these systems are representative of a large sample of world wide web clients.

The utilization of user behavior turns out to be quite use-

ful in practice, since even for specific topics there are often hundreds of thousands of documents, and the size of these collections continues to grow and change every day. The browsing behavior of the users often reflects the changing trends on the world wide web quite effectively, since the user behavior mirrors the changes in the world wide web structure quite well. On the other hand, hyperlinks are frequently outdated and do not adapt quickly to changes in the documents on the web. Consequently, a linkage based system may often spend a fraction of its time trying to access web pages which no longer exist, whereas a recent trace will usually contain only those web pages which are accessible. (Inaccessible web pages would typically be contained in the error log.)

This paper is organized as follows. In the next section, we discuss the general framework for the crawler, and the factors which may be used for the purpose of self-learning. In section 3, we will provide an overview of the statistical model which is used for the purpose of the crawling. Section 4 discusses the empirical results, and the conclusions and summary are discussed in section 5.

## 2. COLLABORATIVE CRAWLING

The collaborative crawler assumes that multiple users browse the web through proxy gateways, which generate web traces containing the access information of the users. We note that such gateways may often serve thousands of users; consequently, the web logs generated are often quite huge even on a daily basis. The collaborative crawling method is implemented as a learning system which identifies the importance of different users during the crawling process in order to bias the crawl towards those web pages which are most likely to belong to the predicate. Specifically the characteristics of the web pages together with the browsing behavior of the users are learned by the collaborative crawler in order to estimate their likelihood of satisfying the predicate.

An important aspect of most web logs is that they often do not contain individual user information but simply the domain names of the accesses. Since some of the domain names are actually proxy accesses from large *groups* of users rather than individuals, this reduces the level of granularity of the available data. Some simple processing can however improve the quality of the trace:

• All source IP addresses which occurred in a larger number of accesses than a given threshold[1] were removed. This filtered out a large fraction of group accesses. • The above criterion was also used for shorter periods of time such as a 1 minute interval. If there were more than a certain number[2] of accesses per unit of time from the same source IP address, then this was indicative of group behavior.

The above preprocessing steps would still result in many IP addresses corresponding to groups of users rather than individuals. However, our primary aim is to discover resources rather than user behavior, and as long as there was considerable correlation between the source IP address and

---

[1] We picked a threshold depending upon the number of days that it represented. Our threshold was 2000 accesses per day. The assumption was that only large groups of people could make so many accesses in a given day.

[2] We used 100 accesses per minute as the threshold. The reason for this high threshold is that multiple access entries in the trace may actually correspond to the different parts (images, frames and html) of the same user access.

access content, we did not consider this to be a serious limitation. For the rest of the paper, we will still refer to a source IP address as a user for the purpose of conceptual abstraction.

The crawler is implemented as an iterative search algorithm which always maintains a set of *candidate web pages* together with corresponding *priorities*. These priorities are determined by relating the statistical behavior of the crawled web pages to the users that have accessed them. Each time a web page is crawled, it is determined whether or not it satisfies the predicate. The statistical behavior of the users that have accessed this web page is utilized to update the priorities of the candidate web pages. The crawler keeps track of the nodes which it has already visited, as well as a potential list of *candidates*. A web page is said to be a candidate when it has not yet been crawled, but some other web page which has been accessed by the same user has already been crawled. The access behavior of this user is incorporated into the statistical model used to decide whether that candidate is likely to satisfy the predicate. This information may be used by the crawler to decide the order in which it visits web pages.

Thus, at each point the crawler maintains candidate nodes which it is likely to crawl and keeps calculating the priorities of the nodes using the information about the user access patterns of different web pages. This statistical information which relates the user behavior to the predicate may be used to learn the nature of the relationship between web pages and users.

## 2.1 Modeling User Experiences

In this section, we will discuss the statistical model which is used to keep track of the crawling behavior. Since the crawler system uses the gateway logs for resource discovery, it is an interesting question as to how the user behavior may be converted into priorities for visiting web pages. It is clear that we need to build a statistical model which connects the user behavior to the predicate satisfaction probability of the candidate web pages.

We assume that the set of features used in the learning process are stored in the set $\mathcal{K}$. This 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:

• **The access frequency behavior of the users for different web pages:** Users that have accessed web pages belonging to a given predicate are more likely to access other web pages belonging to the predicate.

• **The access frequency behavior of the users with the help of *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, In many cases, the access behavior of users in terms of signature topics yields information which cannot be revealed by their access behavior from individual web pages. This is because users may often access web pages belonging to combinations of topics rather than individual combinations of web pages.

• **The temporal patterns of behavior of the different users across different web pages:** This means that users that have just accessed web pages belonging to the predicate are more likely to access the same in the immediate future. Thus, by taking the order of accesses into

account, considerable amount of useful information can be mined.

## 2.2 The Frequency Bias Factor

In this section, we discuss the probabilistic model for utilizing the access frequencies of users in computing priorities. In order to calculate the priorities, 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. We would like to quantify the level of favorable disposition of the candidate page by a value that we will refer to as the *interest ratio*.

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:

$$I^B(U|R_i) = P(U|R_i)/P(U) \tag{1}$$

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 $i_1 \ldots i_k$. Then, a simple definition of the cumulative interest ratio $I(U|R_{i_1}, \ldots R_{i_k})$ is the product of the individual interest ratios for each of the users. Therefore, we have:

$$I^B(U|R_{i_1} \ldots R_{i_k}) = \pi_{j=1}^k I(U|R_{i_j}) \tag{2}$$

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:

$$T(U, R_{i_j}) = \frac{|P(U|R_{i_j}) - P(U)|}{\sqrt{P(U) \cdot (1 - P(U))/N}} \tag{3}$$

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] $t$.

## 2.3 Use of Signature Characteristics and Browsing Content

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.

In order to formalize the above concept, we will assume that $L_{ij}$ is the event that the signature $j$ has been accessed by a given user $i$. Thus, for example, when the event $j$ corresponds to the word $j$ being accessed, it means that the document corresponding to word $j$ is accessed at least once by the user $i$. We define $V_i$ as the event that a document belonging to the predicate has been accessed at least once by the user $i$. We note that the ratio $P(V_i|L_{ij})/P(V_i)$ provides the interest ratio that the access of feature $j$ is interesting for the user $i$. However, we are not just interested in the behavior of a single feature, but that of all the features. Therefore, we denote the signature specific interest ratio of the user $i$ by $SF_i$.

$$SF_i = E_{\text{All Features } j}[P(V_i|L_{ij})/P(V_i)] \tag{4}$$

Here $E[\cdot]$ denotes the expected value over all the different

---

[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 stopwords/rare words have been removed.

features. As in the previous case, let us assume that a given candidate page has been browsed by the users $i_1 \ldots i_k$. The corresponding events are denoted by $R_{i_1} \ldots R_{i_k}$. We denote the event that the web page satisfies the predicate by $U$. Then, the cumulative interest ratio of predicate satisfaction, given that any of the signature characteristics inside it have been browsed by users $i_1 \ldots i_k$, is denoted by $I^{SF}(U|R_{i_1} \ldots R_{i_k})$ and is defined by the following expression:

$$I^{SF}(U|R_{i_1} \ldots R_{i_k}) = \pi_{j=1}^{k} SF_{i_j} \qquad (5)$$

Not all users may show a high propensity for or against a particular predicate. Consequently, we would like to filter out the noise from the above expression. Let $n_s$ be the total number of users for which signature specific interest ratios have been computed. We define the mean $\mu^{SF}$ and standard deviation $\sigma^{SF}$ of the feature-specific interest propensities:

$$\mu^{SF} = \sum_{i=1}^{n_s} SF_i / n_s \qquad (6)$$

$$\sigma^{SF} = \sqrt{\sum_{i=1}^{n_s} (SF_i - \mu^S)^2 / n_s} \qquad (7)$$

Therefore, we define the significance of each user $i$ as follows:

$$SFF(i) = |(SF_i - \mu^{SF})|/\sigma^{SF} \qquad (8)$$

Once the significance of each user $i$ has been determined, we can now use only those users whose significance is above a pre-defined threshold $t$ for the purpose of crawling. This reduces the noise effects in the computation of the interest ratio and ensures that only users that show an access behavior which is significantly related to the predicate are used.

### 2.3.1 An example of signature characteristics

As discussed above, a wide variety of signature characteristics can be used in order to facilitate the crawling process. One example of such a set of characteristics is the use of the topical categories available in the *Yahoo!* taxonomy. The characteristics in a web page are defined as the dominant classes in the taxonomy which are related to it. In order to achieve this goal, we implemented a classifier system similar to that described in [2]. This classifier system was capable of predicting the classes most related to a given web page. We found the $k_{max} = 5$ most closely related classes to the page using the nearest neighbor classifier of [2].

## 2.4 Use of Temporal Locality in Browsing

The web pages accessed by a given user often show considerable temporal locality. This is because the browing 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:

$$\mathcal{TL}(U) = \cup_{i=1}^{k} TLR(A_i) \qquad (9)$$

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:

$$I^{TL}(U) = f_1/P(U) \qquad (10)$$

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

$$I^{TL}(U) = f_2/P(U) \qquad (11)$$

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.

## 2.5 Combining the different factors

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:

$$I^C(U) = I^{TL}(U) \cdot I^{SF}(U|R_{i_1} \ldots R_{i_k}) \cdot I^B(U|R_{i_1} \ldots R_{i_k}) \qquad (12)$$

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

## 3. IMPLEMENTATION DETAILS

The overall sketch of the collaborative crawling algorithm is discussed in Figure 1. The method maintains a candidate list which is denoted by $Cand$ which keeps track of the web addresses that are to be crawled. In each iteration, the highest priority page $F$ of the candidate list $Cand$ is crawled, while the learning information and priorities are updated based on the user access behavior of $F$. These updated priorities are used in order to expand the list of potential candidates. The candidate page $F$ is deleted from $Cand$ after it has been crawled. The process of calculating the priorities is denoted by *CalculatePriorities*, whereas that of expanding the candidate list is denoted by *ExpandList*. This iterative process continues until the list $Cand$ becomes empty. In addition, it is assumed that at least *min-cand* web pages be crawled before the process terminates. If *min-cand* web pages have not yet been crawled and $Cand$ is empty, then the algorithm randomly samples the trace for web pages which are further added to the candidate list. This ensures

**Subroutine** *CalculatePriorities*(WebPage: $F$, Trace: $\mathcal{T}$,
    Learning Statistics: $K$);
**begin**
  Find all the users $\mathcal{P}(F)$ than have accessed $F$;
  Compute the priorities of all web pages $\mathcal{N}$ accessed by
  $\mathcal{P}(F)$ using the computation discussed in section 2;
  **return**($\mathcal{N}$);
**end**

**Subroutine** *ExpandList(CandidateList: Cand,*
    *NewCandidates: $\mathcal{N}$)*;
  { Add those candidates in $\mathcal{N}$ to $Cand$ which
  have priority above a user-defined threshold; }

**Algorithm** *CollabCrawler(Trace: $\mathcal{T}$, 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 $F$;
      if $F$ satisfies the predicate **then** save $F$;
      Update learning statistics $\mathcal{K}$;
      $\mathcal{N} = CalculatePriorities(F, \mathcal{T}, \mathcal{K})$;
      *ExpandList(Cand, $\mathcal{N}$)*;
      Delete $F$ from $Cand$;
      **if** $Cand$ is empty and at least *min-cand*
        candidates have not yet been crawled, then add
        a randomly sampled URL from $\mathcal{T}$ to $Cand$;
    **end**;
**end**

**Figure 1: The Collaborative Crawler Algorithm**

that at least *min-cand* pages have been crawled before the process terminates.

In the actual implementation, two hash tables were maintained, one of which contained information for the URLs (URL hashtable), whereas the other (User hashtable) contained information about the users accessing the URLs. For each user a list of their accesses together with time stamps were maintained in the user hashtable, whereas the list of the users accessing a URL together with timestamps were maintained in the URL hashtable. In addition, a number of statistical values on the access behavior of the user were maintained in the hash table. These values were required in the determination of the predicate access behavior of the user. These values are denoted by the self-learning information $\mathcal{K}$ which is collected by the crawler. The self learning information $\mathcal{K}$ contains the following values:
(1) Total number $N$ of URLs crawled.
(2) Total number $N_u$ of crawled pages satisfying predicate.
(3) Number $q_i$ of crawled pages accessed by user $i$.
(4) Number $q_i^u$ of web pages (satisfying the predicate) accessed by user $i$.
(5) Number of documents $r_{ij}$ accessed by user $i$ containing the signature $j$.
(6) Number of documents $r_{ij}^u$ (satisfying the predicate) accessed by user $i$ containing the signature $j$.
(7) Number of documents $s_1$ belonging to the temporal locality of $U$.
(8) Number of documents $s_1^u$ belonging to the temporal locality of $U$ which satisfy the predicate.
(9) Number of documents $s_2$ which do not belong to the

temporal locality of $U$.
(10) Number of documents $s_2^u$ not belonging to the temporal locality of $U$ which satisfy the predicate.

The quantities (1), (2), (7), (8), (9) and (10) are globally maintained in individual variables, whereas the quantities (3), (4), (5), and (6) are maintained for each user in the hash table. This information are used to estimate the key statistical parameters and probabilities as follows:
(1) $P(U) = N_u/N$
(2) $P(U|R_i) = q_i^u/q_i$
(3) $SF_i = \sum_{\text{All Features } j}(r_{ij}^u/r_{ij})/(\text{Number of Features})$
(4) $IR^U = s_1^u/s_1$
(5) $IR^{\overline{U}} = s_2^u/s_2$
We note that the above set of estimates suffice to compute all the interest ratios discussed in the previous section. However, it is time-consuming to calculate all the interest-ratios after each web page is crawled. In order to achieve better efficiency, the process of updating is executed in a lazy fashion by *CalculatePriorities*. The preference values are updated only for those users which have accessed the web page that was last crawled. Then, the interest ratios for the candidate web pages accessed by these users are updated. Even though this results in the interest ratios of a few web pages being stale, the web pages which are updated are the ones which have the maximum change in their priority values because of the last web page being crawled. In addition, a periodic process of updating the priorities is applied every few iterations in order to update these values. The procedure *CalculatePriorities* returns a list $\mathcal{N}$ of candidate URLs whose interest ratios were updated. All those interest ratios which are larger than 1 are added to the candidate list. This is achieved by the procedure *ExpandList*. After expansion of the list, the candidate page $F$ is removed from $Cand$. This procedure is repeated until at least *min-cand* web pages have been crawled and $Cand$ becomes empty. We note that one of the inputs to the algorithm is the starting seed set $S$. If topic specific resources in the trace are known, then these can be used as the starting seeds; alternatively the use of $S = \{\}$ results in the random sampling of *min-cand* pages from the trace as the default starting point.

## 3.1  Using linkage information

We note that even though this paper is tailored towards the problem of using user relevance in the crawling process, it can be combined with a linkage based system in order to find predicate-specific pages which are less popular. 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 [3], 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 [3]. 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
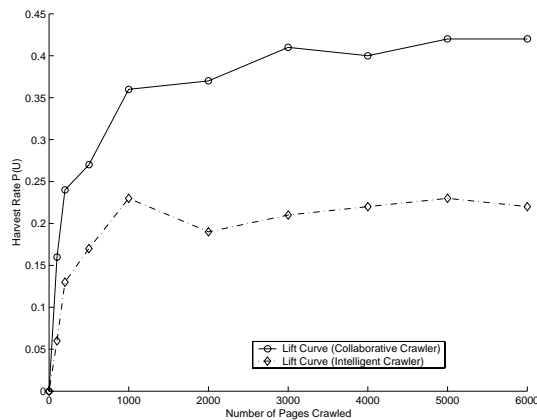
**Figure 2: Performance of Collaborative and Intelligent Crawler (Predicate is category "SPORTS")**
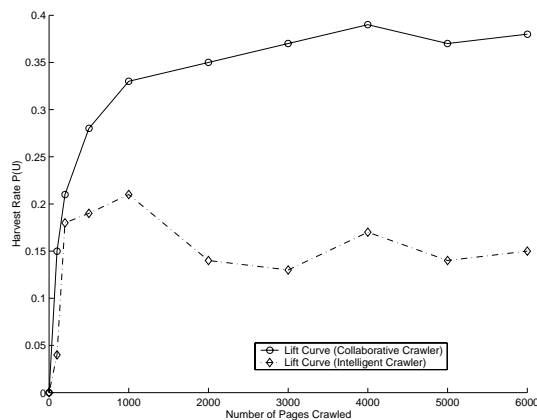


**Figure 3: Performance of Collaborative and Intelligent Crawler (Predicate is category "ARTS")**

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.

## 4. EMPIRICAL RESULTS

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 the Squid proxy traces available from [6].

The performance of the crawler is characterized by using the harvest rate $P(U)$, which is 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. Correspondingly, the percentage of the candidates crawled belonging to the predicate increases as well.

In Figures 2 and 3, we have illustrated an examples of lift curves which show the crawling performance of the system over time. In these curves, we have used two different predicates corresponding to the categories of "SPORTS" and "ARTS" respectively. 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 [3]. 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. 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. In [1], the results are also reported for the incorporation of linkage based information in the collaborative crawler.

## 5. CONCLUSIONS AND SUMMARY

In this paper, we discussed a method for leveraging user experiences for effective topical resource discovery. This is the first topical resource discovery system which deviates from the practice of using hyperlink citation behavior as the primary source of crawling behavior. The utilization of user experiences is critical in designing a system which is sensitive to the behavior of users in finding resources which are both topically accurate and popularly browsed.

## 6. REFERENCES

[1] C. C. Aggarwal. Collaborative Crawling: Mining User Experiences for Topical Resource Discovery. *IBM Research Report*, 2002.

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

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

[4] S. Chakrabarti, M. van den Berg, B. Dom. Focussed Crawling: A New Approach to Topic Specific Resource Discovery. *WWW Conference*, 1999.

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

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