Efficient World-Wide-Web Information Gathering*

TIAN Fan-jiang, WANG Xi-dong, WANG Ding-xing

(Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

E-mail: tfj@cs.tsinghua.edu.cn

http://www.tsinghua.edu.cn

Received June 25, 1999; accepted October 22, 1999

Abstract: With the information available through World-Wide-Web becoming overwhelming, efficient information gathering (IG) tools are necessary. Since the network resources are expensive, so IG is a resource-bounded task. The main purpose of this paper is to find an efficient gathering method for specific topic. This paper presents methods for predicting page's content without downloading it, designs different controlling strategies, and defines several kinds of page downloading priority measures. An IG system, TH-Gatherer, was built to test the methods, and different experiments were carried out. Through experiments, it was found that the content of candidate pages can be predicted approximately without downloading. When the priority based gathering strategy and hybrid measure are used, the gathering efficiency is four times of that of BFS strategy which is used by many current IG tools (including crawlers and off-line browsing tools). The method presented in this paper is suitable for resource-bounded, specific topic information gathering.

Key words: information gathering; world-wide-web application

With the information available through World-Wide-Web becoming overwhelming, efficient information gathering (IG) tools are necessary. Since network resources are expensive, IG is a resource-bounded task. How to gather as much as possible valuable information with as little as possible cost (including time, bandwidth, etc.) is a key problem.

Most current IG systems can be classified into three categories. (1) Crawlers used by most search engines (such as Altavista, Excite, etc.). They do large scale of search, attempt to gather most reachable Web pages, in order to build content indexes[1,2]. (2) Off-line browsing tools, such as webzip[3], teleport. They can download whole Web site. These systems often use BFS (Breadth-First-Search) strategy to determine the sequence of downloading. They are suitable when we know the location of information clearly. (3) Systems designed for problem solving. They often use agent technology, which have to carry out complex syntax and content analysis on gathered information to judge whether it satisfies the condition, as described in Ref. [4]. However, these three kinds of IG method all need to download the information before processing. Since there are often too many pages located on a site, the network traffic of these methods is large and the gathering time is long, so it's not suitable for spe-

* This project is supported by the Fundation of IBM China Research Laboratory (IBM 中国研究中心基金). **TIAN Fan-jiang** was born in 1974. He is a Ph. D. candidate at Department of Computer Science and Technology, Tsinghua University. He received his B. S. degree in computer science from Tsinghua University in 1996. His research interests are parallel computing. **WANG Xi-dong** was born in 1975. He is a master student at Department of Computer Science and Technology, Tsinghua University. He received a B. S. degree in computer science from Tsinghua University in 1998. His research interests are computer networks. **WANG Ding-xing** was born in 1937. He is a professor and doctoral supervisor at Department of Computer Science and Technology, Tsinghua University. His research areas include parallel/distributed computing, computer network.

cific topic, various sites, resource-bounded gathering need.

Harvest system[5] can get summary information of collaborative sites without downloading all information. However, this requires these sites are cooperative, which is impossible in most situations.

Our target is to design an efficient information gathering system, which is suitable for specific topic information gathering. The key problem is to judge the downloading priority of different pages, and the main difficulty is how to predict the content of candidate pages without downloading.

This paper defines measures for calculating the candidate pages' priority and designs controlling strategies for resource-bounded gathering based on the analysis of manual gathering action. An IG system called TH-Gatherer is implemented to test the methods and experiments are carried out.

1 User Action Analysis

Before discussion, let's analyze the action of manual information gathering.

When we need to gather information of specific topic, we submit a query to search engine. Then the search engine will return a sorted URLs list.

Choose a URL from the list, and begin gathering from this initial URL.

During the gathering process, if the downloading speed of a page is very low, we may terminate the downloading of this page and skip all the pages located in this site when we have other choice.

Large pages often mean long downloading time and large storage space, so we would prefer small pages to large pages when they can both satisfy our information need.

For automatic information gathering tools, BFS (Breadth-First-Search) is convenient, but in manual gathering, we would rather use DFS (Depth-First-Search) strategy. The reason is that using DFS we can get a complete topic quickly.

If a page satisfies our information need and the page contains hyperlinks pointing to other pages, we will judge whether to follow the hyperlinks, and to follow which hyperlink according to the page's content and the hyperlink's description.

If a page contains no hyperlink, we will go back to the upper level and find a new page to download.

Repeat this process until all URLs are gathered, or the information need has been satisfied, or there is no resource to use.

The design of TH-Gatherer is based on this analysis. In TH-Gatherer, we simulate the manual gathering process with initial URL. The simulation of searching through search engine will be considered in other paper.

2 Definition of Measures

2.1 Priority (P)

To gather efficiently, the key point is to determine the priority of candidate pages accurately.

When we want to maximize information value, let $P(h)=V(h)$, where $P(h)$ is the priority of page h, $V(h)$ is the value of page h. When we want to maximize the gathering efficiency, let $P(h)=V(h)/C(h)$, where $C(h)$ is the cost of page h. Assume H to be the collection of gathered pages, then $V(H)$ presents total value of H, $C(H)$ presents total cost of H.

$$V(H)=\sum_{h\in H}V(h);C(H)=\sum_{h\in H}C(h)$$

2.2 Value(V)

There are different methods to measure $V(h)$:

(1) The similarity with query (V_{sim})

$V_{sim}(h)$ is defined as the textual similarity between page h and query q. Textual similarity has been well studied in information retrieval (IR)[6]. In IR, document h and query q are both viewed as an n-dimensional vector $\langle w_1,\ldots,w_n\rangle$. The term w_i in this vector represents the significance of the ith word in the dictionary. $V_{sim}(h)$ can be computed as follows:

$$V_{sim}(h)=\frac{\sum_{i=1}^{m}h_iq_i}{\sqrt{\left(\sum_{i=1}^{m}h_i^2\right)\left(\sum_{i=1}^{m}q_i^2\right)}}$$

(2) The number of hyperlinks pointing to page (V_{link})

If there are many hyperlinks pointing to page h, then page h seems important and downloading h can satisfy many pages, so the number of hyperlinks pointing to page h can be used to measure the page's value. The computation of $V_{link}(h)$ is very simple. What we need to do is to extract all the hyperlinks pointing to page h. Let P represent the collection of pages pointing to page h, $L(p,h)$ represent the number of hyperlinks pointing to page h from page p, then:

$$V_{link}(h)=\sum_{p\in P}L(p,H)$$

(3) Weighted V_{link} (V_{w-link})

In V_{link}, all hyperlinks are treated to be of the same importance, however, different pages have different numbers of hyperlinks. If there is only one hyperlink in page h_1, while there are many hyperlinks in page h_2, then the hyperlink in page h_1 is likely to be more important than that of page h_2, since all the hyperlinks will share the importance. We define V_{w-link} (weighted V_{link}):

$$V_{w-link}(h)=\sum_{p\in P}\frac{L(p,h)}{L(p)}$$

where, $L(p)$ represents the number of hyperlinks in page p. It is obvious that V_{link} is the special case of V_{w-link} when $L(p)$ is 1.

(4) Weighted V_{link} incorporating textual similarity ($V_{s-w-link}$)

In V_{w-link}, we don't consider the effect of the similarity of page p on further browsing. In fact, if page p_1 is more close to the information need than page p_2, then it's more likely to select a hyperlink from page p_1 for further browsing. Therefore, we define $V_{s-w-link}$ (weighted V_{link} incorporating textual similarity):

$$V_{s-w-link}(h)=\sum_{p\in P}V_{sim}(p)\frac{L(p,h)}{L(p)}$$

(5) Hybrid measure (V_{hybrid})

Among the four measures we have defined, V_{sim} considers the content of page h. However, it's impossible to compute the V_{sim} of a page which hasn't been downloaded. Instead, we can only obtain V'_{sim}——the estimation of V_{sim}. In contrast, V_{link}, V_{w-link}, $V_{s-w-link}$ all consider the relation between page h and gathered pages. These three measures can be calculated based on gathered pages, but they aren't enough when the page collection is small. To consider the two factors simultaneously, we define a hybrid measure V_{hybrid}:

$$V_{hybrid}(h)=\alpha V'_{sim}(h)+(1-\alpha)V_{(s-w)-link}(h),\ 0\leqslant\alpha\leqslant1$$

where $V_{(s-w)-link}(h)$ can be one of V_{link}, V_{w-link}, $V_{s-w-link}$. In this paper, α takes the value of 0.5.

2.3 Cost(C)

The definition of cost is much simpler than that of value. The main factors affecting the cost of page h are the size of page $Size(h)$ and the downloading speed $Speed(h)$.

We define $C(h)$ as $Size(h)/Speed(h)$.

To avoid too large or too small $C(h)$, $C(h)$ is truncated, so:

$$C(h)=\begin{cases}\text{MINC} & C(h)<\text{MINC}\\ Size(h)/Speed(h) & \text{MINC}\leqslant C(h)\leqslant\text{MAXC}\\ \text{MAXC} & C(h)>\text{MAXC}\end{cases}$$

3 Gathering Strategy

3.1 Predicting Method

In section 2.2, we have mentioned that we can't calculate the V_{sim} of a page that hasn't been downloaded directly. However, during information gathering process, we want to use the V_{sim} of a page without downloading it first. In this situation, predicting method should be developed.

A simple predicting method is Averaging Method, which uses the average of all the V_{sim} values of the pages which have hyperlinks to page h as the prediction of $V_{sim}(h)$. In this method, the predicted similarity value is calculated as follows:

$$V_{sim}(h)=\frac{\sum_{p\in P}V_{sim}(p)}{|P|}$$

This method is similar with that used in Ref. [7]. It can also be extended to a 2-order model, in which:

$$V_{sim}(h)=\alpha\frac{\sum_{p\in P}V_{sim}(p)}{|P|}+(1-\alpha)\frac{\sum_{p\in P'}V_{sim}(p)}{|P'|},\quad 0\leqslant\alpha\leqslant 1$$

where P' is the collection of all pages which have hyperlink to pages in P.

Since the calculation of 2-order model is very complex, 1-order model is used in most situations. Study in Ref. [7] has shown that this predicting method is effective.

However, Averaging Method doesn't take the description of page h in downloaded pages into account. We think the title of hyperlink pointing to page h and the paragraph in which the hyperlink is located are of special importance to describe page h, so we take these two parts into consideration and design a predicting method called Weighted Averaging Method.

Give a page p_i, which contains a hyperlink l to h. p_i contains d_i and t_i. d_i is the paragraph which contains the hyperlink l, and t_i is the title of l. The collection of all d_i is marked as D and the collection of all the titles t_i is marked as T. Then the similarity value can be calculated as follows:

$$V_{sim}(h)=\alpha\frac{\sum_{p\in P}V_{sim}(p)}{|P|}+\beta\frac{\sum_{d\in D}V_{sim}(d)}{|D|}+(1-\alpha-\beta)\frac{\sum_{t\in T}V_{sim}(t)}{|T|},\quad 0\leqslant\alpha,\beta\leqslant 1$$

In experiments described in section 5, α takes the value of 0.5 while β takes the value of 0.3.

3.2 Controlling Strategy

Information gathering system starts off with the URL for an initial page P_0. It retrieves P_0, extracts any URLs in it, and adds them to a queue of URLs to be scanned. Then it selects the next URL to be downloaded (according to certain controlling strategy and measure), and repeats the process.

In this paper, we mainly consider two controlling strategies. One is BFS (Breadth-First-Search), the other is PBS (Priority-Based-Search).

BFS is used by many crawlers and off-line browsing tools. BFS gathers pages according to the sequence of the pages' URLs being extracted.

PBS estimates the priority of each URL in URLs queue using the priority measures defined in section 2, then sorts the URLs queue according to priority. In PBS, the page with the highest priority value is gathered first.

We use PBS in our information gathering system TH-Gatherer, and implement BFS as a baseline.

4 TH-Gatherer

4.1 Architecture

TH-Gatherer is a unit of TH-InfoBase, an Information Gather-Filter-Retrieval System developed by Dept. of Computer Science, Tsinghua University. The architecture of TH-Gatherer is shown in Fig. 1. In Fig. 1, HTML Downloader is responsible for downloading pages. All the information about downloading (including the speed of each Web site) is stored in Site Log. When a Web page is to be downloaded, we send a HEAD request first to get the page size, test the downloading speed and send the information to Link Cost Estimator to estimate this page's downloading cost. In order to speed up the visiting and reduce the network traffic, there is a Site-based Cache, which is responsible for cache downloaded pages. In this paper, we assume the cache is large enough to hold all the pages that have been downloaded and no replacement is needed.

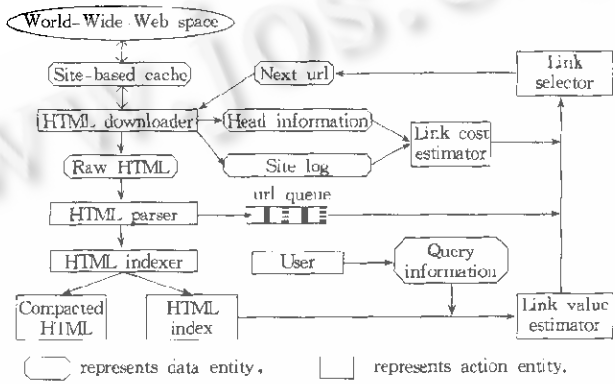

Fig. 1 Architecture of TH-Gatherer

4.2 Algorithm

After the priority measure is defined, the gathering algorithm is very simple. The basic gathering algorithm is shown in Fig. 2.

```
Gathering Algorithm:
Addto_Queue(url_queue,url);                  //add the initial URL into URLs queue
while (not empty (url_queue))                //download until the URLs queue becomes empty
{
    next_url=Movefrom_Queue(url_queue);      //get the first URL in URLs queue for downloading
    page=Download(next_url);                 //download a new Web page
    newurl_list=Parse_Link(page);            //extract all hyperlinks in this page
    Addto_Queue(url_queue,newurl_list);      //add all new hyperlinks into URLs queue
#ifdef PBS                                   //if BFS, this step is skipped
    Sort_Queue(url_queue); //calculate the priority of URLs in queue, and sort the queue
#endif
}
```

Fig. 2 Basic gathering algorithm

5 Experimental Results

5.1 Data set

To test the performance of the controlling strategies and priority measures presented in this paper, we need to construct a test collection of HTML pages. All gathering work is carried out in this collection. Different control-

ling strategies and measures are compared based on this collection. To construct this collection, we submit query "agent" to Yahoo, then select five sites from the result list returned by Yahoo and download all HTML pages on these sites using off-line browsing tool webzip. All these HTML pages form the test collection, which contains 3 547 pages.

5.2 Performance analysis of predicting methods

We use three different methods to calculate the similarity between query and each page in test collection. (1) V_{sim} is calculated based on actual pages' content. V_{sim} is used as the baseline. (2) V'_{sim} is predicted by Averaging Method. (3) V''_{sim} is predicted by Weighted Averaging Method. We use $\Delta V'_{sim}$ to represent the difference between V_{sim} and V'_{sim}, $\Delta V''_{sim}$ to represent the difference between V_{sim} and V''_{sim}.

$$\Delta V'_{sim}=\sqrt{\frac{\sum_{h\in H}(V_{sim}(h)-V'_{sim}(h))^2}{|H|}} \qquad \Delta V''_{sim}=\sqrt{\frac{\sum_{h\in H}(V_{sim}(h)-V''_{sim}(h))^2}{|H|}}$$

Good predicting method should make the Δ value as small as possible. The test is carried out using 10 queries and the results are averaged. To study the effect of number of hyperlinks, we consider pages with different numbers of hyperlinks pointing to separately.

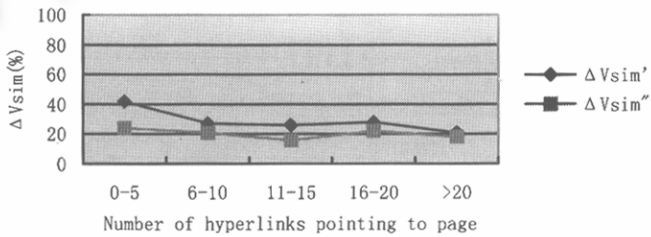

Fig. 3 Performance of predicting methods

As shown in Fig. 3, when the number of hyperlinks pointing to pages is large enough, both predicting methods can predict the value of pages effectively. However, when the hyperlink number is small, the result of method 2 is not as good as that of method 3. The reason is that when there are only small number of hyperlinks pointing to the page, the result of method 2 is affected by incidental factors in known pages. In fact, when there is only one hyperlink pointing to page h from page p, the $V_{sim}(h)$ predicted by method 2 is the same as $V_{sim}(p)$, which is obviously inaccurate. The result tells us it's necessary to consider titles and description paragraphs in method 3, and the experiment result also shows that method 3 performs better than method 2.

5.3 Performance analysis of information value maximization

To evaluate the performance of TH-Gatherer in maximizing information value, we define:

$$\text{Benefit Ratio } R_b=\frac{\text{Current Value of Gathered Pages } V_{current}}{\text{Total Value of All Pages } V_{total}}\times 100\%.$$

When we calculate the benefit ratio, the pages have already been downloaded, so we can use V_{sim} as the value measure. Benefit ratios of different controlling strategies and priority measures are shown in Fig. 4.

As shown in Fig. 4, the R_b of BFS is the lowest. This can be expected because BFS doesn't consider user's need. The benefit ratio using V_{link} measure is not ideal either. This tells that although the V_{link} measure which doesn't consider the page content is useful for crawlers whose target is to gather any "important" pages, it's not suitable for specific topic information gathering. The figure also tells us that methods with more information considered work more efficiently. The best method is PBS using hybrid measure, which can obtain almost 80% value

using only 20% cost, while BFS can only obtain about 20% value using 20% cost.

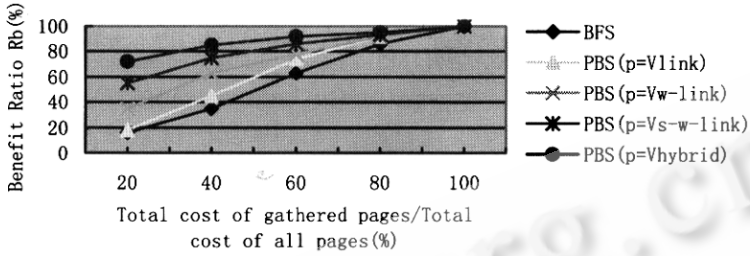

Fig. 4 R_b vs. total cost of gathered pages/total cost of all pages

5.4 Performance analysis of gathering efficiency maximization

To maximize the gathering efficiency, we need to use the priority measure with cost. We also need to define a measure to evaluate the performance of gathering efficiency. For a test collection already downloaded, we can calculate the max (ideal) gathering efficiency E_i. However, in actual gathering process, we can only get an actual gathering efficiency E_r. We define relative gathering efficiency:

$$\text{Relative Gathering Efficiency } E=\frac{\text{Actual Gathering Efficiency } E_r}{\text{Ideal Gathering Efficiency } E_i}.$$

The relative gathering efficiencies of the different controlling strategies and priority measures are presented in Fig. 5.

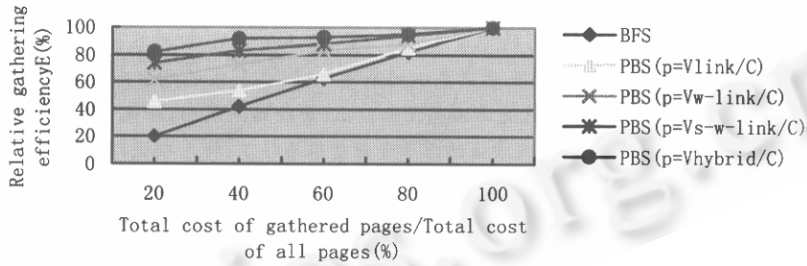

Fig. 5 Relative gathering efficiency vs. total cost of gathered pages/total cost of all pages

The result shown in Fig. 5 is similar with that shown in Fig. 4. However, since we take gathering cost into consideration, the PBS method using hybrid measure performs more effectively.

6 Conclusion

The main purpose of this paper is to find an efficient gathering method. Through experiment, we find that the content of candidate pages can be predicted approximately without being downloaded. When the priority based gathering strategy and hybrid measure are used, the gathering efficiency is four times of that of BFS strategy which is used by many current crawlers and off-line browsing tools. The method presented in this paper is suitable for resource-bounded, specific topic information gathering.

Future work includes using large-scale test collection to do more thorough test on different controlling strategies and measure definitions presented in this paper.

References:

[1] http://altavista.digital.com.

[2] http://www.excite.com.

[3] http://www.webzip.com.

[4] Lesser, V., Horling, B., Klassner, F., *et al*. BIG: a resource-bounded information gathering agent. In: AAAI Press Staff ed. Proceedings of the 5th National Conference on Artificial Intelligence (AAAI-98). Madison, WI: MIT Press, 1998. 243~254.

[5] Bowman, C. M., *et al*. The harvest information discovery and access system. Computer Networks and ISDN Systems, 1995,28(1,2):119~125.

[6] Salton, G. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Reading, Mass: Addison-Wesley, 1989. 146~154.

[7] Dunlop, M.D., Rijsbergen, C.J. van. Hypermedia and free text retrieval. Information Processing and Management, 1993, 29(3):287~298.

高效率 WWW 信息采集

田范江, 王曦东, 王鼎兴

(清华大学 计算机科学与技术系,北京 100084)

摘要: 随着 WWW 上的信息日益丰富,对高效率信息采集(IG)工具的需求日益迫切.由于网络资源非常昂贵,因此,信息采集属于资源受限型任务.主要目标是设计面向特定领域的高效率信息采集方法.提出了在不下载页面的情况下推测页面内容的方法,设计了不同的控制策略,并定义了多种页面下载优先级定量指标,建造了一个信息采集系统——TH-Gatherer,并进行了不同的实验以检验此方法.实验证明,可以在不实际下载页面的情况下,近似推测出候选页面的内容,采用混合尺度的基于优先级的采集方法,在采集效率方面比当前许多信息采集工具(包括 Crawler 和离线浏览工具)常用的宽度优先方法高 4 倍以上.实验结果表明,所设计的获取方法在获取效率方面比当前常用的宽度优先方法高 4 倍以上.此方法适用于资源受限条件下、特定领域的信息采集.

关键词: 资源受限的信息采集;WWW 应用

中图法分类号: TP393 **文献标识码**: A