The Method of Keyword Based Crawler Load Balancing

Mo-ji WEI, Yan-qing ZHAO, Shi-wei ZHU and Ai-qin YANG

Information Research Institute, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China

Keywords: Distributed web crawler, Load balancing strategy, Seeding links allocation strategy.

Abstract. This paper researches feature of different data sources such as web site and social media, and proposes a load balancing method for distributed web crawlers by calculating weights of crawling data from various sources. Firstly, a seeding links allocation strategy is proposed based on analyzing differences of statistical data update frequency of different data sources. Then with the allocation strategy a data crawling solution using domain names and keywords as its task unit is given. Finally, by adjusting allocation among distributed web crawlers with calculating time expenditures of domain names and keywords as its weights, a load balancing method is proposed.

Introduction

In big data era, data sources become more diverse. Besides early websites like policy websites which guide the development of industries, and news websites which introduce competitor product and lasted science & technology trend in related field [1]. Social media such as blog, forum, twitter, facebook, etc. which post users’ comments of products have been paid more attentions. Crawling information from various data sources and fusing with private data could help supporting decision, planning schedule, saving cost, promoting marketing, and improving services for organization itself, and could also provide opportunities for better understanding competitors. Distributed web crawlers start seeding links and go through various data sources of Internet, download information to local servers. Application of crawler system meets information crawling needs of organizations [2-5].

Depending on the deployment and topology of distributed web crawler system, it can be divided into Single-domain Distributed Crawler that crawlers are deployed in one local area network and Multi-domain Distribute Crawler that crawlers are distributed in wide area network [6]. The fundamental function of crawler system, no matter it is Single-domain Distributed crawler or Multi-domain Distribute Crawler, is crawling data from data sources, and the kernel of the system is scheduling strategy which determines the crawl mode, performance and efficiency. The scheduling strategy mainly includes seeding links allocation strategy, load balancing strategy and web page checking strategy, etc.

Seeding links allocation strategy is mainly divided into three categories: independent mode, static mode and dynamic mode [7-9]. In independent mode, crawlers do not communicate with each other and crawl their own data sources independently. In static mode, crawlers are assigned with specific data sources which are pre-divided statically. In dynamic mode, crawlers will be dynamically assigned with new data sources when they finish current crawling tasks. Among all these three modes, in order to reduce communication overhead they take domain name as its unit of assignment of data sources.

Load balancing can be divided into static load balancing and dynamic load balancing [10-12]. In static load balancing, it transfers tasks by the specific strategy such as polling, ratio or priority to balance the load among crawlers [13-15]. While in dynamic load balancing, it transfers tasks by assessing state of each crawler in the crawling process [16-19]. No matter static load balancing or dynamic load balancing, the transferred tasks are usually means domain names. In load balancing, it usually reassigns part of domain names which are gotten from high load crawlers to low load crawlers to balance the load of the whole system.

In this paper, by analyzing the feature of different data sources, we propose a seeding links allocation strategy which is using domain name and keyword hybrid rather than domain name as task
unit, and a load balancing strategy based on such task unit. By this it could reduce communication overhead further and balance load in finer grained way.

**Seeding Links Allocation Strategy**

Data sources which are interested in by various organizations include traditional websites such as policy websites, news websites, etc. and multifarious social media like blog, forum, twitter, facebook, etc. Usually the information in traditional websites is written by editorial staff, therefore update frequency and quantity of information are low, and daily updates are no more than 1000. Contrarily, social media is open platform, every netizen can post his/her opinions to the platform. Accordingly update frequency and quantity of information are extremely high, daily updates of one platform are more than one billion. Facing such a dense update, it is a huge obstacle for any crawler system. Fortunately every organization just concentrates on the information related to business field, such information is a tiny subset of data provided by various data sources. To describing business field, ontology knowledge base, which is composed by entities (represented by keywords) and relations among entities, will be constructed by the organization. Consequently with data sources and keywords in ontology knowledge base, there are two crawling methods to gather helpful information. One is crawling all the data from target data source, and then filtering out irrelevant information with keywords. The other is retrieving information with keywords in the target data sources, and then crawling retrieval results.

Let the number of keywords in the ontology be $N_{kw}$, the amount of information updated per unit time in a target data source be $N_d$, the number of titles in a list page be $C$, the number of helpful information in information updated per unit time be $N_i$. Then the times of server access can be divided into two parts. One is the times of list page access, and the other is the times of content page access.

Suppose that the helpful information is uniformly distributed in the posted information, the access times of gathering all $N_i$ information will be $\frac{N_d}{C} + N_d$ with first crawling method, and with second crawling method it will be $N_{kw} + \frac{N_i}{C} + N_i$. For social media whose updating frequency is extremely high, in this case $N_d \gg C$, $N_d \gg N_i$ and $N_d \gg N_{kw}$, the access times with first method is far greater than the second method. Obviously the second method presents a significant advantages over the first one. On the other hand for traditional website whose updates less than 1000, $N_d$ is not much bigger than $N_i$. Especially if the website is elaborately chosen thematic website by the organization, then $N_d$ is approximately equal to $N_i$. In this case the access times will be greatly influenced by $N_{kw}$. Comparing with retrieving keywords from remote server, filtering irrelevant information in local server will consume less time. Accordingly we incline to use the first crawling method rather than the second one to gather information from websites. In summary, for website the first crawling method is a good option. For social media the second crawling method is more appropriate.

**Data Source Crawling Solution**

Because gathering information from websites uses the first crawling method, master crawler node allocates one domain name to one execution crawler node as other allocation strategies. Nevertheless for social media, the second crawling method retrieves keywords before crawling, which means beyond domain name keywords should be allocated as well. In our solution, to avoid one crawler being banned for crawling too much data from same server, master crawler node allocates one domain name to all execution crawlers, and allocates each execution crawler exclusive keywords.

Let the number of execution crawlers be $N_{cl}$, the number of domain name of websites be $N_{web}$, the number of domain name of social media be $N_{soc}$, and the number of keywords related to the target
field be $N_{kw}$. Initially each execution crawler will be allocated with $\frac{N_{web}}{N_{cl}}$ different domain name of websites, $N_{soc}$ domain name of social media, and $\frac{N_{soc}}{N_{cl}}$ exclusive keywords.

As depicted in Table 1, task allocation list of execution crawlers is registered in master crawler. The structure of website allocation list is shown in Table 1(a) where it is recorded the domain names and weight of websites allocated to each execution crawler. While Table 1(b) shows the structure of social media allocation list. For one social media will be allocated to all the execution crawlers, it is not necessary to record the domain name of social media. The table just needs to record the keywords and its weight of each social media allocated to each execution crawler.

Table 1. Task Allocation List.

<table>
<thead>
<tr>
<th>Spider1</th>
<th>Spider2</th>
<th>Spider3</th>
<th>Spider4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Website</td>
<td>Website</td>
<td>Website</td>
<td>Website</td>
</tr>
<tr>
<td>Web1</td>
<td>Web1</td>
<td>Web1</td>
<td>Web1</td>
</tr>
<tr>
<td>Web2</td>
<td>Web2</td>
<td>Web2</td>
<td>Web2</td>
</tr>
<tr>
<td>Web3</td>
<td>Web3</td>
<td>Web3</td>
<td>Web3</td>
</tr>
<tr>
<td>Weight</td>
<td>Weight</td>
<td>Weight</td>
<td>Weight</td>
</tr>
</tbody>
</table>

(a) Website allocation list

<table>
<thead>
<tr>
<th>Spider1</th>
<th>Spider2</th>
<th>Spider3</th>
<th>Spider4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Website</td>
<td>Website</td>
<td>Website</td>
<td>Website</td>
</tr>
<tr>
<td>Web1</td>
<td>Web1</td>
<td>Web1</td>
<td>Web1</td>
</tr>
<tr>
<td>Web2</td>
<td>Web2</td>
<td>Web2</td>
<td>Web2</td>
</tr>
<tr>
<td>Web3</td>
<td>Web3</td>
<td>Web3</td>
<td>Web3</td>
</tr>
<tr>
<td>Weight</td>
<td>Weight</td>
<td>Weight</td>
<td>Weight</td>
</tr>
</tbody>
</table>

(b) Social media allocation list

The weight $W_{web}$ for website $web_i$ is calculated as (1).

$$W_{web} = \frac{T_{web}}{\sum_{j=1}^{N_{web}}}.$$

In (1), $T_{web}$ is time cost on crawling information from domain name of website $web_i$, $i \in [1, N_{web}]$, $j \in [1, N_{web}]$. Initially when the time cost on every website is unknown, it is supposed that all the websites have equal time cost. Then the weight of each website is $\frac{1}{N_{web}}$.

The weight $W_{soc,kw}$ for keyword $kw_j$ of social media $SOC_i$ is calculated as (2).

$$W_{soc,kw} = \frac{T_{soc,kw}}{\sum_{k=1}^{N_{soc}}}.$$

In (2), $T_{soc,kw}$ is time cost on crawling information which is retrieved by keyword $kw_k$ from domain name of social media $SOC_i$, $i \in [1, N_{soc}]$, $j \in [1, N_{soc}]$, $k \in [1, N_{kw}]$. Initially when the time cost on
crawling information for every keyword of each social media is unknown, it is supposed that they have
equal time cost. Then the weight of every keyword of each social media is \( \frac{1}{N_{\text{web}}} \).

Load Balancing Strategy

Data crawling Time Table

For both of website and social media will be crawled by execution crawler, there are two kind of data
crawling time tables on execution crawler. One is data crawling time table of website which is
composed of three columns: web, cost and completion. As shown in Table 2, column web records the
domain name of website, column cost and column completion respectively record time cost and
completion time on crawling all data from one domain name. The other is data crawling time table of
social media which is composed of four columns: soc, keyword, cost and completion. As shown in
Table 3, column soc records the domain name of social media, column keyword records the allocated
keywords of each social media, column cost and column completion respectively record time cost and
completion time on crawling keyword-related data from one domain name.

<table>
<thead>
<tr>
<th>Web</th>
<th>Cost</th>
<th>Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Webj</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>Webj+1</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>webk</td>
<td>***</td>
<td>***</td>
</tr>
</tbody>
</table>

Table 2. Time Table of Website.

<table>
<thead>
<tr>
<th>Soc</th>
<th>Keyword</th>
<th>Cost</th>
<th>Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>soc1</td>
<td>KW_{s1k1}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>soc1</td>
<td>KW_{s1k2}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>soc1</td>
<td>KW_{s1k}\text{a}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>soc2</td>
<td>KW_{s1k}\text{b}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>soc2</td>
<td>KW_{s1k}\text{b+1}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>socN_{soc}</td>
<td>KW_{s1k}\text{c}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>socN_{soc}</td>
<td>KW_{s1k}\text{d}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>socN_{soc}</td>
<td>KW_{s1k}\text{d+1}</td>
<td>***</td>
<td>***</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>socN_{soc}</td>
<td>KW_{s1kNkw}</td>
<td>***</td>
<td>***</td>
</tr>
</tbody>
</table>

Table 3. Time Table of Social Media.

Every execution crawler sends data crawling time tables to master crawler after each round of
crawling data from its own website and social media. Then the master crawler exports data from the
tables to data crawling time summary table.

Process of Load Balancing

The data crawling time summary table is analyzed once at midnight every day. For website it collects
time cost grouped by domain name firstly, and then computes the average of time cost for crawling
information from each domain name, finally calculates the weight of each website according to (1).
Similarly For social media it calculates the weight of each keyword of each social media according to
(2).

The weight reflects the load of execution crawlers. It can balance the load of execution crawler
through transferring the websites and keywords of social media among execution crawlers to make
the summation of weight of each execution crawler approximation. Here we take one social media
(whose domain name is sm) as an example to explain the process of load balancing strategy. The process can be divided into 8 steps.

Step 1 Under the social media sm, sum weight of all the keyword allocated to each execution crawler, and find the minimum and the maximum, then calculate the difference between them. If the difference is greater than the preset threshold, go to Step 2. Otherwise keep the keywords allocation as it is.

Step 2 Sort the keywords in descending order of weight.

Step 3 Allocate the 1st to the \(N_{el}\)th keywords to the execution crawlers spider1 to spider\(N_{cl}\) respectively, and then allocate the \(N_{el}+1\)th to the \(2N_{el}\) keywords to the execution crawlers spider1 to spider\(N_{cl}\) respectively, iterate the allocation till the last keyword. Sign the keywords distribution as initial status KW_Init_Assignment.

Step 4 In initial status, sum weight of all the keyword allocated to each execution crawler, then calculate variance of weight of all the execution crawlers, the variable Variance_Init is used to represent initial variance of weight. Let the number of iterations be 0, here we take variable Iteration as number of iterations, that is Iteration=0

Step 5 Let Iteration=Iteration+1. If Iteration is less than the preset number of iterations iteration_threshold (that is Iteration< iteration_threshold), let variable Variance_MinIteration be Variance_Init (that is Variance_MinIteration = Variance_Init), and go to Step 6. Otherwise go to Step 8.

Step 6 Randomly select two execution crawler and randomly pick up a keyword from each execution crawler, then re-calculate variance of weight of all the execution crawlers after exchanging the keywords, the variable Variance_Random is used to represent the re-calculated variance.

Step 7 compare Variance_Random and Variance_MinIteration.

Step 7-1 if Variance_Random<Variance_MinIteration, keep the exchanged keywords as it is and let Variance_MinIteration=Variance_Random, counter=0, go to Step6.

Step 7-2 if Variance_Random≥Variance_MinIteration, revoke the exchanged keywords, let counter=counter+1. If counter is less than preset counter_threshold which is used to prevent unlimited circulation, go to Step5. Otherwise Sign the keywords distribution as the Iteration-th status KW_IntermedIteration_Assignment, go to Step5.

Step 8 Compare the Variance_MinIteration from various KW_IntermedIteration_Assignment, let the keywords distribution of KW_IntermedIteration_Assignment which has minimal Variance_MinIteration be the final status KW_Final_Assignment. Finally for social media sm, allocate the keywords to execution crawlers according to KW_Final_Assignment.

The keywords of other social media with different domain names are allocated as above process. The website can be similarly allocated as above as well. The difference is that the keywords should be replaced with websites instead.

Several local optimum can be found by increasing iteration_threshold and counter_threshold. Theoretically when iteration_threshold and counter_threshold tend to infinity, the strategy can converge to the global optimum. In practice thresholds can be set depending on experiences.

Conclusion

From the perspective of application, by analyzing features of crawling data from different data sources, this paper proposes a seed allocation strategy to satisfy such organization which has limited servers and needs large amount of data in a specific field. It also design a load balancing strategy for the crawler system which use the seed allocation strategy.

Acknowledgement

This paper is sponsored by the key research and development program of Shandong province (No.2016GGX101018, No. 2017GGX10118).
References


