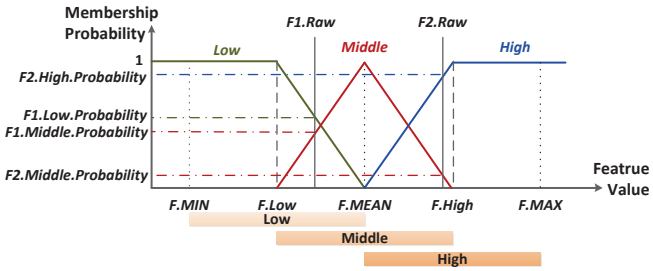

Figure 4: The discretized function of features

where α is a weight value for balancing the importance between the topical region and the non-topical regions. Here, we set the value of α to 0.5. Based on the value of *F.MIN*, *F.MEAN* and *F.MAX*, we then calculate two cut points *F.Low* and *F.High*, where *F.Low* is the average of *F.MIN* and *F.MEAN*, and *F.High* is the average of *F.MEAN* and *F.MAX*. Now, the raw feature values can be represented by their probabilities of falling into three scopes: Low, Middle and High, respectively, as shown in Fig. 4. Note that each feature could only has two non-zero values in Low, Middle and High, and then, we take the larger one as the discretized value of this feature.

C. Topical block segmentation and main text identifying

Once the features are extracted, we can construct a supervised learning classifier with a training set that the positive and negative instants are labeled manually. Various typical machine-learning based classifiers could be employed, including Naive Bayes, C4.5, SVM and so on. The classifier can determine whether a given text line belongs to main text, and the text lines belongs to main text is called topical lines. This initial classification, though rough, labels most of the text lines correctly. However, if there are lengthy copyright notices, comments, and/or descriptions of other stories (not part of the main text), then those will likely to be labeled as topical lines too. Also, if there are descriptions around inline graphics that are part of some advertisement, or lengthy textual advertisements, these may also be labeled as topical lines. False negatives could also be observed when a topical line is not sufficiently long. To address these issues, a sliding window technique is firstly utilized to segment a web page to several potential topical blocks. The process is described below: First, we associate the i^{th} text line with a Boolean variable M_i (TRUE represents that the line is a topical line and FALSE otherwise) according to the classifier in the previous step; and then, scanning the entire HTML source file from top to bottom with a sliding window. The k^{th} potential topical block is represented as $B_k(start_k, end_k)$, where $start_k$ is the start position of the block, and end_k marks where it ends. $B_k(start_k, end_k)$ must satisfy the following conditions:

1) $M_i = FALSE$, if $start_k - \phi \leq i \leq start_k$ or $start_k - \phi \leq i \leq start_k$, where ϕ is the length of the sliding window which is empirically set to 5 generally;
2) $M_i = TRUE$, if $i = start_k + 1$ or $i = end_k - 1$;
3) $\underset{start_k < i < j \leq end_k}{\text{MAX}} d_{ij} \leq \phi$, where M_i and M_j are TRUE.

In other words, in a topical block, no more than $\phi - 1$ continuous non-topical lines can be included. This way, some false negatives from topical line detection can be tolerated.

After extracting potential topical blocks, the proposed algorithm identifies the most informative blocks. An informative block contains meaningful information that would be the target of main text extraction. The other blocks that contain noise information such as advertisements, menus, or copyright statements are considered non-informative blocks. A topical line of a web page usually has a high density of texts with few links or images. Thus we can use a simple Score to measure the informativeness of every HTML line:

$$Score(P_i) = |T_i| + |O_i| + |D_i| - |L_i| - |I_i| \quad (4)$$

where $|T_i|, |O_i|, |D_i|, |L_i|, |I_i|$ are the value of *TextLength, OutputTextLength, Density, LinkNum, ImgNum*, respectively, and normalized in the range of [0,1] as mentioned in Sec. III-B. if $Score(P_i) > 1.5$, the line will be considered a topical line. To recognize informative blocks, we can use the average length of the plain text lines (T), the average length of the output text line (O), the average ratio of plain text to HTML bytes (D), the average number of links (L), and the average number of images (I). The following heuristic rule is used for evaluating the informativeness of a topical block:

$$Score(B_k) = \frac{\sum_{Start_k \leq i \leq End_k} Score(P_i)}{End_k - Start_k + 1} \quad (5)$$

If $Score(B_k) > 1.5$, the block will be considered an informative block, and all the informative blocks will form the main text. It is worth noting that this algorithm can help reduce false positives from topical line detection. A long line of advertising text and some descriptions of the other stories and pictures, which could be marked as topical lines in the previous step, are likely to be more isolated, i.e., more likely to be surrounded by other non-topical text lines, than a typical topical line. Therefore, the Score would be relatively lower for any block that contains that line, when compared with the score of a typical topical block where true topical lines tend to stay closely together.

IV. Experimental Results

In this section, we present experimental results of DGWC. Both the performance and the effectiveness of main text extraction are evaluated. All experiments were performed on a cluster with 8 nodes connected by Gigabit Ethernet. Each node comes with four quad-core E5-2650v2 processors (2.6GHz), 128GB of RAM, 240GB of SSD disk and 600GB of SAS disk.

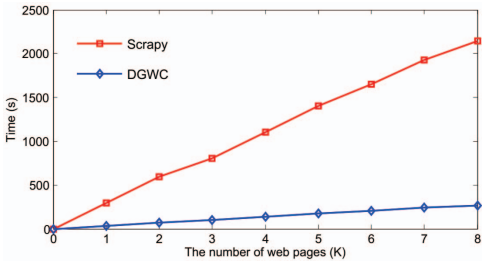

Figure 5: Comparison on spent time between DGWC and Scrapy

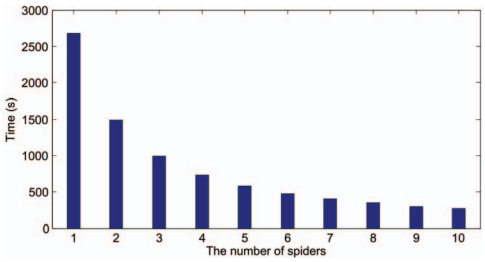

Figure 6: Impact of the spider number on spent time

For performance evaluation, we take the task that to get eight thousand web pages from eight web sites. The sites are listed in Tab. II that half of them are Chinese web sites and the other half are English web sites. In each web site, we get one thousand pages and parse them by `BeautifulSoup` because it is more stable for performance comparison. We first compare the DGWC and standalone version of `Scrapy` by time spent, here the `Scrapy` is run on one node of the cluster and the number of DGWC spiders is 10 (Note that the spider is packaged by `Docker` and one node can run multiple `Docker` packets). To exclude the interference of network delay, we only count the time between response-arrived and parsing-completed. The result is shown in Fig. 5, from which we can see that the DGWC spent obvious less time than `Scrapy` standalone version. Note that the spent time ratio of DGWC and `Scrapy` is a little larger than that of one and spiders number, this is due to the jobs scheduling by shared `Redis` consume some time. However, the results of this experiment are sufficient to validate the efficiency of DGWC. Furthermore, we count the spent time from one spider to ten spiders, as shown in Fig. 6. We can see the speed of time reducing is fall behind the speed of the spider number increasing, which also demonstrate that scheduling jobs between spiders will spent additional time. However, it is worthy that the total time spent in parsing web pages is indeed decreasing.

For the effectiveness valuation of main text extraction, we first select 100 web pages from each web site's 1000 web pages that got in the performance experiment. Then, we construct a dataset with total 800 web pages, and label the main text manually. The main text extraction algorithm proposed in Sec. III with SVM as classifier is utilized, and a 10-flod cross validation is applied to evaluate the effectiveness. We adopt the metrics in [5] to assess our results, which include precision, recall and F-measure. A higher value of precision indicates fewer wrong classifications, while a higher value of recall indicates less false negatives. They are calculated as follows:

Table II: Comparison of main text extraction methods

Web Sites	Our Algorithm			WISDOM	VIPS
	P	R	F	F	F
www.sohu.com	0.976	0.987	0.981	0.955	0.832
www.163.com	0.973	0.989	0.981	0.926	0.771
www.sina.com.cn	0.943	0.967	0.955	0.912	0.692
www.qq.com	0.931	0.952	0.941	0.916	0.875
edition.cnn.com	0.951	0.979	0.965	0.931	0.856
www.nydailynews.com	0.912	0.943	0.927	0.926	0.801
www.newsday.com	0.961	0.983	0.972	0.942	0.855
www.bbc.com	0.955	0.973	0.964	0.933	0.792

$$\boldsymbol{Precision} = \frac{|bag(C) \cap bag(MT(WP))|}{|bag(C)|} \tag{6}$$

$$\boldsymbol{Recall} = \frac{|bag(C) \cap bag(MT(WP))|}{|MT(WP)|} \tag{7}$$

where $bag(C)$ denotes the bag of output text/contect associated with a chunk of text C. $|bag(C)|$ is the length of output text/content (measured in HTML bytes) in C. $MT(WP)$ is the main text of a web page WP. It is common to use the harmonic mean of both measurements, called F-measure, such as the *F1-measure* defined by Eq. (8) which weighs precision and recall equally important.

$$\boldsymbol{F1-measure} = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{8}$$

To validate the effectiveness, we compare our algorithm with two well-known existing DOM based approach WISDOM [6] and Vision-based approach VIPS [7] in accuracy. The results are shown in Tab. II, from which we can see our algorithm achieves high accuracy in main text extraction.

V. Related Work

Various web crawlers such as `Larbin` [2], `Heritrix` [3] and `Scrapy` [4], have been designed for automatically online information extraction. However, few crawler systems are considered carefully for parallel implement and run to meet the requirement of massive data extraction. Besides the performance, another problem bought by the large amount of web sites is pages parsing. In recent years, a large number of researches have addressed to automatically identify information from web sources [8]. Differentiated by their scopes, these works can be categorized into Document Object Model (DOM) based [6][9], vision-based [7][10], and statistics-based [11][12] approaches. Among the three methods, the first two both need to render the page, and exploit

specific extraction strategies during the extraction process. They are indeed unsuit for the massive heterogeneous web pages, thus only the statistics-based approaches can be used in large-scale crawler systems. The exist statistics-based approaches usually use density [11] or word count [12] as statistics objects, however, more features in web pages could be used to achieve better accuracy. In this paper, we propose a distributed and generic web crawler system to parallel access and parse web pages. And furthermore, a self-designed main texts extraction algorithm is integrated to parse web pages uniformly.

VI. Conclusion

This paper presents the design of a distributed and generic web crawler system (DGWC) for online information extraction. To improve the performance, we take `Redis` as a shared storage to schedule jobs among multiple `Scrapy` spiders in order to parallel access and parse web pages. Furthermore, the crawler program and dependencies are packaged as a `Docker` container to make the crawler system could be easily horizontal scaling. To extract main text from various heterogeneous web pages, we propose a statistics-based two stages approach. We first extract a set of features and construct a supervised-learning classifier, then identify topical lines using this classifier. To improve the accuracy, a slide window is utilized to recognize potential topical block and extract the final main text based on block score. The experimental results show that DGWC has satisfactory performance to extract meaningful information from massive heterogeneous web pages.

Acknowledgment

This research was partially supported by National Natural Science Foundation of China under Grants 61502222, 71571093 and 71372188, National Key Technologies R&D Program of China under Grants 2014BAH29F01, National Center for International Joint Research on E-Business Information Processing under Grant 2013B01035, Industry Projects in Jiangsu S&T Pillar Program under Grant BE2014141, Natural Science Foundation of Jiangsu Province of China under Grant BK20150988, Surface Projects of Natural Science Research in Jiangsu Provincial Colleges and Universities under Grants 15KJB520012 and 15KJB520011, and Pre-Research Project of Nanjing University of Finance and Economics under Grants YYJ201415.

References

[1] Q. S. Zhang, B. L. Xie, and X. M. Zhang, "Uncertain internet public opinion emergency decision-making method under interval-valued fuzzy environment," in *Applied Mechanics and Materials*, vol. 713. Trans Tech Publ, 2015, pp. 2024–2028.

[2] M. Yan and S. Wang, "System architecture of larbin web crawler," *Computer Study*, vol. 4, p. 045, 2010.

[3] G. Mohr, M. Stack, I. Ranitovic, D. Avery, and M. Kimpton, "An introduction to heritrix an open source archival quality web crawler," in *In IWAW04, 4th International Web Archiving Workshop*. Citeseer, 2004.

[4] J. Wang and Y. Guo, "Scrapy-based crawling and user-behavior characteristics analysis on taobao," in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012 International Conference on*. IEEE, 2012, pp. 44–52.

[5] E. S. Laber, C. P. de Souza, I. V. Jabour, E. C. F. de Amorim, E. T. Cardoso, R. P. Rentería, L. C. Tinoco, and C. D. Valentim, "A fast and simple method for extracting relevant content from news webpages," in *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 2009, pp. 1685–1688.

[6] H.-Y. Kao, J.-M. Ho, and M.-S. Chen, "Wisdom: Web intrapage informative structure mining based on document object model," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 5, pp. 614–627, 2005.

[7] J. Kang, J. Yang, and J. Choi, "Repetition-based web page segmentation by detecting tag patterns for small-screen devices," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 2, pp. 980–986, 2010.

[8] Z. Bu, C. Zhang, Z. Xia, and J. Wang, "An far-sw based approach for webpage information extraction," *Information Systems Frontiers*, vol. 16, no. 5, pp. 771–785, 2014.

[9] S. Gupta, G. Kaiser, D. Neistadt, and P. Grimm, "Dom-based content extraction of html documents," in *Proceedings of the 12th international conference on World Wide Web*. ACM, 2003, pp. 207–214.

[10] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, "Extracting content structure for web pages based on visual representation," in *Asia-Pacific Web Conference*. Springer, 2003, pp. 406–417.

[11] J. C. Alex, "The easy way to extract useful text from arbitrary html," 2007.

[12] B. Zhou, Y. Xiong, and W. Liu, "Efficient web page main text extraction towards online news analysis," in *e-Business Engineering, 2009. ICEBE'09. IEEE International Conference on*. IEEE, 2009, pp. 37–41.