Crowdcrawling Approach for Community Based Plagiarism Detection Service

Sergey Butakov
Concordia University College of Alberta
7128 Ada Boulevard, Edmonton, AB,
Canada
sergey.butakov@concordia.ab.ca

ABSTRACT
In the era of exponentially growing web and exploding online education the problem of digital plagiarism has become one of the most burning ones in many areas. Efficient internet plagiarism detection tools should have a capacity similar to that of conventional web search engines. This requirement makes commercial plagiarism detection services expensive and therefore less accessible to smaller education institutions. This work-in-progress paper proposes the concept of crowdcrawling as a tool to distribute the most laborious part of the web search among community servers thus providing scalability and sustainability to the community driven plagiarism detection. It outlines roles for community members depending on the resources they are willing to contribute to the service.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Information filtering, Relevance feedback, Retrieval models; H.3.4 [**Systems and Software**]: Distributed systems.

General Terms
Management, Design, Human Factors.

Keywords
Crowdcrawling; Crowdsourcing; Plagiarism Detection; Web Information System Architectures.

1. INTRODUCTION
The problem of web plagiarism is not new to the education but such factors as exponentially growing web, education moving to online environment and ever increasing information pressure on the students make internet plagiarism one of the most important problems in education [12][15]. Manual plagiarism detection is not an option in most subjects with the exception of very narrow research areas where an educator may be familiar with most of the texts available online. Fortunately, technology is here to help: there are some successful commercial plagiarism detection services (PDS) available on the market such as Turnitin (www.turnitin.com) or SafeAssign (www.safeassign.com). But these providers do require expensive subscriptions which may not be affordable to smaller institutions. The potential solution would be to create a free (or at least more affordable) community driven plagiarism detection tool. Community efforts to tackle complex large scale project have been proven as a successful tool in crowdsourcing projects such as Wikipedia or numerous large scale open source software development. The central idea of this paper is to replicate the same community based approach on the task of internet plagiarism detection.

A typical PDS provides two types of services - local search for papers located in the institutional database and global search for the text available online. This paper considers only the second case - internet search which typically includes two phases: 1) narrowing down the results from billions of potential sources available on the web to more manageable volume; and 2) detailed comparison of limited number of documents with the document in question. Among these two phases, this paper concentrates on the first one – providing web search service.

Similar to a typical web search engine [2] internet PDS should have two major components: Crawler - to grab content from the web and Indexing - to transform searchable objects in the form that allows fast search. Scale of indexing mechanism depends on PDS workload but scale of Crawler should be comparable to the one of major search engines. The latter is required to assure the quality of internet detection. Theoretically PDS can rely on a conventional search engine but practically this would be expensive as major search engines such as Google, Bing or Yahoo require subscription to use their API. Web plagiarism detection tools require extensive use of the search engine. For example, one of the popular plagiarism detection plugins for Moodle Learning Management System (LMS) may send as many queries to the search engine as number of words in document in question. Using current pricing model for Google API it would cost $25 to check one document of 5000 words. PAN workshops and competitions on plagiarism detection (http://pan.webis.de/) have a separate track on Source Retrieval that has minimization of the queries to the search engine as one of the tasks for competition. Therefore the need for inexpensive (or free) web search engine has become the central problem this work-in-progress paper is trying to address. More specifically it concentrates on building the Crawler component of the web search service.

Utilizing idle community resources could potentially help addressing crawler scalability problem. This paper proposes a top view description of mechanism for such utilization. The main idea will be to use underloaded community resources to perform distributed crawling. It means that potential users of the PDS will have to contribute to the crawling efforts and feed the results of crawling to a centralized (or decentralized) Indexing mechanism. The latter is not discussed in this paper due to space and resource constraints. The concept of crowdcrawling proposed in this project is similar to the concepts of crowdfunding and crowdsourcing. The similarity is twofold: members volunteer to participate in crawling and they assume responsibility for the quality of their work.

WWW'14 Companion, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2745-9/14/04.
http://dx.doi.org/10.1145/2567948.2580057

The rest of the paper is organized as follows: the next section covers related research primarily discussing projects related to distributed crawling and collaborative plagiarism detection. The third section proposes the draft architecture for community-driven PDS based on the crowdcrawling idea. The fourth section discusses the potential and limitations for the outlined architecture.

2. RELATED RESEARCH

Three areas have been reviewed for the related research: distributed crawling, human involvement into web crawling projects, and collaborative plagiarism detection.

The idea of distributed crawling is not new: exponential growth of the web and its geographical dissimilation forced researchers and practitioners to look into the possibility of allocating web search jobs to different nodes. Researchers from Yahoo! Research listed the following internal and external issues related to Crawling function in a distributed web search environment [3]:

- External issues: Web growth, Content change, Network topology, Bandwidth, DNS, QoS of Web servers
- Internal issues: URL assignment, Re-crawling, URL exchanges.

External issues represent main driving forces for distribution of the workload. As an example, some of these issues could be solved with proper geographical allocation of crawling units [5][11]. Internal issues require one or more units on the distributed network to assume coordination responsibilities to make sure crawling efforts are performed in the most efficient way. Numerous projects offer different strategies for workload distribution among crawling nodes. For example, the LiDi Crawl proposal offers two strategies to perform crawling task: 1) distributed crawlers get fixed set of URLs which they must work on; and 2) distributed crawlers get initial seed pages they must start from [10]. First strategy gives more control (and workload) to the master node while second would give more decision power to the remote crawlers but require extra coordination mechanism to reduce duplicated crawling efforts from different distributed (slave) crawlers. LiDi Crawl example illustrates a centralized approach for workload allocation in a distributed crawling environment. Another example provided in [17] shows peer-to-peer oriented approach where each node has certain degree of participation in workload distribution. In another project, crawling effort efficiency has been proposed to be a driving force to manage a community of distributed crawlers [6]. Many other related project on distributed crawling offer similar coordination mechanisms based on various level of centralization [4][7][14]. Regardless of the coordination and workload distribution mechanism, any crawling effort would require seed sites and some level of quality control for the information that crawlers feed back to the search engine.

Quality of crawling defines quality of the web search. Conventional search engines like Yahoo! or Google dedicate noticeable resources to avoid so called web spam. Apart from web garbage filtering tools there are other mechanisms available to address the quality issue. For example, EverLast architecture for digital web preservation suggests employing "human assisted crawling" from web surfers to improve the accuracy of crawling [1]. In EverLast it is suggested that users supply relevant web links from their browsing history via project-connected browser plugins. Similar suggestion mechanisms could be employed for plagiarism detection where professors act as subject matter experts by suggesting potential sources for plagiarized papers.

Community based control has been proven as efficient mechanism for quality assurance in many projects, including Wikipedia and Google Translate. It could be also utilized in the similar way in plagiarism detection projects. When users (professors) review plagiarism detection results they may provide the feedback which in turn could vary from star based ranking of the results to suggestion of the links that would be a better "match" for plagiarized paper.

There have not been many projects that assume a collaborative approach to tackle the web plagiarism detection problem. A noticeable project named CPDNet (Collaborative Plagiarism Detection Network) was proposed as a research prototype in 2009-2010 [16]. The project was tailored around the idea of having the network of nodes each of which would maintain its own segment of the database with the documents that have been previously submitted to the system for the checkup. Internet search capacity was listed in the project but problem of crawling scalability was not actually discussed. CPDNet site is not accessible since 2011 and there is no additional information available on the project continuity. Although the CPDNet proposed collaborative network of nodes, its main idea was to distribute the Index component of the search engine. The proposed approach is significantly different as its aimed at using collaborative mechanisms to build distributed Crawler component using community resources.

Crowd crawling mechanism has been discussed as a tool to crawl information from social networks in the research communities [7]. The major difference in the proposed approach is that it targets unstructured web and therefore has to implement advanced mechanism to coordinate crawling efforts.

Based on the general concepts of community driven services and obvious advantages of distributed web crawling next section proposes a new approach to build PDS where the participants will assume part of the workload for two tasks: (a) web crawling; and (b) generating lists of seed sites to crawl.

3. PROPOSED COMPONENTS AND INFORMATION FLOW

General data flow model for proposed architecture is presented in Figure 1. There are three main roles in the proposed architecture:

- Centralized Coordination and Search Facility (CCSF). The main function for this unit is to provide indexing (search) service and coordinate crawling efforts. The approach similar to LiDi Crawl mentioned earlier [17] could be used to build CCSF. This unit should also provide quality assurance and user feedback mechanisms. The degree of centralization should follow patterns discussed earlier. Crawling assignments allocation should be centralized in the beginning of the project life cycle while some degree of freedom should be provided to independent crawlers once trust / reputation mechanism is established in the system. CCFS will have significantly different coordination mechanisms comparing to Task assignment (TAM) and Result collection modules (RCM) outlined in [7]. Discovering of additional links will be done by crawlers

not by RCM thus providing more freedom to the crawlers.

- Crawlers. This unit represents the core of crowdcrawling architecture. Its main function will be to obtain seed sites from the CCSF, perform the crawling and feed crawling results back to CCSF. As industry surveys show many organizations use infrastructure which has been acquired to be capable to cope with peak load but average utilization rarely exceeds 5% [13]. It means that large institutions should have enough capacity to run a virtual crawling appliance in spare CPU cycles. Usually Internet service providers do charge clients based on the allocated bandwidth therefore crawling in the off-peak (night) time will not be affecting bandwidth required to perform normal duties. The distributed nature of crawling efforts has an additional advantage in terms of protection from blocking of crawling efforts. Paper mills that distribute plagiarized papers will not be able to block crawler by IP address because there will be no single centralized crawler. The Crawler box could be provided to project participants as a virtual machine as discussed below. Some open source projects supported by Apache Foundation such as Apache Nutch crawler (http://nutch.apache.org/) could be used for this purpose.

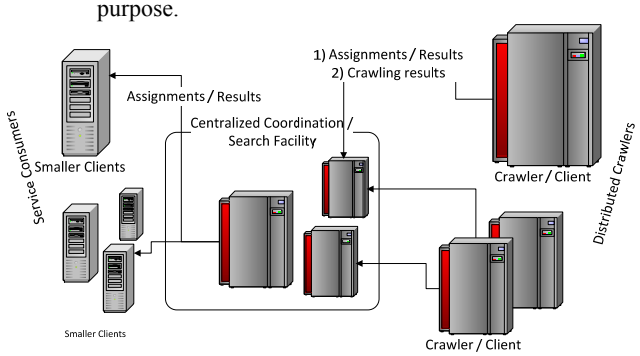

Fig. 1. Information flow in the proposed architecture.

- Clients. These are the community members who use the service but they do not have enough spare resources to participate in crawling efforts. Clients are not passive elements: they will serve as a vehicle for subject matter experts (professors) to suggest potential sources of plagiarism.

Obviously the architecture makes sense only if members of the community are willing to contribute some resources to serve as Crawlers. Extended list of crowdsourcing success factors includes such items as motivation and participant capabilities [9]:

- Motivation: Free access to PDS should motivate institutions to participate in the network. Similar to crawling of social networks [7] certain level of participation in crawling efforts will be required to get access to the system. Given the fact that Crawling could be scheduled in the off-peak hours, lack of CPU and bandwidth resources should not prevent educational organizations from joining the community.

- Capabilities: the need to have additional technical expertise may have a potentially negative influence on the participation decision. This issue can be addressed by reducing the level of technical expertise required to setup a crawler on the participant's infrastructure. Potential solution would be to prepare off-the-shelf virtual appliances with crawling tools with all required settings. In this case the potential participant needs only to put downloaded virtual appliance into any popular hypervisor such as VMware or Hyper-V and start it.

A small experiment was conducted to test the potential solution for the capability issue. Virtual appliance based on CentOS Linux was prepared with 2GB RAM / 30GB HDD and preloaded with Apache Nutch™ crawler and set of scripts. Scripts have been tailored to automate the following three tasks: (1) load initial crawling seeds from simulated CCFS; (2) start crawling; and (3) feed the results back to CCFS database. System administrator with no previous experience with Nutch software has been asked to follow simple instructions on how to load appliance in the VirtualBox hypervisor and schedule scripts for the off-peak time. The administrator was able to complete the setup in less than two hours. The appliance was running for five days in the night time. On average the crawl speed was ~ 35,000 documents per one 6 hour night crawling cycle. The average download error was about 3%. Ten paper mills were used as initial seeds for the experiment. These test indicated that participants' capability issue could be addressed with properly prepared launch-ready virtual appliance which does not consume bandwidth or CPU power in the off-peak time. Moreover similar resources could be rented as low cost virtual private servers for less than $50 per month. General outline provided above have some potential limitations. Next section discusses some of them.

4. DISCUSSION

The outlined architecture for crowdcrawling based PDS has good potential to overcome the need for expensive outsourcing of internet search for web plagiarism detection. It brings together the power of distributed web crawlers and community based quality control. There are three factors that define the potential success of community based systems: trust, motivation, and engagement. Trust in the results of plagiarism detection will be defined by trust in the community as a whole. It will take some time to build reputation based trust for the proposed system. Additional verification and arbitration mechanisms should be engaged on the CCSF to verify the content from community crawlers. Overlapped crawling described in the next paragraph could be used for this purpose. Motivation will be defined by access to either free or inexpensive plagiarism detection service. Engagement will be done on the two levels. First will be organizational level that provides feedback from the CCSF to the crawler. On the personal level additional mechanism should be considered to encourage teachers to contribute to the list of the resources to be crawled. Such mechanisms may include reputation and ranking system that would promote personal participation.

There are some potential limitations to this approach. These limitations could be placed into two major groups: misbehavior of community members and attacks from external users. Deliberate misbehavior from community members may be expressed by members who are trying to submit low quality or biased or alternated content that does not match the actual content of the web sites they crawled. To address the issue some arbitration mechanisms should be embedded in CCSF. One of such mechanisms could be overlapped crowdcrawling. It would allow CCSF to compare results obtained by independent crawlers and make quality decisions based on embedded rating mechanism.

Also results ranking mechanisms proposed earlier should be of the help here.

If we put aside such common attacks on the web services as DDoS then deliberate attacks from the external users may include endless loops on the crawled content and web spam. Most of the crawlers have some protection mechanisms to address the issue of looping content. Plus web site size limitations could be added to avoid this attack. Issue of web spam could be addresses by connection with community driven blacklisting services such as Spamhouse (www.spamhouse.org). Attacks specific to text plagiarism such as changed character encoding or text obfuscation could be addressed in the Indexing component of the system (CCSF). For example, measures on this stage may include such mechanism as language and character encoding consistency checkup, text style analysis, etc.

CCSF should have enough capacity to handle indexing of web content. Eventually the database of crawled content may become too big to handle by the conventional search engine. Especially this could be an issue if PDS will be configured to preserve the evolution of web resources. In this case some form of distributed indexing could be implemented instead of centralized search engine. Such development could potentially lead to some kind of crowd-based indexing similar to the one developed in CPDNet project [16]. If these issues are carefully addressed the proposed architecture can serve as blueprint for the community driven PDS.

5. CONCLUSION

The crowdcrawling mechanism proposed in this paper focuses on the issue of scalability of crawling efforts for the internet plagiarism detection projects. Novelty of the proposed approach includes use of community spare resources belonging to participating organizations to perform distributed web crawling instead of one distributed entity being responsible for the job. Community members will contribute underutilized IT resources to the project in form of distributed crawlers in exchange of access to the service. Additional contribution from the community may include suggestions of seed sites for potential sources of plagiarism. Three major roles outlined in the paper include Coordinator, Crawler, and Client. Potential obstacles and attacks on the service have been outlined and linked to the possible solutions.

REFERENCES

[1] Anand A., Bedathur S., Berberich K., Schenkel R., and Tryfonopoulos C. EverLast: a distributed architecture for preserving the web. In Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries (JCDL '09). ACM, New York, NY, USA, 331-340. (2009)

[2] Baeza-Yates R. and Ribeiro-Neto B. Modern Information Retrieval. AddisonWesley, (1999)

[3] Baeza-Yates R., Castillo C., Junqueira F., Plachouras V., and Silvestri F., Challenges in distributed information retrieval, in International Conference on Data Engineering (ICDE), Istanbul, Turkey: IEEE CS Press (2007).

[4] Boldi, P., Codenotti, B., Santini, M. and Vigna, S., UbiCrawler: a scalable fully distributed Web crawler. Softw: Pract. Exper., 34: 711–726. doi: 10.1002/spe.587 (2004)

[5] Cambazoglu B. B., Plachouras V., Junqueira F., and Telloli L. On the feasibility of geographically distributed web crawling. In Proceedings of the 3rd Int. Conf. on Scalable information systems (InfoScale '08). ICST, Brussels, Belgium, Article 31 , (2008)

[6] Costa F. and Frasconi P. Distributed community crawling. In Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters (WWW Alt. '04). ACM, New York, NY, USA, 362-363. DOI=10.1145/1013367.1013476 (2004)

[7] Ding, C., Chen, Y., & Fu, X.. Crowd crawling: towards collaborative data collection for large-scale online social networks. In Proceedings of the first ACM conference on Online social networks (pp. 183-188). ACM.

[8] Exposto J., Macedo J., Pina A., Alves A., and Rufino J.. Efficient partitioning strategies for distributed web crawling. In Proceedings of the 21st International Conference on Information Networking, (2007).

[9] Geiger D., Seedorf S., Schulze T., Nickerson R., and Schader M., Managing the Crowd : Towards a Taxonomy of Crowdsourcing Processes, Proceedings of the Seventeenth Americas Conference on Information Systems, Detroit, Michigan August 4th-7th 2011, pp. 1-11, (2011).

[10] Kc M., Hagenbuchner M., and Tsoi A.C. A Scalable Lightweight Distributed Crawler for Crawling with Limited Resources. In Proceedings of the 2008 IEEE/WIC/ACM Int. Conference on Web Intelligence and Intelligent Agent Technology - Volume 03 (WI-IAT '08), Vol. 3., 663-666.

[11] Liu S., Xu X., Li D., Zhang W., and Liu X. A GNP-Based Scheduling Strategy for Distributed Crawling. In Proceedings of the 2009 International Conference on Web Information Systems and Mining (WISM '09). IEEE Computer Society, 651-655. (2009)

[12] Maurer, H., F. Kappe, and B. Zaka. Plagiarism - A Survey. J. of Universal Computer Science, 12(8): 1050-1084. (2006)

[13] SERVER VIRTUALIZATION with Advanced Management Retrieved on March 27, 2010 from http://download.microsoft.com/download/F/5/D/F5DDFB8C-86C5-486A-85BF-A15773C1FF52/Server_Virtualization_Datasheet.pdf (2010)

[14] Shkapenyuk V, Suel T. Design and implementation of a high-performance distributed web crawler. IEEE International Conference on Data Engineering (ICDE), 2002. IEEE Computer Society, (2002).

[15] Underwood, J., Szabo, A. Academic offences and e-learning: Individual propensities in cheating. British Journal of Educational Technology, 34 (4), pp. 467-477. (2003)

[16] Zaka, B. Empowering Plagiarism Detection with a Web Services Enabled Collaborative Network. J. of Information Science and Engineering, 25, 1391–1403. (2009).

[17] Zhu K., Xu Z., Wang X., and Zhao Y.. A full distributed web crawler based on structured network. In Proceedings of the 4th Asia information retrieval conference on (AIRS'08). Springer-Verlag, Berlin, Heidelberg, 478-483. (2008)