On Bias-free Crawling and Representative Web Corpora

Roland Schäfer
Freie Universität Berlin
Habelschwerdter Allee 45
14195 Berlin, Germany
roland.schaefer@fu-berlin.de

Abstract

In this paper, I present a specialized open-source crawler that can be used to obtain bias-reduced samples from the web. First, I briefly discuss the relevance of bias-reduced web corpus sampling for corpus linguistics. Then, I summarize theoretical results that show how commonly used crawling methods obtain highly biased samples from the web. The theoretical part of the paper is followed by a description my feature-complete and stable *ClaraX* crawler which performs so-called Random Walks, a form of crawling that allows for bias-reduced sampling if combined with methods of post-crawl rejection sampling. Finally, results from two large crawling experiments in the German web are reported. I show that bias reduction is feasible if certain technical and practical hurdles are overcome.

1 Corpus Linguistics, Web Corpora, and Biased Crawling

Very large web corpora are necessarily derived from crawled data. Such corpora include COW (Schäfer and Bildhauer, 2012), LCC (Goldhahn et al., 2012), UMBC WebBase (Han et al., 2013), and WaCky (Baroni et al., 2009). A crawler software (Manning et al., 2009; Olston and Najork, 2010) recursively locates unknown documents by following URL links from known documents, which means that a set of start URLs (the *seeds*) has to be known before the crawl. Diverse crawling strategies differ primarily in how they queue (i. e., prioritize) the harvested links for download. A typical real-world goal is to optimize the queueing algorithm in a way such that many good corpus documents are found in the shortest possible time, in order to save on bandwidth and processing costs (Suchomel and Pomikálek, 2012; Schäfer et al., 2014).

Such an efficiency-oriented approach is reasonable if corpus size matters most. However, the goals of corpus construction might be different for many corpora intended for use in corpus linguistics. Especially in traditional corpus linguistics, where forms of *balanced* or even *representative* corpus design (Biber, 1993) are sometimes advocated as the only viable option, web corpora are often regarded with reservation, partly because the sources from which they are compiled and their exact composition are unknown (Leech, 2007). Other corpus linguists are more open to web data. For example, in branches of cognitively oriented corpus linguistics where the *corpus-as-input* hypothesis is adopted—e. g., Stefanowitsch and Flach (2016 in press)—, nothing speaks against using large web corpora. Under such a view, corpora are seen as reflecting an average or typical input of a language user. Consequently, the larger and thus more varied a corpus is, the better potential individual differences between speaker inputs are averaged out.

Even under such a more open perspective, corpus designers should make sure that the material used for a web corpus is not heavily biased. Naive crawling can lead to very obvious biases. For example, Schäfer and Bildhauer (2012, 487) report that in two large-scale crawls of the *.se* top-level domain, the Heritrix crawler (Mohr et al., 2004) ended up downloading 75% of the total text mass that ended up in the final corpus from a single blog host. The final corpus was still 1.5 billion tokens large, and seemingly large size does thus *not* prevent heavy crawling bias in web corpora, as the Swedish web most certainly does not consist of 75% blogs.

Apart from such immediately visible problems (which, admittedly, can be solved by relatively simple countermeasures) there are structural and

Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task, pages 99–105,
Berlin, Germany, August 7-12, 2016.

hard to detect biases introduced by all variants of the ubiquitously used *breadth-first search* (BFS) crawling algorithm.[1] As theoretical work has shown, BFS is biased towards web pages that have a high *in-degree*, i. e., pages to which many other pages link (Achlioptas et al., 2005; Kurant et al., 2010; Maiya and Berger-Wolf, 2011). It follows that crawling algorithms used for corpus construction so far do not give each page the same chance of being sampled. They do not perform *uniform random sampling*, and it is mathematically impossible to correct for BFS bias post-hoc.

Although the problem has been mentioned sporadically in the web-as-corpus literature, for example by Ciaramita and Baroni (2006, 131) or Schäfer and Bildhauer (2013, 29–34), nobody has ever tried to investigate whether such fundamental biases pose a problem. As of today, it is simply unclear whether even corpus linguists of the more permissive type (w. r. t. corpus composition) can rely on web corpora as being good samples of the whole text mass on the web.[2] Thus, retrieving unbiased (and thus technically speaking *representative*) samples from the web is not only important for fundamental research, but it might ultimately help to improve the acceptance of web corpora in corpus linguistics. I want to point out that the term *representative(ness)* in the remainder of this paper is used in a purely statistical—i. e., sampling-theoretic—way: a web corpus is representative of the documents on the web if each page had the same chance of being sampled.[3]

In this paper, I mainly describe the features and configurability of an open-source crawler which can be used for bias-corrected sampling from the web. I also show some preliminary results from the analysis of large experimental crawls in the German-speaking segment of the web. In Section 2, I briefly discuss crawling algorithms which allow for the (partial) correction of crawling biases. The system description of the crawler follows in Section 3. Finally, I present the experimental results in Section 4.

2 Methods for Bias Correction

In the theoretical literature, algorithms for bias-free crawling have been proposed. When considering such algorithms, it is vital to understand that the web forms a *directed graph* and that all crawlers implement a strategy by which they explore this graph. The web pages are the *nodes* of the graph, and each link from one page to another forms an *edge*. Any web crawler moves from node (page) to node by following edges (links), and it consequently implements a graph search algorithm (like BFS). The web graph is *directed* (and not *undirected*) because links cannot be followed backwards.[4]

It has been suggested by Henzinger et al. (2000) and Rusmevichientong et al. (2001) that bias-free samples can be obtained from directed graphs by applying Random Walk algorithms (RW) instead of BFS. See also the summary in Schäfer and Bildhauer (2013, 29–34). A RW jumps from page to page by randomly selecting exactly one outgoing link, following it, and discarding all others. No additional restrictions are imposed on the walker's search path, and thus revisits of pages seen before are conceptually desired.[5] A subtype of the RW algorithm reserves a certain probability at each step of jumping to a random URL instead of following a link.[6] Fundamental results show that RW crawling is also biased, but in a way that we can correct for.

[1] The simplest BFS prioritizes harvested links in the order that they were harvested. Optimizations usually depart slightly from BFS and add mechanisms by which those links receive higher priority which promise to lead to better content according to some metrics.

[2] I want to point out in passing that Google searches are most likely not an appropriate method of obtaining unbiased samples from the web, especially because we have no way of knowing how Google selects and sorts search results. Biber and Egbert (2016, 9) call their corpus based on Google queries 'representative' but at the same time admit that the sampling method does not guarantee representativeness. See Kilgarriff (2006) or Schäfer and Bildhauer (2013, 6–7) for summaries of why Google is not a good choice for sampling corpus documents.

[3] While such samples might ultimately not be the optimal samples for certain specific research questions, they are clearly required in order to establish a basis for any further (informed/stratified) sampling. A common example in introductory statistics courses teaches students that obtaining a sample for an opinion poll at the convention of a single party is useless for predicting the outcome of an election, no matter how large the sample is. It would be highly biased without any chance of correcting the bias through additional stratification. The work presented here will ultimately help to make sure that our web corpus sampling procedures do not suffer similar fatal biases.

[4] Technically, because a page i can have n_{ij} links pointing to any page j (with $n_{ij} \in \mathbb{N}_0$), the web graph is a *network*, and n_{ij} is the *weight* of the edge between i and j.

[5] This is very different in efficiency-oriented crawling, where a lot of effort is invested into avoiding revisits.

[6] For all practical applications, the random URL has to be taken from a very large database of known (thus pseudo-random) links.

Essentially, RWs sample pages with a probability that is dependent on their *PageRank*. The PageRank (Brin and Page, 1998) is a well known metric and essentially a generalization of the indegree. See the accessible summary in Bryan and Leise (2006). While the exact PageRank of each page can only be calculated if the whole graph is known, Henzinger et al. (2000) show that a page's PageRank can be estimated from the number of times a long RW revisits the page. Bias correction is then just a matter of applying a form of *rejection sampling* to all pages visited by the RW (the *biased sample*): by sampling pages from the biased sample with a probability inverse to their estimated PageRank, one can create an *unbiased sample*. Rusmevichientong et al. (2001) show that Henzinger's rejection sampling method, while strongly alleviating the bias, does not remove it completely because the PageRank estimation is inexact. They suggest a modified algorithm which increases the precision of the estimation by performing additional independent RWs originating from each node of the original RW (for mathematical details see their paper).

The crawler described in Section 3 can be used for both types of bias correction. However, preliminary results reported in Section 4 show that only Henzinger's algorithm might be feasible for web crawling, and even that only with certain modifications.

3 An Experimental Random Walker

In this section, I describe a highly configurable experimental crawler called *ClaraX* that performs random walks through the web graph: a *walker* rather than a crawler. I call it *experimental* because it is intended for experiments and fundamental research, not for the construction of large web corpora. The software is feature-complete and stable, compiles on GNU/Linux and OSX, and it is made available (including the source code) under a maximally permissive 2-clause BSD open-source license.[7]

3.1 Crawling Architecture

The basic crawling strategy implemented in the walker is a simple RW. In other words, the walker walks from document to document, always following a single randomly selected outgoing link from the current document, discarding all other links. Consequently, it starts with a single seed URL. A *random jump probability* can be specified, in which case a file with a list of seed URLs must be passed. The walker will then jump to a random link from the list instead of following a link from the current page with the specified probability. The walker implements all essential crawler functionality. This includes

- URL scope restriction via regex
- URL block regexes
- politeness restrictions (including *robots.txt*)
- obfuscation through User-Agent forging and randomized waits
- web page caching
- HTTP time-out control
- crawl step limit/maximal path length

The basic URL selection scheme is simply random selection of one link from each page (see Section 2). However, for practical reasons, the walker can be configured to follow

- links to entirely different hosts
- links to different virtual hosts (such as *www.host.com* and *forum.host.com*)
- links to the exact same host
- any combination of the above

Further URL selection is implemented based on the integrated post-processing described in Section 3.2. If the walker jumps to a page which turns out to be too short, too bad in terms of text quality, written in the wrong language, etc., then the walker can be set to discard this step and try another random link from the *previous* page. This effectively allows users to define sub-graphs of the web graph which the walker should explore.

Finally, the walker offers ways of dealing with dead ends. A dead end is reached when a page does not contain any links, or if all outgoing links from a page have been tried but none of the linked documents fulfilled the defined criteria. Since a RW always follows a single non-branching path through the web graph, it cannot continue from such a page. In this case, a *forced jump* to another seed URL can be performed, or the walk can be terminated. Alternatively, the walker can *backtrack*. This means that it follows its own path backwards and tries alternative paths.[8]

[7] `https://github.com/rsling/texrex`

[8] Theoretically, the walker would ultimately find the longest possible path beginning at the initial seed URL by

3.2 Built-in Processing and Output Formats

The walker integrates a full post-processing tool chain consisting of diverse modules, such as an HTML stripper, a UTF-8 converter and NFC normalizer, a boilerplate detector, and a language detector/text quality evaluator based on frequencies of function words. The post-processing modules are re-used from the previously developed *texrex* software (Schäfer and Bildhauer, 2012; Schäfer et al., 2013; Schäfer, 2016b; Schäfer, 2016a). The walker documents the progression of the RW in a short and a long file format. Python scripts are available which convert these files to JSON, allowing anyone to easily read in the data. Also, the original HTML documents are stored in a subset of the ISO WARC file format.[9] Furthermore, a processed clean corpus is stored in the simple (but fully well-formed) XML that is also used for the COW corpora. Finally, in order to locate near-duplicate documents in the resulting corpus, w-shingles (Broder, 2000) are stored in separate files for later analysis with included tools.

4 First Experiments

In this section, I present results from two experiments performed using the walker described in Section 3. For both experiments, the walker was configured to:

- walk only within the top-level domains *.at*, *.ch*, and *.de*, which are associated with countries where German is the (or one of the) major official languages
- only proceed if the documents found were written in German
- obfuscate the fact that it was a crawler, transmitting a false User-Agent header and not respecting *robots.txt*
- be very polite with a minimal wait of 10 seconds between requests to a host
- use a list of over 15 million seed URLs extracted from the large German DECOW14 web corpus

In other words, the experiments relate to the sub-segment of the web that can be called the *German-speaking web*.

using backtracking. Given the size and complexity of the web graph, however, backtracking can only be used effectively combined with a relatively low maximal desired path length.

[9] `http://www.iso.org/iso/catalogue_detail.htm?csnumber=44717`

Steps	Host
91,442	www.vsw-news.de
40,806	pauls-blog.over-blog.de
35,787	fielders-choice.de
34,411	www.my-bikeshop.de
34,091	www.bremer-treff.de
24,769	www.deutscher-werkbund.de
24,114	www.vau-niedersachsen.de
24,096	www.icony.de
22,299	www.discover.de
20,093	www.dewezet.de

Table 1: The 10 longest RW segments spent on a single host during the first experiment

Exper.	Runtime	Steps	Hosts	St./Host
1	12.75d	1,093,047	1,227	890.83
2	25.36d	2,090,443	204,053	10.25

Table 2: Key figures for the two experiments

4.1 Link Structures on the Web

The first experiment was a baseline experiment intended to establish how web pages and web hosts link to each other, allowing an estimation of the feasibility of any subsequent sampling experiments. The walker was configured to follow *any* link, including host-internal links. The essential numbers are reported in Table 2. While the average number of steps made before the walker jumped to a new web host was as low as 16.42, the walk often bounced back and forth between two or three hosts which strongly linked to each other, leading to an average 890.83 documents per host in the whole experiment. The 10 longest single-host segments of the RW are shown in Table 1.

These results are not surprising because it is known that web hosts strongly link internally, and that there is strong linking within clusters of hosts, not necessarily but often for purposes of search engine optimization. What this experiment establishes is that we cannot perform naive RWs jumping from page to page and expect bias correction algorithms to work in any real-world web corpus creation scenario. Link structures between single pages are so pathologically biased that we would have to crawl for much longer than feasible. What seems more appropriate than page-level bias correction is host-level bias correction, to which I turn in the next section.

4.2 Host Walking and Bias Reduction

The first experiment showed that just following any link makes RWs practically useless. In the second experiment, the walker was therefore configured to follow only links leading to *different hosts*. This changes the interpretation of the web graph as explored by the walker: it is viewed as a graph composed of hosts (not pages) as nodes. Furthermore, the random jump probability was set to 0.1, making sure that the walker could not get stuck between neighboring hosts of a link farm, etc. The essential figures are reported in Table 2. Compared to the first experiment, the average number of pages per host drops dramatically from 890.83 to 10.25.

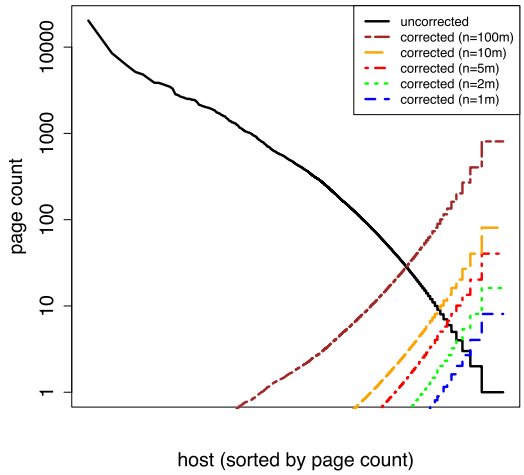

Figure 1: Number of pages (y) visited in the second experiment per host (x), sorted in decreasing order, and the theoretically expected document counts when applying Henzinger's rejection sampling method depending on the targeted bias-reduced corpus size, given as n; log-log axes

I then projected the expected corpus sizes and the per-host probabilities for the rejection sampling process. The logic behind these projections is that aggressive rejection sampling can easily lead to a situation where hosts with a high PageRank receive a near-zero probability of being sampled from the crawl and making it into the final corpus. Figure 1 shows the expected page counts per host in the biased and bias-corrected corpora if a final corpus of a specific size is desired. The lines for the bias-corrected corpora show the expected number of pages per host that would be retained after naive and aggressive bias-correction. For example, if we target a bias-reduced corpus of 1 million documents, most of the very prominent hosts from the original RW receive an extremely low probability of being sampled from the walk. On the other hand, hosts which had a very low document count in the original RW would have to contribute more documents than we actually have. If we perform the rejection sampling such that hosts which were visited only once during the original RW contribute (on average) one document to the bias-corrected corpus, we can only keep approximately 125,000 documents in total, in which case the 108,523 most prominent hosts are (on average) not represented at all in the bias-corrected corpus. In other words, a RW with 2 million steps is too short for aggressive rejection sampling, which only goes to show how strong the bias in the original walk is.

5 Outlook

The type of experiment described in Section 4.2 appears suitable for the creation of web corpora which are representative samples of the population of web documents. However, we obviously need to run much longer RWs, and we need to perform simulations on artificial graphs in order to test how well less aggressive (but more practically feasible) bias-reduction works, which would enable us to retain more documents in the rejection sampling step.

Apart from implementing these steps, I will also explore the effects of bias reduction on the composition of web corpora through automatic classification of the documents in the resulting corpora, for example by content and register.[10] This will finally make it possible to compare different methods of crawling (BFS as used for the COW corpora and bias-corrected RWs) in terms of the linguistically relevant effects on corpus composition that they might have.

Acknowledgments

My research presented here was funded by the German Research Council (Deutsche Forschungsgemeinschaft, DFG) through grant SHA/1916-1 *Linguistic Web Characterization*.

[10] See also Schäfer and Bildhauer, this volume.

References

Dimitris Achlioptas, Aaron Clauset, David Kempe, and Cristopher Moore. 2005. On the bias of traceroute sampling: or, power-law degree distributions in regular graphs. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, STOC '05, pages 694–703, New York, NY, USA. ACM.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky Wide Web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Douglas Biber and Jesse Egbert. 2016. Using grammatical features for automatic register identification in an unrestricted corpus of documents from the open web. *Journal of Research Design and Statistics in Linguistics and Communication Science*, 2:3–36.

Douglas Biber. 1993. Representativeness in corpus design. *Literary and Linguistic Computing*, 8(4):243–257.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International World Wide Web Conference*, pages 107–117. Elsevier Science.

Andrei Z. Broder. 2000. Identifying and filtering near-duplicate documents. In R. Giancarlo and D. Sanko, editors, *Proceedings of Combinatorial Pattern Matching*, pages 1–10, Berlin.

Kurt Bryan and Tanya Leise. 2006. The $25,000,000,000 eigenvector: The linear algebra behind Google. *SIAM Review*, 48(3):569–581.

Massimiliano Ciaramita and Marco Baroni. 2006. Measuring web-corpus randomness: A progress report. In Marco Baroni and Silvia Bernardini, editors, *WaCky! Working papers on the Web as Corpus*, pages 127–158. GEDIT, Bologna.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May. European Language Resources Association (ELRA).

Lushan Han, Abhay L Kashyap, Tim Finin, James Mayfield, and Johnathan Weese. 2013. UMBC_EBIQUITY-CORE: Semantic Textual Similarity Systems. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics.

Monika R. Henzinger, Allan Heydon, Michael Mitzenmacher, and Marc Najork. 2000. On near-uniform URL sampling. In *Proceedings of the 9th International World Wide Web conference on Computer Networks: The International Journal of Computer and Telecommunications Networking*, pages 295–308. North-Holland Publishing Co.

Adam Kilgarriff. 2006. Googleology is bad science. *Computational Linguistics*, 33(1):147–151.

Maciej Kurant, Athina Markopoulou, and Patrick Thiran. 2010. On the bias of BFS (Breadth First Search). In *International Teletraffic Congress (ITC 22)*.

Geoffrey Leech. 2007. New resources or just better old ones? The Holy Grail of representativeness. In Marianne Hundt, Nadja Nesselhauf, and Carolin Biewer, editors, *Corpus linguistics and the web*, pages 133–149. Rodopi, Amsterdam and New York.

Arun S. Maiya and Tanya Y. Berger-Wolf. 2011. Benefits of bias: towards better characterization of network sampling. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 105–113, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2009. *An Introduction to Information Retrieval*. CUP, Cambridge.

Gordon Mohr, Michael Stack, Igor Ranitovic, Dan Avery, and Michele Kimpton. 2004. Introduction to Heritrix, an archival quality web crawler. In *Proceedings of the 4th International Web Archiving Workshop (IWAW'04)*.

Christopher Olston and Marc Najork. 2010. *Web Crawling*, volume 4(3) of *Foundations and Trends in Information Retrieval*. now Publishers, Hanover, MA.

Paat Rusmevichientong, David M. Pennock, Steve Lawrence, and C. Lee Giles. 2001. Methods for sampling pages uniformly from the World Wide Web. In *In AAAI Fall Symposium on Using Uncertainty Within Computation*, pages 121–128.

Roland Schäfer and Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 486–493, Istanbul, Turkey. European Language Resources Association (ELRA).

Roland Schäfer and Felix Bildhauer. 2013. *Web Corpus Construction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool, San Francisco.

Roland Schäfer, Adrien Barbaresi, and Felix Bildhauer. 2013. The good, the bad, and the hazy: Design decisions in web corpus construction. In Stefan Evert, Egon Stemle, and Paul Rayson, editors, *Proceedings of the 8th Web as Corpus Workshop (WAC-8)*, pages 7–15, Lancaster. SIGWAC.

Roland Schäfer, Adrien Barbaresi, and Felix Bildhauer. 2014. Focused web corpus crawling. In Felix Bildhauer and Roland Schäfer, editors, *Proceedings of the 9th Web as Corpus workshop (WAC-9)*, pages 9–15, Stroudsburg. Association for Computational Linguistics.

Roland Schäfer. 2016a. Accurate and efficient general-purpose boilerplate detection for crawled web corpora. *Language Resources and Evaluation*. Online first: DOI 10.1007/s10579-016-9359-2.

Roland Schäfer. 2016b. CommonCOW: Massively huge web corpora from CommonCrawl data and a method to distribute them freely under restrictive EU copyright laws. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4500–4504, Portorož, Slovenia. European Language Resources Association (ELRA).

Anatol Stefanowitsch and Susanne Flach. 2016, in press. A corpus-based perspective on entrenchment. In Hans-Jörg Schmid, editor, *Entrenchment and the psychology of language: How we reorganize and adapt linguistic knowledge*. De Gruyter, Berlin.

Vít Suchomel and Jan Pomikálek. 2012. Effcient Web crawling for large text corpora. In Adam Kilgarriff and Serge Sharoff, editors, *Proceedings of the seventh Web as Corpus Workshop*, pages 40–44.