The Downside of Markup: Examining the Harmful Effects of CSS and Javascript on Indexing Today's Web

Karl Gyllstrom
Computer Science Department
Katholieke Universiteit Leuven
Leuven, BE
karl.gyllstrom@cs.kuleuven.be

Carsten Eickhoff
Delft University of Technology
Delft, NL
c.eickhoff@tudelft.nl

Arjen P. de Vries
Centrum Wiskunde & Informatica
Amsterdam, NL
arjen@acm.org

Marie-Francine Moens
Computer Science Department
Katholieke Universiteit Leuven
Leuven, BE
sien.moens@cs.kuleuven.be

ABSTRACT

The continued development and maturation of advanced HTML features such as *Cascading style sheets* (*CSS*), *Javascript*, and *AJAX*, as well as their widespread adoption by browsers, has enabled web pages to flourish with sophistication and interactivity. Unfortunately, this presents challenges to the web search community, as a web page's representation in the browser (i.e., what users see) can diverge dramatically from its raw HTML content (i.e., what search engines index and retrieve). For example, interactive pages may contain content in regions that are not visible before a user action, such as focusing a tab, but which are nonetheless still contained within the raw HTML. We study this divergence by comparing raw HTML to its fully rendered form across a number of metrics spanning presentation, geometry, and content, using a large, representative sample of popular web pages. We find that a large divergence currently exists, and we show via a historical analysis that this divergence has grown more pronounced over the last decade. The general finding of our study is that continuing to index the web via simple HTML parsing will diminish the effectiveness of retrieval on the modern web, and that the IR community should work toward more sophisticated web page processing in indexing technology.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]; H.3.3 [**Information Storage and Retrieval**]

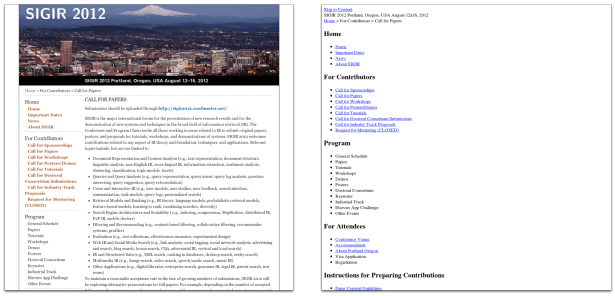
Keywords

Web, HTML, indexing, rendering

1. INTRODUCTION

Browsers and web technology grow more sophisticated each day, allowing users a deeper and more interactive experience with web sites. Many current web sites rival desktop applications in functionality, from online productivity suites to music players; and increasingly powerful design control has removed most limits on aesthetics for web authors. Although this trend benefits both producers and consumers of web content, it poses a challenge to the web retrieval community: as *Javascript* and *CSS* provide tools and abstractions to designers, they create a level of indirection between the raw, indexable web (as defined by the HTML contents) and the final presentation to users. We believe that neglecting rendering effects on web indexing represents an important gap in the state of the art.

We define *post-rendering* as effects on the content and presentation of a web page that are not explicitly encoded in the raw HTML. In this paper, we limit this definition to *Javascript* and *CSS*, though there are a number of other likely contributors, including Adobe Flash, Microsoft's Silverlight, and the forthcoming HTML5 and CSS3 technologies. For example, Figure 1 depicts the top 1000 pixels of two renderings of the web page for SIGIR 2012's call for papers; the first with full *Javascript* and *CSS* enabled, and the second with neither enabled. The former is the representation we would expect from a modern web browser, while the latter is an approximation of a much older browser – or one equivalent to the parsing capabilities of modern search tools like Lucene. These representations are quite different, visually, and as we show in this paper, they also in how their content is indexed and retrieved by search tools. These differences are growing more pronounced over time.



(a) Fully rendered (b) No *CSS* or *Javascript*

Figure 1: The web page for the SIGIR 2012 call for papers, rendered with and without *Javascript* and *CSS*. Even the header image is not loaded until the *CSS* is executed (*zoomable*).

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

Figure 2 depicts the inclusion of *post-rendering* over time. We observe a substantial increase in scripting, both in terms of inline and imported scripts. Imported style sheets significantly increase in usage frequency, while inline style sheets experience only a modest increase. As we show in subsequent experiments, this trend correlates with the divergence between rendered and non-rendered content across a number of measurements.

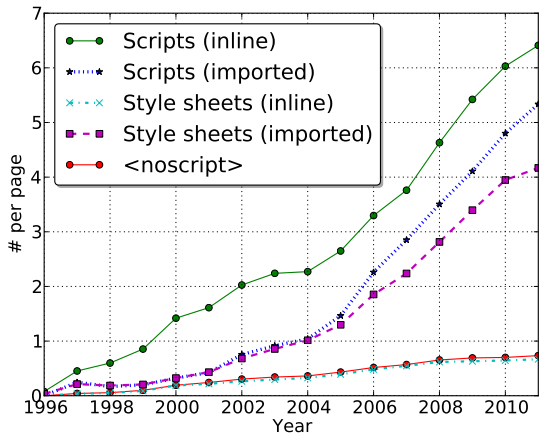


Figure 2: Use of *CSS* and *Javascript* over time on D_{alexa}

By ignoring *post-rendering*, we face two major problems: First, we can *get things wrong*. For example, as we show later, *post-rendering* can modify what content from the HTML is actually presented to the user; in other words, we risk indexing content that the user never sees (false positive), and missing content that the user would see (false negative). Second, we lose the ability to promote secondary properties of web documents. For example, in retrieval contexts, we may want to more heavily weight content appearing on the front page of web sites, or content that is displayed in emphasized/large text. As we show in this paper, these are qualities that are becoming increasingly difficult to accurately measure without considering *post-rendering* effects.

2. BACKGROUND AND RELATED WORK

Cascading Style Sheets (*CSS*) complement HTML by allowing the associating of style and presentation attributes to elements. For example, *CSS* enables a web developer to specify the font size for the text contained within an element or the absolute position to be occupied by the element on the web page. Explicitly this can be handled via the "style" attribute within elements, but typically it is done implicitly: style attributes are defined for a general class, and elements are assigned to classes. While this allows abstraction and automation for web developers, it adds an additional level of indirection between HTML parsers/browsers and content; in other words, to assess the final effects of *CSS* requires some form of HTML/*CSS* processing, a core component of the rendering process.

A core step of the indexing process is parsing content. Terrier and Indri are two popular examples of state-of-the-art research search toolkits that contain HTML parsers. These parsers, however, are limited to the removal of tags and the extraction of the text content within them. As we show in this paper, this approach may be at a risk of becoming insufficient; browsers, HTML, *Javascript*, and the technologies built upon them, are all maturing such that a page's rendered form can be quite different than what is expressed via the tags.

A complementary and related problem lies within the ClueWeb09 dataset, which has been heavily useful and influential for the information retrieval community. Although it provides a large, representative snapshot of the web, it lacks imported *Javascript* and *CSS* files; as we show later, this inhibits our ability to render – and hence, thoroughly process – web pages in retrieval experiments.

Much previous work has addressed the non-uniformity of web pages, which we believe is likely to be further affected by rendering. Some work has addressed structural/visual aspects of pages via the Document Object Model (DOM), including segmenting pages into coherent regions [3], and cleaning data [4]. Other work has approached web pages as a collection of structural fields (e.g., body, title) with distinct language models [5]; as we show later, these regions may be more difficult to identify via tags alone, and our retrieval results provide further evidence of the advantages of field-based retrieval approaches.

3. APPROACH

In this paper, we measure the effects of *post-rendering* by comparing fully and partially rendered forms of web pages within a large dataset. Furthermore, we conduct a comparison across the historic versions of pages to show general trends. We explore the effects of *post-rendering* on available content, presentation, and layout/positioning (Section 4).

3.1 Technical overview

In traditional HTML indexing, a web page is processed top down, with elements and their internal content extracted. Elements can be excluded based on their informativeness; for example, the `<script>` tag encapsulates Javascript to be executed by the browser's compiler, rather than content shown directly to the user. By contrast, in our approach, we first render the page to an internal buffer, from which we extract the DOM tree. As with HTML parsing, this gives us an element-by-element view of the page, but we can also capture aspects that are unavailable from the tags alone, such as the position occupied on the page, or whether or not the element is visible at load time; it also gives us access to content that is not available in the raw HTML (e.g., that is loaded dynamically via an AJAX call at load time), and gives us a "final" view of the data (e.g., how the element will appear after *CSS* and *Javascript* have been executed). We use the Qt web toolkit for this approach[1], which enables us to load a web page by URL and acquire programmatic access to the DOM. Qt's underlying rendering engine is Webkit, upon which modern browsers such as Chrome and Safari are built. Furthermore, it provides the ability to isolate the effects of *post-rendering* by controlling whether or not *Javascript* and *CSS* are applied during the rendering process.

For each page, we configured the virtual browser window to the dimensions of 1028×768. Historically, this has been

[1] `http://qt.nokia.com/`

the most common screen size of browsers, and is still highly prevalent today [2].

3.2 Datasets

As mentioned previously, ClueWeb09 is an insufficient dataset for our purposes due to its lack of *CSS* and *Javascript* files; as shown in Figure 2, modern pages generally import many style sheets and script files, which can heavily influence their rendered appearance. As an alternative, we undertook a large crawl of popular web pages.

We define D_{alexa} as our primary dataset. This dataset was constructed from the list of most heavily trafficked web sites, as listed by Alexa.com, an Amazon-owned site metrics company. We sampled heavily from the top 250,000 most frequently accessed websites on the web[3]. This collection has a very high coverage of the web; we measured the top 10,000 pages in terms of portion of global web traffic, finding that they account for over 60% of all web traffic. In total, we crawled and rendered more than 150,000 web pages from this collection.

We sought to diversify this set, as Alexa's list of URLs is limited to base sites, rather than individual pages. Hence, it is biased toward the front page of websites. To mitigate this, we created the $D_{WebTrack}$ data set, which represents a collection of search results to user queries. We drew the 150 queries defined in the TREC Web Track 2009-2011 [2], for which we collected 300 search results per query from the Bing search engine. We crawled and rendered a large sample from this pool: slightly more than 38,000 pages.

A useful property of Alexa is that it also contains, for most of the web pages in this list, the most popular queries through which web users find the pages. This allowed us to build query/document pairs, which we applied in several of the experiments we perform.

For these sets, we also collected historic information using the Wayback archive[4]. This archive enables the traversal of previous states of a given URL. For each URL in the data sets, we executed a history crawl on the archive for an entry from each year, keeping approximately the same month each year where available, which we call W_{alexa}. This dataset contains over 250,000 pages, spread over 15 years.

4. TRENDS IN ADVANCED MARKUP

In this section, we measure the divergence between raw HTML and its fully rendered form across a number of metrics spanning content, presentation, and geometry. We define two forms of rendering for our experiments: *Partial*, where only the raw HTML is rendered; and item *Full*, where both *Javascript* and *CSS* are rendered.

Generally, we consider *Full* to be *fully rendered* pages (i.e., equivalent in representation to that of the average web browser), while the other forms are *partially rendered*. We emphasize that *Partial* is not the same as using the raw HTML, as it is still rendered (albeit with no *post-rendering* effects); hence, there is still a layout of elements; font sizes are established, etc, but it serves as a useful surrogate for raw HTML.

[2] `http://gs.statcounter.com/#resolution-ww-yearly-2008-2012`

[3] `http://s3.amazonaws.com/alexa-static/top-1m.csv.zip`

[4] `http://www.archive.org/web/web.php`

In the following sections, we cover 3 categories in which the divergence between fully and partially rendered pages surfaces. In Section 4.1, we cover a case in which the visible content on pages is affected by *post-rendering*. In Section 4.2, we show how presentation is affected, with a study on the effects of *post-rendering* on font size and variability. Finally, in Section 4.3, we describe how layout is affected; namely, how *post-rendering* alters where content ultimately appears on a page.

4.1 Content: The (In)visible Text of Pages

Interactive pages often feature content which is only made visible after interaction with the user. For example, a tabbed interface allows users to change the visible content by selecting new tabs. Nonetheless, the entirety of the page contents – both visible and not – is typically still contained within the HTML. Hence, though the content is available to a crawler/indexer, it is not necessarily immediately visible to the user. On the other hand, there are cases in which content is visible to the user without actually being contained within the HTML. An extreme, but increasingly common, example is a site loading content from a server via AJAX calls. In other cases, *Javascript* can alter content in such a way that it presents differently to the user. Consider the ClueWeb09 document for `http://www.mahalo.com/abo-obama/`, which is a match for the Web Track topic "obama family tree". The query terms appear as a link in the raw HTML, but after JavaScript rendering, they do not appear – the Wiki-based *Javascript* actually converts the link into a numeric reference link, in which the terms are not shown. Hence, content that is visible to the indexers is not actually visible to users.

We measured the extent to which this problem has presented in web pages over time. First, we measured the amount of page text that is not technically visible at the point when the page is loaded, depicted in Figure 3. This is reported as the portion of the total page text contained within invisible regions. We observe a growing divergence among the four forms of rendering, especially between *Full* and *Partial*. The comparative stability of *Full* leads us to believe that page authors are increasingly relying upon *post-rendering* to achieve the same page functionality.

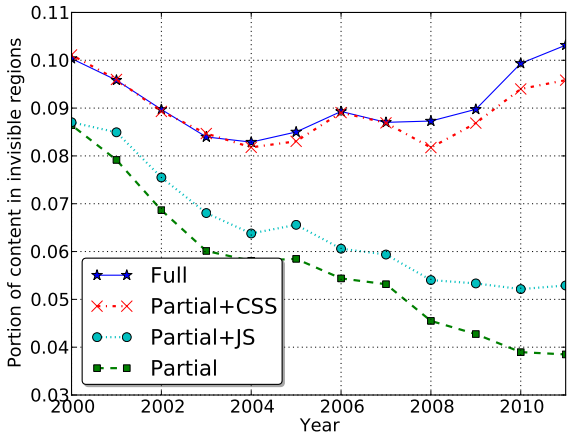


Figure 3: "Invisible" text per page, as a portion of total text on page.

We further examined how this divergence affects *relevant* content. For each page, we measured the portion of the page's total occurrences of query terms appearing in invisible content. Recall that for D_{alexa}, we have query data as provided by Alexa.com, while for $D_{WebTrack}$, we have the queries for which we executed Bing searches. We observed a similar trend to that depicted in Figure 3. Not only is *Partial* failing to differentiate between visible and invisible content, but much of this content contains query terms, and hence is more likely to be relevant.

These data indicate that invisible content is becoming a more integral part of web pages. This presents a number of consequences for web indexers. Primarily, it risks false positives: search systems match queries to documents containing query terms; if those terms only appear in initially invisible content (i.e., content which will only be made visible by user actions) this can create confusing results for users, as they could be presented web results that do not apparently contain the query terms. Indeed, indexers today are more vulnerable to this problem than ever.

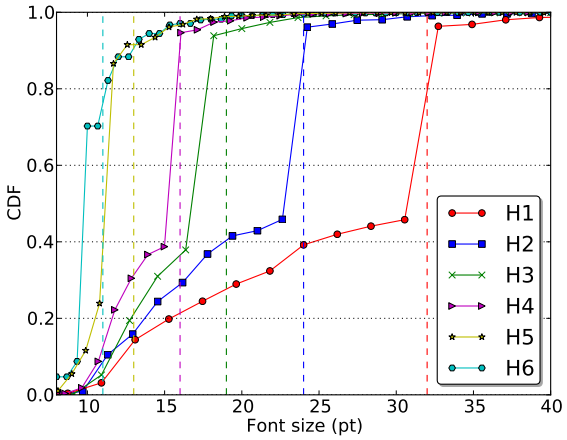


Figure 4: CDFs of font sizes for HTML headers when *CSS/Javascript* rendering is used (D_{alexa}). Vertical lines indicate the default size for the header when rendered on our settings.

4.2 Presentation: Text size variability and prominence

In this section, we measure the effects of rendering on the ultimate font size of text content, finding that rendering is increasingly important for determining it. In particular, we show how web pages are increasingly depicting prominent text via style rather than explicit headers. These findings will present challenges to search engines that apply font size in assessing content importance, such as Google [1].

4.2.1 Header variability

The HTML tags `<H1, H2, ..., H6>` are used to indicate headers of various sizes (with `<H1>` being the largest). One challenge is that *CSS* and *Javascript* can alter these sizes, even, if desired, causing a "smaller" header (e.g., `<H6>`) to be rendered larger than a "larger" header (e.g., `<H2>`). Headers are useful to study because they represent an explicit markup by the web author: the author is, in essence, delineating content as titles and subtitles, much in the same way that this subsection's title "*Header variability*" is more prominent than its contained text. *Post-rendering* presents the challenge that web designers can now more easily recreate prominent text without header tags; hence we lose the ability to extract such information.

We measured font sizes across header tags, depicted as CDFs by font size in Figure 4. Clearly, advanced markup allows page authors to introduce substantial variability within header tags; for example, approximately 50% of `<H1>` tags font sizes are different than the default – primarily smaller. This is a sign that *post-rendering* is already widely applied. Conversely, the smaller headers are often rendered at sizes larger than their defaults. The variability is generally high for all headers. Additionally, there is overlap among the headers; for example, around 20% of `<H1>`s, 30% of `<H2>`s, and 40% of `<H3>`s are rendered at 16*pt* or smaller.

Although this comparison is inter-page and not intra-page, we believe the variability is nonetheless remarkable. It is evidence that *CSS* affords such considerable control over headers that, in practice, their application is subsequently widely variable. For example, if any header can be easily rendered at any relative size, there is less reason for web authors to use headers conventionally (e.g., to only use `<H2>` tags as subtitles to `<H1>` tags). Consequently, indexers' assumptions about importance (e.g., that `<H2>` is generally very prominent and hence important) are increasingly undermined by this decoupling. Indeed, as we see in the next section, evidence indicates that web authors are decreasingly relying upon headers to indicate prominent titles and content.

4.2.2 Do headers still predict prominent content?

Historically, header tags were used to indicate text to be presented larger and to displace sections. However, as shown in the previous section, styles can be altered such that headers exhibit high variability in sizes – and can even violate their intended order (e.g., by allowing an `<H1>` tag to be rendered smaller than an `<H2>` tag). Given the additional presentation control available to web page authors, it seems that we would witness a trend toward decoupling of headers and their relative font sizes.

To measure this, we extracted the *prominent text* from each page, which we define as any text rendered at a font size that is at least one standard deviation above the mean font size for the page. We then measured the portion of this text that was drawn from headers. Intuitively, this tells us how much of the most prominent page text is actually drawn from header tags (*explicit*), rather than normal content that is rendered large from external *CSS* and *Javascript* actions (*implicit*).

Indeed, the trend is toward a looser coupling between headers and *prominent text*. Figure 5 depicts, for *Full* and *Partial*, the portion of pages' *prominent text* that is drawn from header tags. *Full* shows a looser coupling, as a greater proportion of the largest page text is drawn from tags that are not headers. Notably, this divergence has expanded over the previous decade. This divergence is also present when we isolate content containing query terms, indicating that it is impacting not only content in general, but also the most relevant page content.

We offer two main consequences of these findings. First, since font size can be used to indicate prominence, as larger text is more obvious than smaller text, it is valuable for

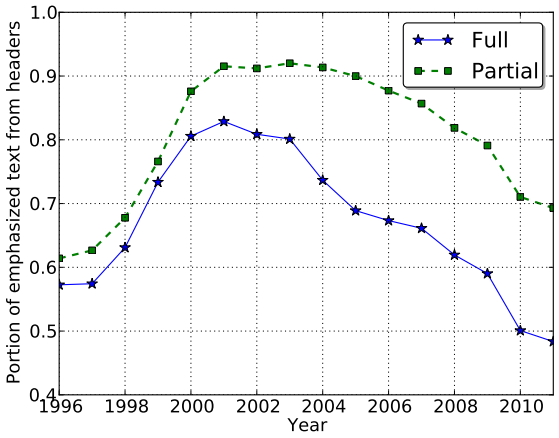


Figure 5: Portion of *prominent text* drawn from headers. The current trend is that a greater portion from *Full* is drawn from non-headers.

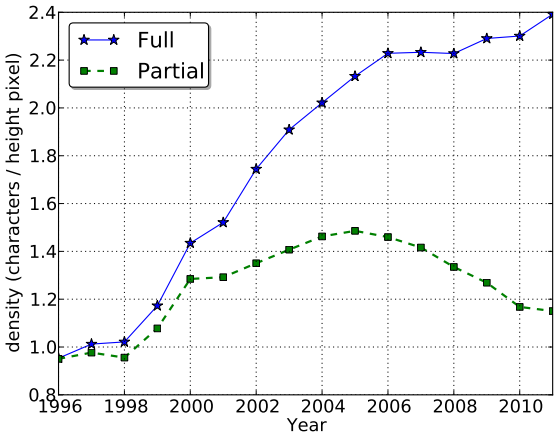


Figure 6: Content density, defined as character per height pixel.

retrieval systems to consider. However, it is growing more difficult to extract relative size information from raw HTML. Hence, this potentially valuable source of information is diminishing in usefulness as we lose the ability to assess it. The second consequence is that this is further evidence of a growing rift in the representation of pages between *post-rendering* and raw HTML.

4.3 Content density and the front page

When rendering without *post-rendering*, modern pages are typically stretched vertically and lose compactness. Consequently, elements that are intended to be presented on the front page may not appear until subsequent pages, which, if accessed by the user, would require scrolling (Figure 1 depicts such a scenario).

We define *content density* as the amount of visible text within the screen space, measured by the number of characters divided by the page height (in pixels). As shown in Figure 6, we observe an upward trend in general page density until around 2005; this is likely caused by web authors simply adding more content in general. At 2005, the density begins to drop for *Partial*, while still growing for *Full*; we attribute this to a transition of layout control from raw HTML to *post-rendering*.

5. CONCLUSIONS

We explored the problem of divergence between rendered and raw HTML content across several metrics, producing a number of relevant findings: The use of scripts and style sheets has dramatically increased in the past decade (Section 1), and a growing amount of HTML content is not immediately rendered to users; this content is also relevant, in terms of containing a relatively higher density of query terms (Section 4.1). Font size variability increases with *post-rendering*, with explicit headers being displaced by *post-rendering* to control prominent text, as more relevant content is appearing in prominent text which is not drawn from headers (Section 4.2). Page density is not effectively captured without *post-rendering*, while it is generally more difficult to determine the position of elements without *post-rendering*, and these regions contain relevant content (Section 4.3).

Perhaps none of these findings, alone, constitutes an urgent problem; we also emphasize that we did not identify a comprehensive set of divergence metrics. Rather, they are intended to be illustrative and intuitively grounded. Importantly, they all provide evidence of a rift between raw and rendered web content. Of particular importance is the fact that *this rift seems to be expanding*; as the web grows more advanced, our ability to index it by current methods declines. If the recent trends continue, this is likely to become a significant problem for the web retrieval community.

6. REFERENCES

[1] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW '98*, pages 107–117, Amsterdam, The Netherlands, 1998. Elsevier Science Publishers B. V.

[2] C. Clarke, N. Craswell, and I. Soboroff. Overview of the trec 2009 web track. In *Proceedings of TREC 2009*, 2010.

[3] D. Fernandes, E. S. de Moura, A. S. da Silva, B. Ribeiro-Neto, and E. Braga. A site oriented method for segmenting web pages. In *SIGIR '11*, pages 215–224, New York, NY, USA, 2011. ACM.

[4] F. Sun, D. Song, and L. Liao. Dom based content extraction via text density. In *SIGIR '11*, pages 245–254, New York, NY, USA, 2011. ACM.

[5] K. Wang, X. Li, and J. Gao. Multi-style language model for web scale information retrieval. In *SIGIR '10*, pages 467–474, New York, NY, USA, 2010. ACM.