Measuring the Quality of Experience of Web users

Measuring the Quality of Experience of Web users

Enrico Bocchi
Telecom ParisTech
enrico.bocchi@polito.it

Luca De Cicco
Politecnico di Bari
luca.decicco@poliba.it

Dario Rossi
ENST
dario.rossi@enst.fr

ABSTRACT

Measuring quality of Web users experience (WebQoE) faces the following trade-off. On the one hand, current practice is to resort to metrics, such as the document completion time (onLoad), that are simple to measure though knowingly inaccurate. On the other hand, there are metrics, like Google's SpeedIndex, that are better correlated with the actual user experience, but are however quite complex to evaluate and, as such, relegated to lab experiments.

In this paper, we first provide a comprehensive state of the art on the metrics and tools available for WebQoE assessment. We then apply these metrics to a representative dataset (the Alexa top-100 webpages) to better illustrate their similarity, differences, advantages and limitations. We next introduce novel metrics, inspired by Google's SpeedIndex, that (i) offer significant advantage in terms of computational complexity, (ii) while maintaining a high correlation with the SpeedIndex at the same time. These properties makes our proposed metrics highly relevant, and of practical use.

Keywords

Quality of Experience; DOM; onLoad; TTFB; TTFP; Above-the-fold; SpeedIndex; ByteIndex; ObjectIndex; MOS

1. INTRODUCTION

In the Internet, Web is still King. The browser has indeed become the preferential platform through which a plethora of services can be accessed, including but not limited to search, entertainment, productivity, business, social and personal communication, etc. At times of important evolutions –from HTTP/1 protocol family to HTTP/2, SPDY and QUIC– having reliable ways to compare protocol performance becomes crucial before massive deployments can take place [18].

A number of studies have pointed out the importance of *delay*, and their direct relationship to the value of these business – for example, Amazon [2] and Google [12] report losses in the 0.6-1.2% range for delay increasing by 0.4-1 sec, whereas Shopzilla [12] reported an +12% revenue increase for a 5 sec reduction of onLoad after a major site redesing. The hidden correlation between these factors is of course the impact that the delay has on the Quality of Experience of Web users (WebQoE): the higher the delay, the lower the WebQoE, the worse the experience, the higher the likelihood of user disengagement, the larger the economic losses.

While the existence of a relationship between delay and WebQoE is beyond any doubt, it is however more difficult to precisely pinpoint the delay of "which" event is the most important during the lifetime of a Webpage, and to furthermore map it to a quantized quality level (≈MOS). Indeed, Webpages have grown to quite complex entities including hundreds of objects hosted, which are fetched opening several tens of connections directed to multiple domains. Requests for such objects are often dynamically generated by JavaScripts (or related technologies) executed as part of the page construction process.

It thus appears obvious that *no single event* –from the time at which the first byte is received (TTFB), to time at which the first object is painted on the browser (TTFP), to the parsing of the Document Object Model (DOM), to the completion of the full page (onLoad)– can express all sort of intricate dependencies [22] between the rendering process and the user experience. As such, there have been proposals for new metrics that are better suited to capture the actual quality of experience of Web users, such as for instance Above-the-fold [14] and SpeedIndex [1], which have both been recently proposed by Google (in 2011 and 2012 respectively). The SpeedIndex is particularly interesting as it explicitly considers delay of *all events* in a Webpage lifetime, but have so far been limitedly used due to its computational complexity.

As a result, the (regrettable) state of the art in today WebQoE evaluation, both in the research sphere [16, 22, 21, 23] as well as in industry practice [3, 4, 5], is still to express QoE via the page completion time, i.e., onLoad. For instance, Alexa [5] reports the onLoad and directly exposes quantiles of the delay, while Google uses onLoad delay to rank search results [3], although with a small weight [6].

Our contributions in this paper are as follows. We first provide a complete taxonomy of the existing WebQoE metrics and tools, and introduce our proposed generalization of the SpeedIndex (Sec.2). We next present a comprehensive illustration of all WebQoE metrics on the top-100 Alexa webpages, further elucidating relationships among metrics and experimental methodologies. Notably, we show that (i) an indetermination principle emerges when computing the

Table 1: Metrics to express user perceived quality

	Metric name	Layer 3	Layer 4	Layer 7	Unit/ Range	Description
Time Instant	TTFB	≈	✓	✓	*sec*	Time at which the first byte of payload is received
	DOM	-	-	✓	*sec*	Time at which the Document Object Model (DOM) is loaded
	TTFP	-	-	✓	*sec*	Time at which the first object is painted
	OnLoad	-	-	✓	*sec*	Time at which all bytes of payload have been received
	ATF [14]	-	-	✓	*sec*	Time at which the content "above the fold" has been rendered
Time Integral	ObjectIndex	-	-	✓	*sec*	Integral of complementary object-level completion
	ByteIndex	≈	≈	✓	*sec*	Integral of complementary byte-level completion
	SpeedIndex [1]	-	-	✓	*sec*	Integral of complementary visual progress
Comp. Score	YSlow [13]	-	-	✓	[0,100]	Yahoo's compound score (23 weighted heuristics)
	PageSpeed [11]	-	-	✓	[0,100]	Google's PageSpeed Insight heuristics
	dynaTrace [10]	-	-	✓	[0,100]	dynaTrace's compound score
	MOS	-	-	-	[1,5]	User rating

SpeedIndex as its computation alters the nature of the experiment, and that (ii) our proposed metrics remains highly correlated to the SpeedIndex despite their simplicity (Sec.3). We finally discuss further generalization of these metrics that would allow one to embed psycho-behavioral models of user spatio-temporal perception at limited cost (Sec.4).

2. WebQoE METRICS

Tab. 1 reports the most prominent metrics to measure Web user QoE (WebQoE). In particular, the table groups metrics in four categories: ① **Time-instant metrics**, which are computed by measuring the time instant a particular event occurs. ② **Time-integral metrics**, that are computed by integrating over all events of a given type tracked during the evolution of a page progress; in this category fall the two metrics proposed in this paper, namely the ByteIndex and the ObjectIndex, which generalize the SpeedIndex proposed by Google. ③ **Compound scores**, weighting altogether several domain-expert heuristics, to yield a score in the range [0,100]. For the sake of completeness, the table also reports the ④ **Mean Opinion Score (MOS)**, computed by averaging users' subjective ratings. MOS can be regarded as a benchmark for the other metrics, but it is admittedly hard to collect MOS points. For this reason, we disregard it in what follows, leaving it as future work.

2.1 Time-instant metrics

Metrics belonging to this category have the clear advantage of being easily measurable, since they only track the occurrence of a specific event. As a consequence, metrics in this category are widely used nowadays. Nonetheless, they are arguably simplistic since they disregard the complex chain of events that triggered the measured event. In a nutshell, such metrics compress the whole waterfall chart [1] to a single time instant. Intuitively, two different experiments having the same time-instant metric could be associated to significantly different user experiences. Despite this, the onLoad (also known as Page Load Time, PLT), which measures the time taken to completely load all the objects of a page, is still considered as the main KPI in the vast majority of recent scientific work, from both the industrial [3, 4, 5] and the academic [16, 22, 21, 23] perspectives.

Other interesting metrics in this category include the TTFB, i.e., the time instant at which the first byte of payload is received (that expresses the page reactivity) and the DOM event, i.e., the time at which the Document Object Model is completely downloaded and parsed (after which the rendering can start). Simple tracking of the visual progress is expressed by the TTFP which measures the time at which the first object is rendered. To further refine the tracking of visual progress, Google proposed the Above-The-Fold (ATF) metric, which is defined as the time the content shown in the visible part of the webpage is completely rendered.

It is important to notice that only few of these metrics, such as the TTFB and onLoad, can be measured at the network (L3) or transport (L4) layers, whereas the vast majority –for instance all those related to render events– mandates the instrumentation of the web browser (L7) for their measurement. Additionally, it is worth to notice that while most of the metrics in this category require few computation (if any), ATF is significantly more complex as it requires to take screenshots during the rendering process, as well as a post-processing stage of the captured frames.

2.2 Time-integral metrics

Metrics in this category are characterized by the explicit use of all events in the webpage waterfall. In particular, Google introduced the SpeedIndex in 2012, in order to consider the whole process leading to the visual completion of a webpage to better account for user experience.

[1] `http://chimera.labs.oreilly.com/books/1230000000545/ch10.html#RESOURCE_WATERFALL`

In this paper, we generalize such metric and we introduce the family of *time-integral* metrics, defined as follows:

$$X = \int_0^{t_{\text{end}}} (1 - x(t))dt \tag{1}$$

where X is the value of the metric, t_{end} is the time the last event is triggered, and $x(t) \in [0, 1]$ is the time evolution of the progress to reach such event. Fig. 1 illustrates computation of a time-integral metric, where the blue line represents $x(t)$, and the gray-shaded area represents the result of the integral (1). Trivially, the smaller the area above the curve $x(t)$, the lower the score X, the better the user experience.

In order to make a concrete example, let us consider the SpeedIndex [1]. In such a case $x(t) = \text{painted}(t)/\text{total}$ is the progress of the rendering process, and t_{end} corresponds to the ATF time-instant metric that marks the completion of the rendering. Under this light, the rationale of (1) is simple: not all the sub-events, i.e., the rendering of specific objects, are considered equally important. In particular, (1) gives *more weight to objects being rendered at the beginning* and vanishingly less weight to the objects rendered towards the end. In other words, such metric assigns a lower score to pages (or web browsers) rendering as much content as possible in the beginning with respect to pages (or browsers) rendering all the objects near $t \approx t_{\text{end}} = \text{ATF}$.

Bounds of time-integral metrics. It is immediate to notice that the time-integral metric X defined by (1) is lower-bounded by t_{TTFB} and upper-bounded by t_{end}.

Consider indeed Fig. 1, and observe that $x(t)$ is a monotonically increasing function 0 at $t = 0$ and equal to 1 at $t = t_{\text{end}}$. Hence, the worst case time-integral metric is obtained when $x(t) = \mathbb{1}_{t \geq t_{\text{end}}}$ where $\mathbb{1}$ is the indicator function: with such a progress function, all the work is done in correspondence to the event of interest t_{end}. In this case X is the area of the rectangle of base t_{end} and height 1, i.e. $X = t_{\text{end}}$, which implies $X \leq t_{\text{end}}$.

Conversely, the best case is obtained when all the work is done at the beginning. Notice that in practice, regardless of the considered time-integral metric, no progress whatsoever can be done before the first byte of payload (TTFB) is received by the web browser. Thus, the best case scenario is obtained when $x(t) = \mathbb{1}_{t \geq \text{t}_{\text{TTFB}}}$, which corresponds to the area of the rectangle with base TTFB and height 1, which implies $X \geq \text{t}_{\text{TTFB}}$.

Relationship to time-instant metrics. Extending the above reasoning, it follows that any time-instant metric t_X can be considered as the upper bound of the time-integral metric having $t_{\text{end}} = t_X$, or in other words time-instant metrics can be considered as *projections* of the corresponding time-integral metric. Particularizing this observation to Google's ATF and SpeedIndex proposals, we have that $\text{t}_{\text{TTFB}} \leq SI = \int_0^{\text{ATF}} (1 - x(t))dt \leq \text{ATF}$, which shows that time-integral metrics allow for a much more fine grained measure.

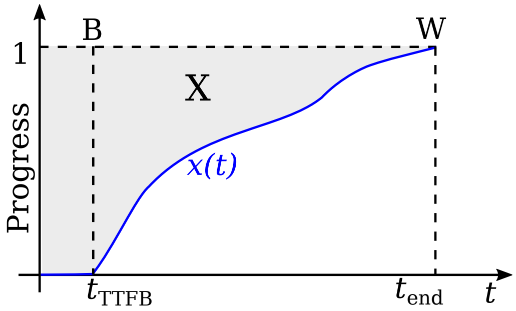


Figure 1: Time-integral metrics computation

Proposed metrics. The way the SpeedIndex metric is actually computed [7] is to take snapshots, by default at a frame rate equal to 10fps, of a web browsing session. Such video frames compose a filmstrip which is analyzed in order to produce the visual completion fraction $x(t)$. More specifically, the color histogram of each frame is computed and compared to the histogram of the last frame, which represents the webpage at rendering completion.

We however show in Sec. 3 that performing such operations burdens computational resources, significantly inflating the time needed to run the experiment, thus distorting it. To overcome such a limitation, we propose two metrics:

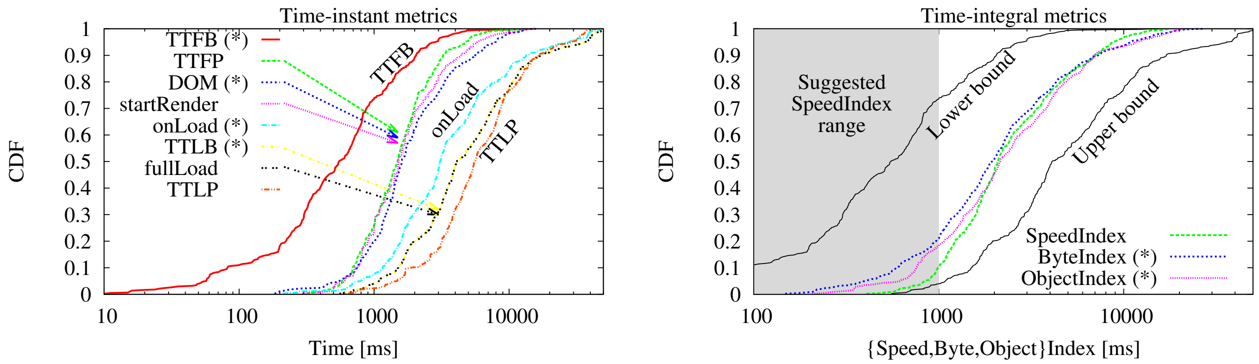
$$\begin{aligned} \text{ByteIndex} &= \int_0^{\text{onLoad}} (1 - x_B(t))dt \\ \text{ObjectIndex} &= \int_0^{\text{onLoad}} (1 - x_O(t))dt \end{aligned}$$

where $x_B(t)$ and $x_O(t)$ is the percentage of the objects and bytes retrieved at time t, respectively. Observe that both metrics require a negligible computational cost, as they can be computed by simply taking into account the time instants in which objects are fully downloaded. Finally, both the ByteIndex and the ObjectIndex can be considered as generalizations of the onLoad time-instant metrics.

The rationale of these metrics is to avoid complex visual rendering, and leverage the fact that objects received are directly (e.g., images) or indirectly (e.g., CSS) rendered by the browser. Second, as the SpeedIndex, these metrics take into account all webpage events, with a temporal-bias towards earlier events. Finally, ObjectIndex treats all objects equally, whereas ByteIndex introduces a spatial-bias as it implicitly states the size of an object to be correlates with its importance for the user (e.g, image vs CSS).

2.3 Compound metrics

Finally, compound scores such as Yahoo's YSlow [13], Google's PageSpeed Insights [11] and dynaTrace [10] encode expert knowledge, usually expressed as a set of heuristics (e.g., 23 in YSlow), combined with heterogeneous weights (e.g., 2% to 30% in YSlow). Such heuristics assess the effectiveness of a webpage design to: reduce computation (e.g.,



(a) (b)

Figure 2: Characterization of (a) time-instant and (b) time-integral metrics (Alexa top-100, WPT)

avoid CSS expressions, Alpha image load, avoid image scaling), speedup rendering (e.g., limit DOM elements, CSS at top, javascript at bottom), reduce data volume (e.g., compress data, minify javascripts and CSS, use small cookies), reduce delay (e.g., reduce DNS lookups, avoid redirect).

As such, while relevant to assess the effectiveness of the adopted webpage design –and indeed, generally used to measure progress/regression of webpages– these heuristics are unrelated to events timing, and hard to map to WebQoE. onload

3. WebQoE EXPERIMENTS

3.1 Methodology

We illustrate WebQoE metrics with an experimental methodology. We consider the top-100 Alexa webpages as this is a widely used benchmark in the industry [8] as well as academia [15]. Given that we perform experiments over the wild Internet, we expect variability across experiments due to load balancing, transient congestion, etc. Hence, we repeat the experiments 10 times for each page.

For the sake of simplicity, we consider a single browser. Since differences in rendering and processing engines across browsers can play a determinant role, we argue that this would also unnecessarily introduce complexity in the analysis. As such, we consider Google Chrome that has become by far the most popular browser, representing about half of the browser market share. We use both (i) and unmodified Google Chrome (CHR) browser, as well as the popular WebPageTest (WPT) that we have deployed on a local machine to orchestrate experiments. Notice that several metrics, including the SpeedIndex, are available only under WPT; conversely, {Object,Byte}Index can be computed over both WPT and CHR.

We run experiments from a single vantage point located in Paris, which corresponds to the case where CDN nodes are close. Additionally, we consider only the "desktop" version of each website. While the methodology, definition and metrics would apply to the mobile Web world as well, we believe that interactivity of the webpage plays an even greater importance in the mobile Web – which can be quite easy to convince of by considering that mobile webpages are designed to minimize the visual cluttering and reducing the time to perform a useful action. As such, putting mobile and desktop versions within the same basket would introduce bimodal behaviors in the metrics of interest, which we prefer to avoid.

3.2 Results at a glance

We start by showing in Fig. 2 (a) time-instant and (b) time-integral metrics early defined, of which we report their empirical cumulative distribution function, gathered with WPT. In the legend, a star symbol (*) denotes metrics that can be computed under both WPT and CHR.

Considering the *time-instant* first, it can be seen that, as expected [9], events have an order relationship: e.g., no paint (TTFP) can happen before the first byte is received (TTFB), parsing of the DOM is necessary for the rendering process to start and the reception of the full data (onLoad) can happen well before the last paint event (TTLP). It can also be seen that the TTFB and TTLP curves constitute the envelope of the process, and are separated by over two orders of magnitude, as they pertain to rather different activities. For each metric, it can also be noted a significant variance: the median DOM (onLoad) is about 1.5 (3) seconds, while the 90th percentile is above 5 (13) seconds. Finally, it can be observed that some groups of metrics appear to quite closely clustered (e.g., DOM, TTFP and startRender; TTLB and fullLoad) implying that there is some redundancy between the events definition and reporting.

Moving to the *time-integral* next, it can be seen that, as expected SpeedIndex, ByteIndex and ObjectIndex fall between the TTFB and TTLB envelopes. Additionally, these metrics are quite clustered, hinting to the fact that our simpler proposals have intrinsic similarities with the original SpeedIndex proposal. The dark-shaded region in the plot highlights

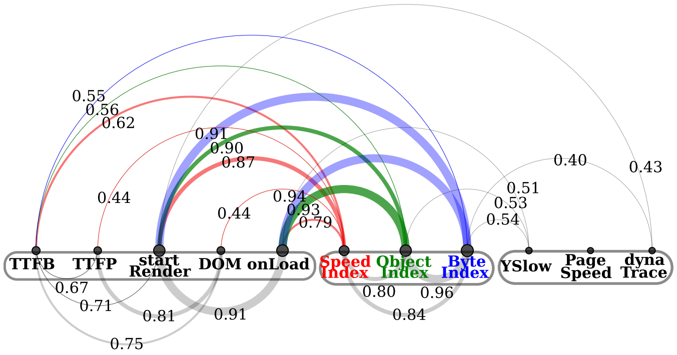


Figure 3: Arc diagram representing the correlation matrix between metrics pairs.

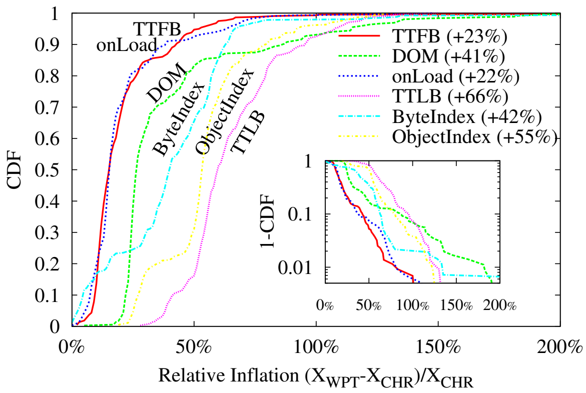


Figure 4: Relative inflation of time-instant & time-integral metrics under WebpageTest vs plain Chrome.

the zone of advised [17] SpeedIndex values for responsive websites: this hints to the fact that WPT slows down the whole rendering process (see Sec.3.4). A closer look reveals that ByteIndex and ObjectIndex *climb faster than SpeedIndex* in the short-time frame regime. This is due to the fact that (i) the completion ratio for {Byte,Object}Index increases even before the DOM event, and that (ii) {Byte,Object}Index neglect computational and render time, i.e., they consider byte/objects useful for the user experience as soon as they are received by the browser. Conversely, ByteIndex and ObjectIndex *climb slower than SpeedIndex* in the tail, as they consider objects that are possibly not painted (i.e., those that are below-the-fold). While in Sec.4 we discuss how it would be possible to fine-tune ByteIndex and ObjectIndex to even more closely approximate the SpeedIndex, we believe that the main takeway is instead their remarkable proximity.

3.3 Relationship among metrics

To further assess the relationship among metrics, we report in Fig. 3 the Pearson correlation matrix between metrics pairs, represented as an arc diagram. For completeness, we additionally consider scores that are popular in the industry such as Yahoo's YSlow, dynaTrace and Google's PageSpeed Insights (recall that these are compound scores weighting altogether a number of heuristics defined by domain experts) computed by [8] on the Alexa top-100.

In the plot, metrics are arranged into time-instant (left), time-integrals (middle), and compound scores (right). Correlations within group are reported in gray below the label name, while inter-group correlations are reported above. Correlations of the time integral group are highlighted in red (SpeedIndex), green (ObjetIndex) and blue (ByteIndex). For the ease of visualization, we only report correlations larger than 0.4, with actual correlation values annotated in the plot. To let the strongest correlation emerge, we quantize line width, doubling it every 0.1 correlation steps.

The picture reinforces the soundness of our proposal as it clearly appears that: (i) our proposed byte-level and object-level replacements do exhibit correlation with several time-events, similarly to the SpeedIndex – and as a results, our proposals are highly correlated with the SpeedIndex as well; (ii) the YSlow, dynaTrace and PageSpeed heuristics are poorly correlated among them (and with any other WebQoE metric) and thus do not represent valid alternatives.

3.4 Relationship among experiments

We finally contrast the same metrics gathered in WebPageTest (WPT) vs Chrome (CHR). Specifically, WPT computes a larger basket of metrics, notably including those related to rendering (TTFP, ATF, SpeedIndex, etc.). At the same time, computing such metrics affects the very same experiment: indeed, as recognized in the community [23], they require cumbersome screen captures and significantly slow-down[2] the rendering process.

To quantify the extent of the distortion in WPT vs CHR, we consider the subset of 6 metrics that can be computed in both, and define as $(X_{\rm WPT} - X_{\rm CHR})/X_{\rm CHR}$ the relative inflation of a generic metric X in the set. The cumulative distribution function of the relative inflation is depicted in Fig.4, annotating the average inflation for each metric in the label. A zoomed inset shows the complementary CDF, to better assess distortion in the tail. It can be seen that (i) inflation is non-linear (ii) average inflation ranges from +20% to +66%, (iii) in the worst 10% of the cases, the ObjectIndex is doubled (and so is the TTLB). Otherwise stated, distortion in the experiment introduced by computational complexity makes the SpeedIndex of little practical relevance.

4. DISCUSSION

In this paper we have provided a comprehensive view of the metrics available for WebQoE assessment, highlighting their merits, limits and dependencies by conducting experiments over the top-100 Alexa webpages. Our main contribu-

[2] The problem is that even if the SpeedIndex can be computed a posteriori, the screen capture process *itself* constitutes a significant CPU bottleneck

tion is to introduce a generalization of Google's SpeedIndex, which we instantiate into two very simple indexes, ObjectIndex and ByteIndex, having negligible computational complexity. Experimental results show that ByteIndex and ObjectIndex retain perceptual properties of SpeedIndex, without incurring its prohibitive computational complexity. This work opens a number of interesting future directions, which we now briefly discuss.

Closer SpeedIndex approximation. {Object, Byte}Index provides optimistic lower bounds to the SpeedIndex: this happens because {Object,Bytes}Index completion increase upon reception of any objects, including (i) those that are not painted (e.g., scripts) as well as (ii) those that take time to render (e.g., alpha images, complex CSS). Conditioning over content type (e.g., setting a null weight for scripts) would cope with (i), while taking into account execution times (e.g., no completion increase before DOM, estimation of time from reception to paint) could address (ii).

Psycho-behavioral model: content bias. Extending the above reasoning, it could be argued that taking explicitly into account object type or size could be worth investigating. For instance, for some object types a logarithmic reward can be expected from their byte-wise size (e.g., size of a JPG image encoded with higher quality may significantly increase, but the added value is likely sub-linear). Similarly, it may be argued that users perceive advertisement with a different value than content, which could be factored in by define a weight function $w_i = 1 - \mathbb{1}_{\text{Adblock}(i)}$ with $\mathbb{1}_{\text{Adblock}(i)} = 1$ whenever the domain name of object i belongs to the Adblock list. Finally, position of objects in the page (e.g., center vs corners) is likely to have an impact, so that geometry of the object/paint could be valuable to explicitly accounted for (unlike WebpageTest SpeedIndex, since it is based on histograms).

Psycho-behavioral model: time bias. The {Speed, Object, Byte}Index metrics do take into account the fact that not all paints/objects/bytes are equally useful, and thus give *implicitly* larger weight to those happening early in the Webpage lifetime. However, we believe that it would be interesting to *explicitly* controlling time dependence in reason of classic psycho-behavioral studies [19] (later adapted to the computer network domain [20]), which show a logarithmic separation of human perception timescales. This could be accounted for by integrating over t^{α} in (1): notice that, implicitly, the current SpeedIndex definition assumes $\alpha = 0$, and is thus a particular case of this larger family of metrics.

{Object,Byte}Index in-browser computation. Computation of our proposed metrics has been done offline from HAR archives. A useful addition of practical relevance would be to develope an in-browser version – of which we have a preliminary prototype which is however (i) limited to chrome and (ii) requires the NetDeveloper extension (in reason of HAR access). Extensions to other frameworks/browsers would be of course very useful to gather a more complete evaluation of the proposed metrics.

Correlation with MOS. Mean Opinion Score (MOS), obtained by experiments involving real users is an obviously missing, and utterly important, piece of this puzzle. Albeit challenging in nature, obtaining a corpus of HAR files annotated with user MOS would be an important contribution for the whole QoE community.

Large scale study. An obvious improvement of this work could then be to extend the characterization we conduct over the Alexa-100 dataset by either (i) considering pages beyond the top-100, or (ii) performing the same experiment by considering geographically-dispersed vantage points (e.g., Mlab nodes, PlanetLab nodes, Amazon EC2 nodes, etc.)

Acknowledgments

This work has been carried out at LINCS (`http://www.lincs.fr`). This work benefited from support of NewNet@Paris, Cisco's Chair "NETWORKS FOR THE FUTURE" at Telecom ParisTech (`http://newnet.telecom-paristech.fr`). Any opinion, findings or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of partners of the Chair.

5. REFERENCES

[1] https://sites.google.com/a/webpagetest.org/docs/using-webpagetest/metrics/speed-index.

[2] http://www.fastcompany.com/1825005/how-one-second-could-cost-amazon-16-billion-sales.

[3] https://googlewebmastercentral.blogspot.fr/2010/04/using-site-speed-in-web-search-ranking.html.

[4] http://googleresearch.blogspot.fr/2009/06/speed-matters.html.

[5] http://www.alexa.com.

[6] https://www.youtube.com/watch?v=muSIzHurn4U.

[7] http://webpagetest.org.

[8] http://www.showslow.com.

[9] https://www.w3.org/TR/2012/REC-navigation-timing-20121217.

[10] dynatrace. http://dynatrace.com/.

[11] Pagespeed insights. https://developers.google.com/speed/docs/insights/rules.

[12] Velocity and the bottom line. http://radar.oreilly.com/2009/07/velocity-making-your-site-fast.html.

[13] Yslow ruleset matrix. http://yslow.org/ruleset-matrix/.

[14] J. Brutlag. Above the fold time: Measuring web page performance visually, 2011.

[15] M. Butkiewicz, D. Wang, et al. Klotski: Reprioritizing web content to improve user experience on mobile devices. In *USENIX NSDI*. 2015.

[16] J. Erman, V. Gopalakrishnan, et al. Towards a spdy'ier mobile web? In *ACM CoNEXT*. 2013.

[17] P. Irish. Delivering the goods, 2014.

[18] M. Varvello, K. Schomp, D. Naylor, J. Blackburn, A. Finamore, and K. Papagiannaki. Is The Web HTTP/2 Yet? In *Passive and Active Measurement (PAM)*. 2016.

[19] R. B. Miller. Response time in man-computer conversational transactions. In *Proc. AFIPS Fall Joint Computer Conference*, pages 267–277. ACM, 1968.

[20] J. Nielsen. Response times: The 3 important limits, 1993.

[21] F. Qian, V. Gopalakrishnan, et al. Tm3: Flexible transport-layer multi-pipe multiplexing middlebox without head-of-line blocking. In *ACM CoNEXT*. 2015.

[22] X. S. Wang, A. Balasubramanian, et al. How speedy is spdy? In *USENIX NSDI*. Seattle, WA, 2014.

[23] Wang, Xiao Sophia and Krishnamurthy, Arvind and Wetherall, David. Speeding up web page loads with shandian. In *USENIX NSDI*.