The Rise of Chrome

Jonathan Tamary, Dror Feitelson

Since Chrome's initial release in 2008 it has grown in market share, and now controls roughly half of the desktop browsers market. In contrast with Internet Explorer, the previous dominant browser, this was not achieved by marketing practices such as bundling the browser with a pre-loaded operating system. This raises the question of how Chrome achieved this remarkable feat, while other browsers such as Firefox and Opera were left behind. We show that both the performance of Chrome and its conformance with relevant standards are typically better than those of the two main contending browsers, Internet Explorer and Firefox. In addition, based on a survey of the importance of 25 major features, Chrome product managers seem to have made somewhat better decisions in selecting where to put effort. Thus the rise of Chrome is consistent with technical superiority over the competition.

The Rise of Chrome

Jonathan Tamary Dror G. Feitelson
School of Computer Science and Engineering
The Hebrew University, Jerusalem, Israel

Abstract

Since Chrome's initial release in 2008 it has grown in market share, and now controls roughly half of the desktop browsers market. In contrast with Internet Explorer, the previous dominant browser, this was not achieved by marketing practices such as bundling the browser with a pre-loaded operating system. This raises the question of how Chrome achieved this remarkable feat, while other browsers such as Firefox and Opera were left behind. We show that both the performance of Chrome and its conformance with relevant standards are typically better than those of the two main contending browsers, Internet Explorer and Firefox. In addition, based on a survey of the importance of 25 major features, Chrome product managers seem to have made somewhat better decisions in selecting where to put effort. Thus the rise of Chrome is consistent with technical superiority over the competition.

1 Introduction

The most notable use of the Internet is the World Wide Web (WWW). The web was created by Tim Berners-Lee and his colleagues at CERN (The European Organization for Nuclear Research) in 1989. In order to consume information from the web, one must use a web browser to view web pages. The first web browser (which was in fact named WorldWideWeb) was developed at CERN as part of the WWW project [1]. But the first popular browser, which set the growth of the web in motion towards the wide use we see today, was Mosaic, which was developed by Marc Andreessen and Eric Bina at the National Center for Supercomputing Applications (NCSA) in 1993 [38].

The open nature of the web makes it possible for different browsers to co-exist, possibly providing different features, user interfaces, and operating system support. Over the years different browsers have competed for the user's choice. This has led to several "browser wars" — periods of fierce competition between different web browsers that are characterized by technological innovation and aggressive marketing, typically leading to the eventual dominance of one browser and the fall of another. In recent years we have witnessed such a shift (albeit somewhat protracted) from Microsoft's Internet Explorer to Google's Chrome.

Our goal is to explain why this has happened, and what are the implications for large-scale software development. In particular, we wanted to assess the technical quality of chrome and compare it with the quality of its rivals. To do so we downloaded all the version of Chrome, Firefox, and Internet Explorer that were released over a period of five years, and compared them using an array of benchmarks. As far as we know this is by far the widest study of its kind.

In a nutshell, we find that Chrome is indeed technically superior to other browsers according to most commonly-used benchmarks, and has maintained this superiority throughout its existence. Also, based on a survey of 254 users, the features pioneered by Chrome ahead of its competitors tend to be those that the users consider more important. Thus Chrome's rise to dominance is consistent with technical superiority. However, one cannot rule out the large effect of the Google brand and the marketing effort that was invested as factors that contributed greatly to the realization of this technical potential.

2 The Browsers Landscape

2.1 Browsers History

Not long after the release of the Mosaic web browser in 1993 it became the most common web browser, keeping its position until the end of 1994. The factors contributing to Mosaic's popularity were inline graphics, which showed text and graphics on the same page, and popularizing the point and click method of surfing. Moreover, it was the first browser to be cross-platform including Windows and Macintosh ports. Amazingly, by the end of 1995 it's popularity plummeted to 5% of the web browser market [26]. This collapse in it's popularity was concurrent to the rapid rise of Netscape Navigator which was released in December 1994 and managed in less than two years to reach around 80% market share (different sources cite somewhat different numbers).

Several factors are believed to have caused the fast adoption of Netscape by users. First, it was a natural followup of Mosaic as it was developed by the same people. Second, Netscape introduced many technological innovations such as on-the-fly page rendering, JavaScript, cookies, and Java applets [26]. Third, Netscape introduced new approaches to testing and distribution of web browsers by releasing frequent beta versions to users in order to test them and get feedback [40].

Netscape's popularity peeked in 1996 when it held around 80% market share. But in August 1995 Microsoft released the first version of Internet Explorer based on an NCSA Mosaic license. A year later, in August 1996, with the release of Internet Explorer 3, a browser war was on. By August 1999 Internet Explorer enjoyed 76% market share [39].

During this browser war it seems that Internet Explorer did not have any technological advantage over Netscape, and even might have been inferior. Therefore, other reasons are needed to explain Internet Explorer's success. One reason was that Netscape's cross platform development wasn't economical: instead of focusing on one dominant platform (Windows) it had approximately 20 platforms and this caused a loss of focus. Meanwhile, Microsoft focused on only one platform. Second, Microsoft bundled Internet Explorer with Windows without a charge, and as Windows dominated the operating systems market it was immediately available to the majority of users without any effort on their part. In an antitrust investigation in the U.S., Microsoft was found guilty of abusing its monopoly in the operating systems market by bundling Internet Explorer with Windows for free.

Once Internet Explorer was entrenched, it's market share grew due to a positive feedback effect. The standard tags used in HTML (Hyper-Text Markup Language, in which web pages are written) are defined by the W3C (World Wide Web Consortium). However, both Microsoft and Netscape extended the HTML standard with their own special tags, thus creating two competing sets of HTML tags and behaviors. Web developers with limited resources then had to choose one of these tag sets, and as Internet Explorer usage grew they opted to use Internet Explorer's extensions, thereby making it ever more preferable over Netscape [33]. The dominance of Internet Explorer was so strong that Microsoft didn't bother to release a major version of Internet Explorer from 2001 until 2006, making do with a Service Pack for Internet Explorer 6 as part of a Windows XP Service Pack.

Up to this point browsers were proprietary software, even if distributed for free. But with the collapse of Netscape's market share, Netscape released its Netscape Communicator 5.0 source code for community involvement in the development via mozilla.org in March 1998 [2]. From this source code release the Mozilla Suite was created. However, the development continued to be influenced by Netscape Communications Corporation, which had been acquired by AOL. David Hyatt, Joe Hewitt, and Blake Ross were not pleased with this alliance of Mozilla with Netscape, which was hurting Mozilla independence and more importantly led to feature creep and bloat. So in mid 2001 they created an experimental branch of the Mozilla suite, which kept the user interface but reimplementing the backend from scratch [3]. This became the open source Firefox browser. And on August 9, 2003 Mozilla released a revised road map that reflected a shift from

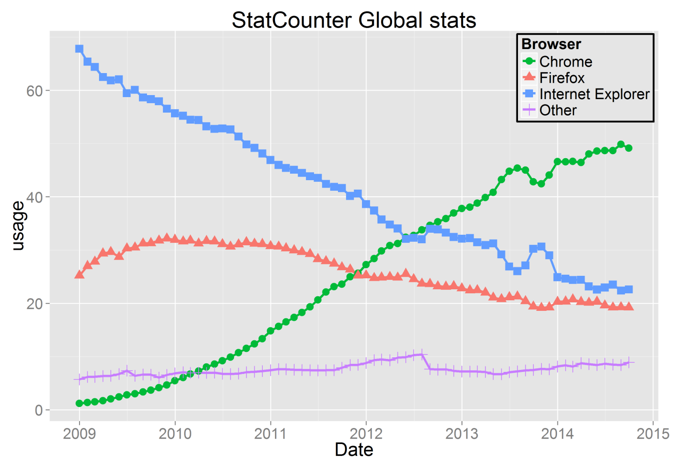


Figure 1: Browser's usage data from StatCounter.com.

the Mozilla suite to Firefox [4], which was finally released on November 9, 2004 [5]. later, in March 2011, Mozilla moved to rapid development with a 16-week cycle and then a 6 weeks cycle [6, 7].

Firefox's market share grew slowly, and by 2009 it managed to wrestle away up to 30% from Internet Explorer. But by this time a new contender had arrived. Google first released Chrome 1.0 on September 2, 2008. Concurrently, Google released most of the browser's source code as an open source project named Chromium thus establishing an open source community. The main reason was the belief that a strong community will help improve Chrome [8]. Additional reasons were to help drive the web forward as other open source projects like Firefox and WebKit did, and enabling people to create their own projects based on Chromium. As of today the development of Chrome is based on the development of stable releases of Chromium, and the two browsers are identical in most aspects [9]. However, it is important to distinguish Chrome from Chromium, as Chrome has several features that are absent from Chromium such as a non-free PDF viewer. Chrome and Chromium moved to a rapid development strategy in mid 2010 [10].

2.2 Browsers Usage Statistics

In the six years since its release Chrome has dethroned Internet Explorer, and Firefox's market share has also decreased, as shown in Figure 1. Data for such statistics is obtained as follows. Browser usage can be tracked using a counter embedded in the source code of web sites. This is implemented using a request to a counting service, enabling the counting service to extracts the browser information from the request and to use it to tabulate browser usage statistics. The data shown is from one of these services, StatCounter.com [11].

There are two main methods to interpret web browsers usage. The first method is to measure how many page loads came from each type of browser in a certain period of time. The second method counts how many unique clients (installations) were active in a certain period of time. Therefore, if a user visits 10 web pages, the first method will count this as 10 uses of the browser, while in the second method will count it as one user. Since the two methods measure different parameters their results may differ. The first method favors browsers that are used by heavy users, while the second method just counts the number of unique users without taking their activity into account. If we consider users who use the web extensively to be more important, this is a drawback. Moreover, identifying unique users is non-trivial and requires manipulating the raw data. We therefore use the raw counts data, and specifically the data for desktop browsers only not including mobiles and tablets. (The nick in the graphs at August 2012 represents the beginning of collecting data about tablets separately.)

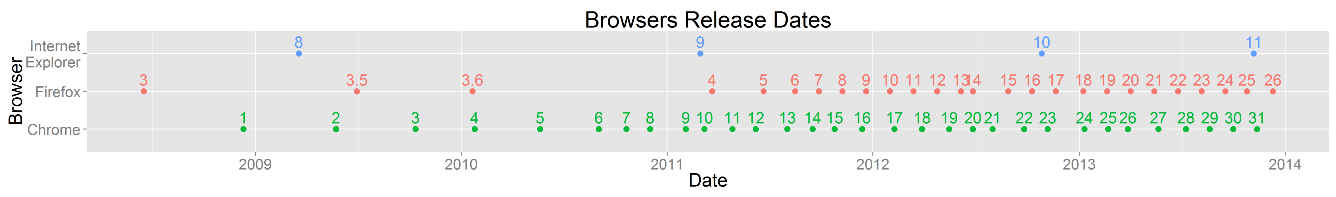


Figure 2: The release date of each version tested.

As shown in the graph, Chrome's market share has risen consistently over the years, largely at the expense of Internet Explorer. As of January 2015, Chrome is responsible for 51.7% of the page loads while Internet Explorer is responsible for 21.1%, Firefox for 18.7%, and other browsers for 8.4%.

Note that in fact any site can track the distribution of browser usage among the users who access this site. This may lead to different results if users of a certain type prefer a certain browser. For example, w3schools.com (a site devoted to tutorials about web development) also publishes data about browser usage. Their results for January 2015 are that 61.9% use Chrome, 23.4% use Firefox, and only 7.8% use Internet Explorer. This probably reflects a biased user population of web developers who tend to work on Linux platforms rather than on Windows. There is a danger that the StatCounter data is also biased, perhaps in the opposite direction, but even if so it probably reflects common usage by the public on popular web sites.

Regardless, it is obvious that Chrome now dominates the web browser market. The question is whether superior technology and smart feature selection contributed to the rise in popularity of Chrome. To answer this we tested the three major browsers which together account for 91% of the market share. (Thus we did not initially test Opera and Safari, whose market share is very low; Safari is also less relevant as it is tightly linked to Mac OS X, with a Windows version available only for several years and then discontinued.) The tests covered two distinct aspects. The first aspect is to test which browser has better technical performance, by evaluating their performance with a series of benchmarks. The second aspect is to test whether the product managers of those browsers made good decisions in choosing which features to develop and when.

3 Technical Performance

Timing is important to web page designers, because it affects the user experience [37]. But the precise definitions of performance metrics are complicated to pin down [32]. As a result quite a few different benchmarks have been proposed. We used these benchmarks to evaluate the technical performance of the different browsers, selecting a set which in our opinion cover a wide range of functionality. These benchmarks are divided to two categories. The first category is performance, and tests the performance of different aspects of the browsers such as JavaScript processing. The second category is conformance, and tests the conformance of the different browsers to common standards such as the HTML5 standard.

3.1 Methods

The experiments covered all Chrome versions from 1 to 31, all Firefox versions from 3 to 26, and all Internet Explorer versions from 8 to 11, meaning all the browsers versions in a five year span starting in mid 2008 until the end of 2013 (Figure 2). This is the period from the first release of Chrome until it achieved around 50% market share. They were conducted on two identical Core i5 4GB RAM computers, one running Windows 7 32 bit and the other running Windows 7 64 bit. The browser versions used on the 32 bit machine were Chrome 1–12, Firefox 3–5, and Internet Explorer 8–9, i.e. all the browsers released up to May 2011. The browsers versions used on the 64 bit machine were Chrome 13–31, Firefox 6–26, and Internet Explorer 10–11. The versions were divided between the machines since we encountered some compatibility issues

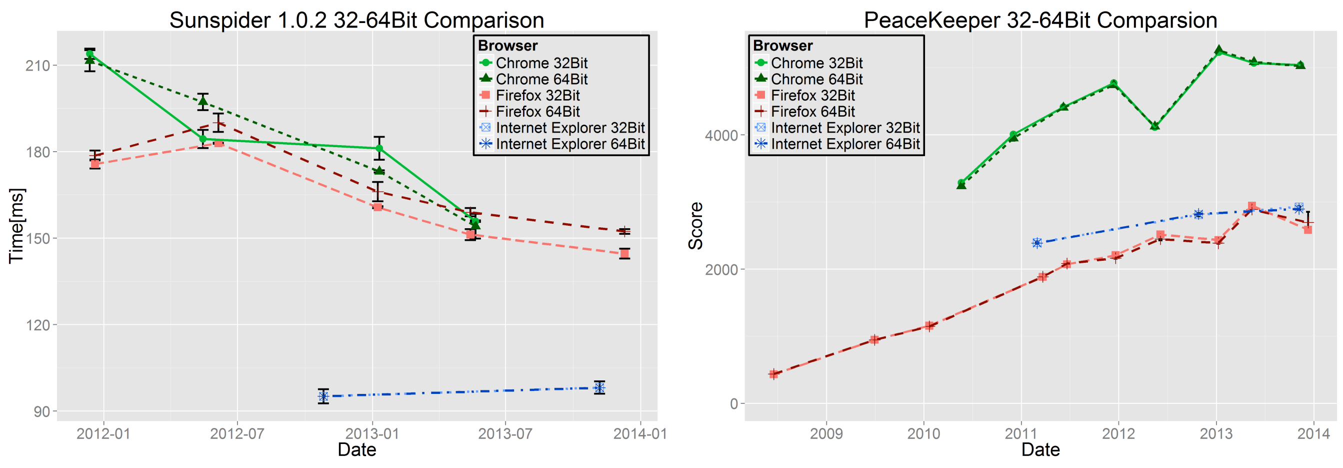


Figure 3: Comparing two benchmarks on 32 bit and 64 bit platforms.

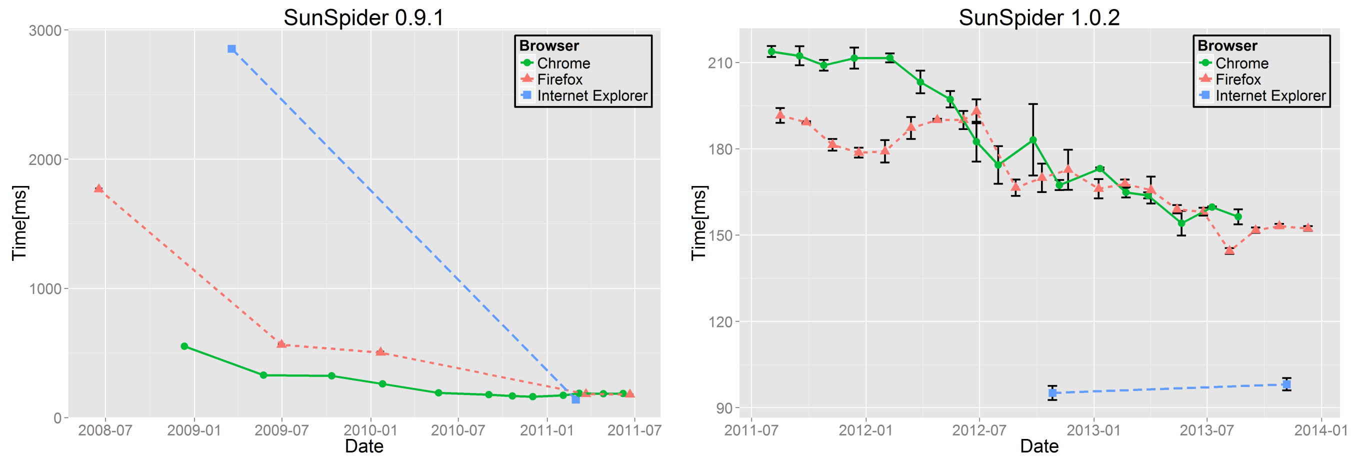
Type	Benchmark	Repetitions	Versions with no data		
			Internet Explorer	Firefox	Chrome
Performance	SunSpider 0.9.1	3	10,11	≥6	≥13
	SunSpider 1.0.2	3	8,9	≤5	≤12,30,31
	BrowserMark 2.0	3	8	3,3.5,3.6	1
	CanvasMark 2013	3	8	3	1,2,3
	PeaceKeeper	3			1,2,3,4
	HTML5 Benchmark	3	8	3	1,2,3
	Start-up Times	20			
Conformance	Browser Scope Security	N/A			
	CSS3 Test	N/A		3	
	HTML5 Compliance	N/A		4	2,7

Table 1: The benchmarks used in this study, how many iterations were conducted, and which browser versions could not be measured.

with earlier versions of Chrome on Windows 7 64 bit. Moreover, it makes sense to switch to 64 bit along the way since more and more machines are using Windows 64 bit.

To make sure that the experiments are consistent between 32 bit and 64 bit operating systems and eliminate operating system bias we have checked a third of the versions on both systems focusing on versions with a six months release gap. Examples for two of the benchmarks are shown in Figure 3. SunSpider 1.0.2 is an example of a relatively big difference between the results on the two platforms, and PeaceKeeper is an example of a small difference. In general we did not see any dramatic differences between the platforms. We therefore do not present any more results about such comparisons.

On all performance benchmarks we ran 3 repetitions of each experiment, except for the start-up times measurements where we ran 20 repetitions. Error bars are used to show the standard error. Not all the experiments ran properly with all the versions, especially with earlier versions. This is due to the fact that most of the benchmarks were designed and written later than some of the browser early versions, and used some features or technology that were not yet implemented in those early versions (Table 1).



(a) (b)

Figure 4: SunSpider 0.9.1 and 1.0.2 results. Note difference in scale for the two versions.

3.2 Performance Results

3.2.1 SunSpider

SunSpider is a well known benchmark developed by WebKit to measure pure JavaScript performance. WebKit designed this benchmark to focus on problems that developers solve with JavaScript [12]. However, the behavior on the benchmark may actually not mimic actual JavaScript work on real sites as noted by Richards et al. [34]. The benchmark measures the time to perform a set of tasks, so lower values are better. In the study we chose to use version 1.0.2, which was introduced by WebKit in order to make the tests more reliable [13, 14]. However, version 1.0.2 didn't work on old browser versions (Table 1). Therefore, we used version 0.9.1 on old browser versions [15], specifically those that were tested on the 32 bit machine.

Using SunSpider 0.9.1 we find that when Chrome was introduced it scored significantly better than Internet Explorer and Firefox. In the second version tested of Firefox (Firefox 3.5) the score was greatly improved but still lagged the parallel Chrome version. Although Internet Explorer 8 was released a couple of months after Chrome 1 it was five times slower. It took more than two years for Firefox and Internet Explorer to catch up with Chrome's parallel version (Figure 4a). In fact, Internet Explorer 9 not only caught up with Chrome but surpassed it. This has been attributed to its JavaScript optimization for dead code elimination, which some say was specifically done to boost SunSpider performance [16, 17].

In the SunSpider 1.0.2 tests Internet Explorer continued to show significantly better results compared to its rivals. Firefox and Chrome showed similar results most of the time (Figure 4b). For some reasons Chrome versions 30 and 31 had problems with this benchmark, but these were fixed in Chrome 32.

3.2.2 BrowserMark 2.0

BrowserMark 2.0 is a general benchmark developed by Rightware. Originally designed to test mobile and embedded devices, it is nevertheless commonly used to also test desktop browsers. The benchmark tests page loading, page resizing, conformance to HTML5, and network speed, as well as WebGL, Canvas, HTML5, and CSS3/3D. The calculated score combines all of these and higher scores are better.

The early versions of Internet Explorer and Firefox did not work with this benchmark. All of the browsers latest versions tested showed improvement compared to the first versions tested (Figure 5). Chrome in all of its versions was better than the equivalent rivals and showed a steady improvement over time. Inter-

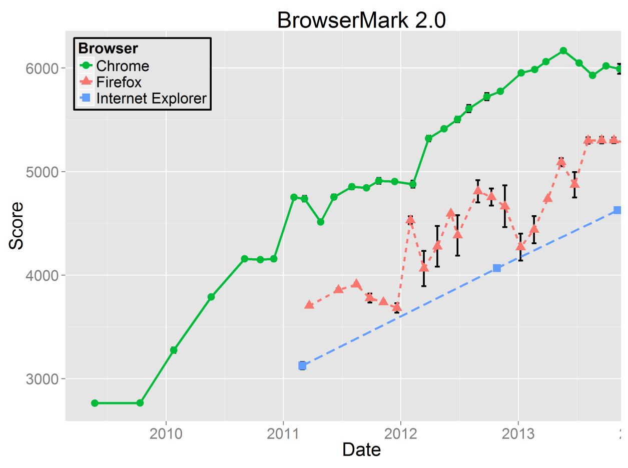


Figure 5: BrowserMark 2.0 results.

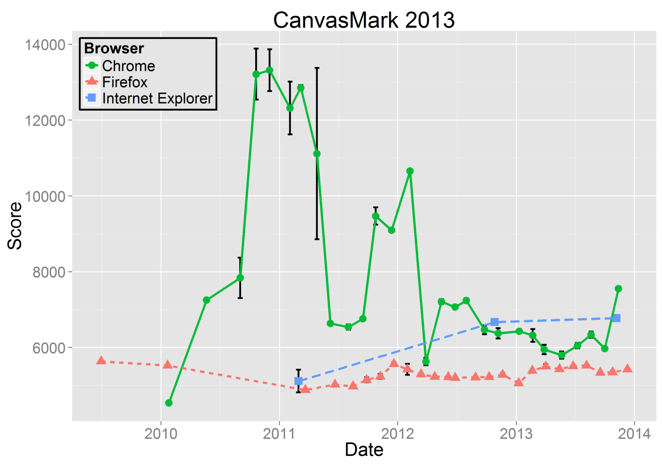


Figure 6: CanvasMark 2013 results.

net Explorer also showed an improvement over time but always came in last from all the browsers tested. Firefox performance was between Chrome and Internet Explorer. Interestingly, it showed an inconsistent improvement in benchmark score.

3.2.3 CanvasMark 2013

CanvasMark 2013 is a benchmark for HTML5 canvas 2D performance testing [18]. Higher scores are better. In this benchmark's documentation there was a note for Chrome users using Windows, encouraging them to change a setting in order to get better results due to a bug in the GPU VSync option for the Windows version of Chrome. However, we did not disable the setting since we want to test the versions as the average user would.

The results of running the benchmark show that Chrome exhibited inconsistent results over time (Figure 6). A great improvement was achieved from version 4 to 7. In contrast there was a sharp decline from version 10 to 12. Later, an improvement occurred from version 14 to 17, immediately followed by a sharp decline in version 18 of 50% of the score. But in spite of all these inconsistencies it was still better than Firefox and Internet Explorer most of the time. Internet Explorer showed an improvement in each version tested. On the other hand, Firefox does not show improvement over time.

3.2.4 PeaceKeeper

PeaceKeeper measures the browser's performance by testing its JavaScript functionality, as stated in FutureMark's website, the developer of PeaceKeeper [19]. However, it also tests other aspects of modern browsers such as WebGL, Canvases, DOM etc. Taking into account its many aspects we therefore consider it to be a general benchmark.

Chrome scored noticeably better results its rivals for this benchmark, throughout the period of time that we checked (Figure 7). However, note that PeaceKeeper did not run on earlier versions of Chrome (Table 1). Firefox and Internet Explorer scored similar results, both showing an improvement over time but still lagging Chrome.

3.2.5 HTML5 Benchmark

The HTML5 benchmark tests the smooth running of games rendered with the <canvas> element in HTML5. This is a specific feature of HTML5. Surprisingly the results obtained for this benchmark were very incon-

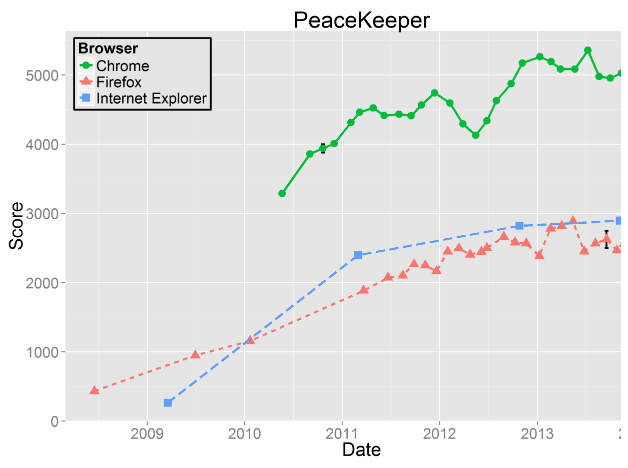


Figure 7: PeaceKeeper results.

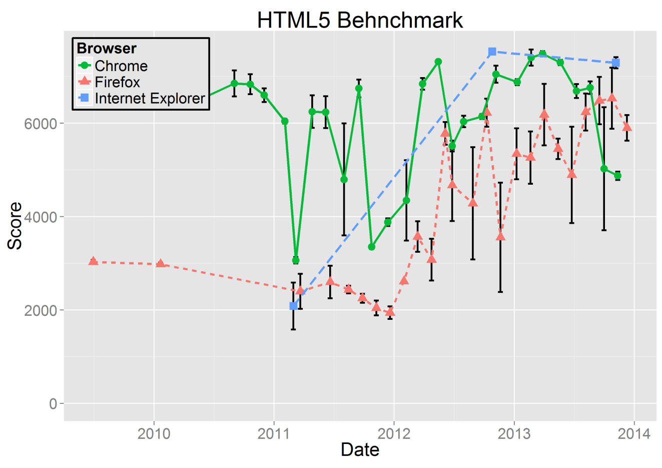


Figure 8: HTML5 benchmark results.

sistent, at least for Chrome and Firefox (Figure 8). Furthermore, even for repetitions performed on the same version the variability was sometimes very high, especially for Firefox. Nevertheless, it is possible to see a marked improvement in Internet Explorer performance from version 9 to 10, and also a generally improving trend for Firefox since version 10. In early versions Chrome seems to have been better than the others, but recently it is not and Explorer seems to be the best browser according to this benchmark.

3.2.6 Start-up Times

An important feature of all browsers, which may affect user satisfaction, is their startup times when they are launched. As we did not find a suitable benchmark that evaluates this we conducted an experiment to test the browser's cold start-up times. A cold start-up time is when the browser starts for the first time since the operating system was booted.

We tested the start-up times as follows. We wrote a script that runs during the operating system start-up. This script launches the browser one minute after the script starts running. The lag is meant to let the operating system finish loading. A time stamp is created just before launching the browser in order to mark the start time. The browser was set to open with our own crafted page. The script passed the time stamp to the crafted page via a URL parameter. The crafted page creates a second time stamp indicating the start of the page processing. The difference between the two time stamps was defined as the start-up time. The start-up times are then sent to a server for logging. Advantages of this procedure are, first, that it is independent of network conditions, and second, the test is similar to the user's real experience of launching the browser and loading a page.

The first versions of Chrome were the fastest to load (Figure 9). However, as Chrome's development advanced, it's start-up times crawled up. In Chrome version 7 the start-up times improved dramatically, but then continued to crawl up from version 13. In version 29 there was a spike in the start-up time, a 2.5 fold increase compared to the previous version, followed by a partial correction in versions 30 and 31. Surprisingly Firefox start-up times look steady with a slight decrease, notably in version 7. As a result, while it was the slowest by a wide margin in 2009, it became the fastest in 2013. Internet Explorer start-up time consistently increased over time, making it roughly twice as slow as the others in recent years (except for the spike in Chrome performance since version 29).

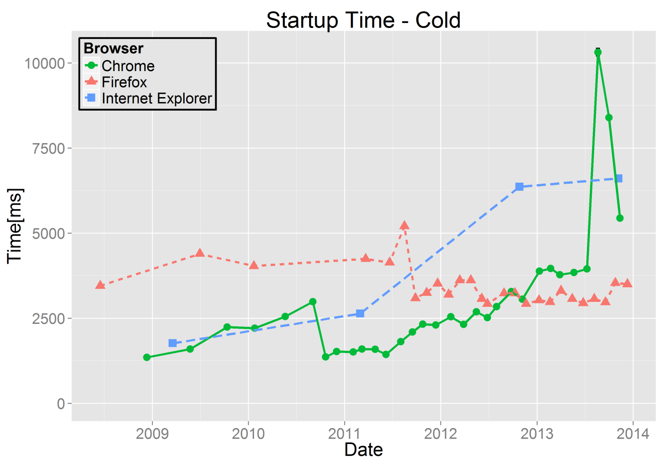


Figure 9: Cold start-up times.

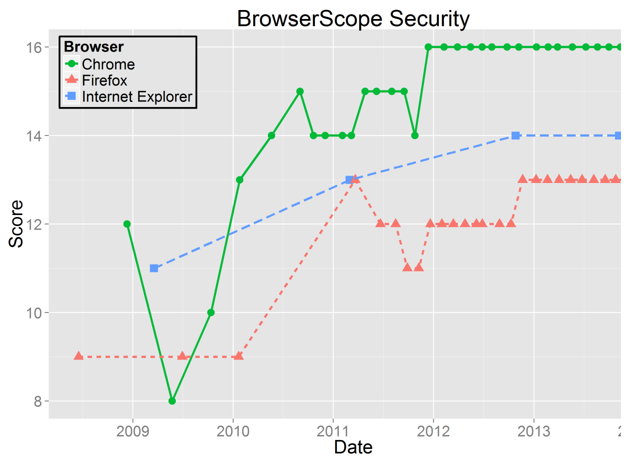


Figure 10: Browserscope Security results.

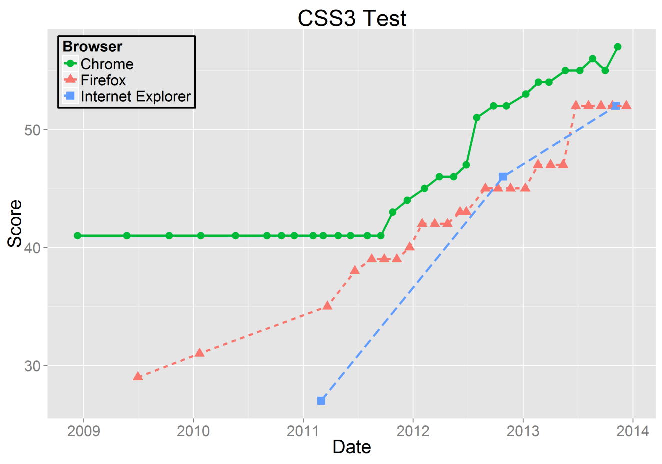


Figure 11: CSS3 Test results.

3.3 Conformance Results

3.3.1 Browserscope Security

Security is obviously an important feature for web browsers. Browserscope Security is a collection of tests meant to check "whether the browser supports JavaScript APIs that allow safe interactions between sites, and whether it follows industry best practices for blocking harmful interactions between sites" [20].

All three browsers exhibited a general (although not always monotonic) improvement in their benchmark results over time. The relative ranking according to this benchmark is very consistent between browser versions (Figure 10). Across the whole period Chrome had the highest score, Firefox had the lowest, and Internet Explorer was in between. The only exception is a large dip in score for Chrome versions 2 and 3, where version 2 was the worst of all parallel browser versions. This was surprising because of the overall consistency, and the fact that in the first version released Chrome had the highest score compared to its rivals.

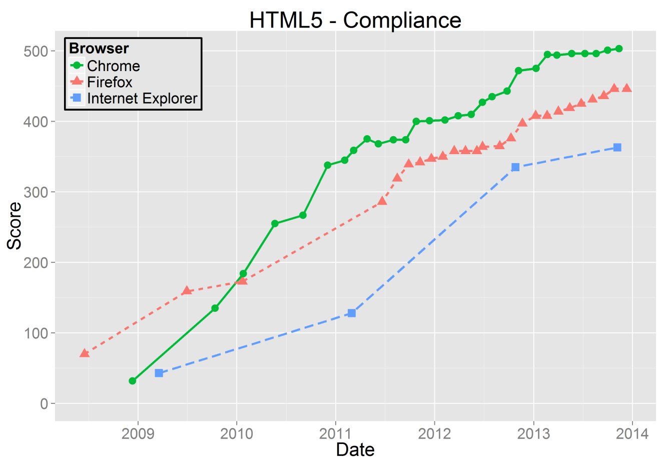


Figure 12: HTML5 compliance results.

3.3.2 CSS3 Test

CSS3 Test checks how many CSS3 elements in the W3C specification does a certain browser recognize [21]. This means CSS3 Test only checks the recognition itself but does not check the implementation or the quality of the implementation.

Interestingly Chrome's score does not change in the first three years, though it still managed to have a better score than its rivals. From version 15 Chrome consistently improved until the last version tested, remaining better than its rivals (Figure 11). Firefox showed several improvements in a stepwise manner. Internet Explorer had the lowest score in the first version tested (version 9) but improved its score dramatically in versions 10 and 11, achieving essentially the same level as Firefox.

3.3.3 HTML5 Compliance

The HTML5 Compliance benchmark consists of three parts. The main part is checking the conformance of the browser to the HTML5 official specification. The second part is checking specifications related to HTML5 such as WebGL. The third part is checking the specification for experimental features that are an extension to HTML5 [22].

The results for this benchmark show that all the browsers improve over time. Firefox had the best score until version 3.6, and after that Chrome version 4 and up had the best score (Figure 12). Internet Explorer always had the lowest score.

3.4 Summary of Benchmarks Results

Table 2 summarizes the results of all the benchmark and conformance tests. These indicate that Chrome generally has an advantage over its competitors.

4 Feature Selection

Another aspect in which the browsers differ from one another is their features. Not all features have the same importance, so it is advantageous for a browser to have the most meaningful features as early as possible. The feature portfolio of each browser evolves through time. In this section we tabulate which browsers released features early and which browsers lagged in releasing features. In addition we wanted to evaluate

Benchmark	Result
SunSpider	Chrome used to be better, now Internet Explorer is significantly better
BrowserMark 2.0	Chrome is better, Explorer worst
CanvasMark 2013	Chrome is relatively good but inconsistent, Firefox worst
PeaceKeeper	Chrome is significantly better
HTML5 Benchmark	Results are inconsistent, Explorer somewhat better
Start-up Times	Chrome was better but now Firefox is better, Explorer has deteriorated
Browserscope Security	Chrome is better, Firefox worst
CSS3 Test	Chrome is better
HTML5 Compliance	Chrome is better, Explorer worst

Table 2: Summary of benchmark results.

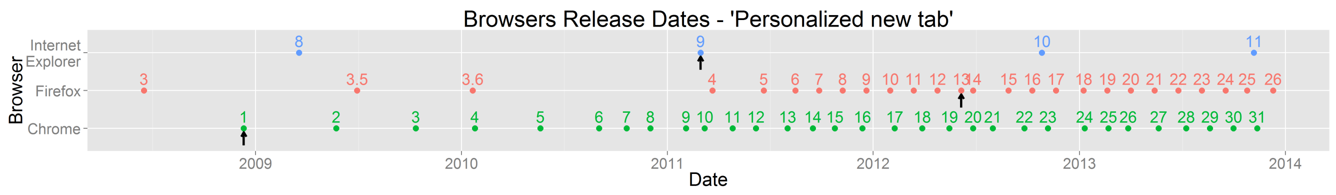


Figure 13: Feature release margin example: The "personalized new tab" feature was released in Chrome 1, Internet Explorer 9, and Firefox 13 (marked with arrows).

the importance of each feature. We used an online survey to assess the importance of each feature to the end user.

4.1 Methods

4.1.1 Feature Selection

We chose 43 features which in our opinion represent a modern browser. As Chrome 1.0 was our starting point, 11 features included in this version and also developed prior to this point by the competing browsers were excluded from consideration; for example, this included multiple tab browsing. Seven further features were excluded because they were released at about the same time by all three browsers, so they did not confer any competitive advantage (see below for details). Therefore, the study was conducted on 25 features (Table 3, Note that the features are listed in a random order).

4.1.2 Feature Release Margin

First we dated the release of each of the 25 selected features in each browser (Table 4). Then we developed a metric which states whether a certain browser released a feature ahead of its competitors by a meaningful margin and/or whether a certain browser lagged both of its competitors in the release by a meaningful margin. A browser was awarded a "win" if it released a feature ahead of all its competitors, and a penalty or "loss" was given if a browser lagged or did not released a certain feature. Note that each feature can have a maximum of one "winner" and a maximum of one "loser". If a feature had neither a "winner" nor a "loser" it was excluded from the study as no browser had a competitive advantage or disadvantage.

#	Feature	Explanation
1	Add-ons manager	Allows you to disable/remove previously installed add-ons
2	Download manager	Allows you to view/pause current downloads and view previously downloaded files
3	Auto-updater	Silently & automatically updates the browser if there's a new version
4	Caret navigation	Allows you to navigate through a site using the arrow keys (just like in any document processor e.g. in Microsoft Word)
5	Pinned sites	Allows you to have faster access to your favorite sites like Facebook or your Email provider
6	Sync	Allows you to sync you favorites/preferences/saved passwords etc. through computers and platforms
7	Session restore (automatically)	Upon a crash, the browser will restore the sites you were surfing before the crash
8	Crash/security protection	Allows you to continue browsing although a site/plugin crashed or hanged
9	Malware protection	Enables the browser to warn and block suspicious sites that are known to be harmful
10	Outdated plugin detection	Allows the browser to detect if a plugin has become incompatible/vulnerable with the browser's version
11	Do not track	Allows the browser to request sites not to track the browsing
12	Themes	Allows you to personalize the browser appearance by changing the skin
13	Experimental features	Allows you to try experimental features in the browsers that aren't turned on by default
14	Multiple users	Allows you to have multiple profiles (different bookmarks/saved passwords/history) on the same computer user
15	Apps	Allows you to install applications that will run in the browser (like games or other applications)
16	Developer tools	Allows you to examine a site's interaction with the browser
17	Personalized new tab	Allows you to see your most visited sites upon the launch of the browser (the first tab that is opened on launch of the browser)
18	Click-to-Play	Disables the automatic launch of a plugin's content. The user must explicitly click on the flash/applet in-order to load and play it
19	Print preview	Allows the user to view the page before printing it
20	Per-site security configuration	Allows you to control which sites will block popups/cookies/images/scripts/plug-ins etc.
21	Web translation	Allows the browser to translate a page automatically to a desired language
22	Spell Checking	Marks misspelled input you typed and corrects it
23	Built-in PDF viewer	Allows the browser to open PDF files without any 3rd party plugins
24	Sandboxing	A security concept that certain parts of the browser run individually with restricted privileges only
25	RSS reader	Allows the browser to know when a certain site, that supports RSS, was updated. News feeds etc.

Table 3: Selected features [arbitrary order].

#	Feature	Feature release version										Survey results				
		Explorer		Firefox		Chrome		1	2	3	4	5
1	Add-ons manager	pre 8		pre 3		4	**L**	10	24	39	100	81
2	Download manager	9	**L**	pre 3		1		7	24	48	84	91
3	Auto-updater	9		16	**L**	1	**W**	28	39	66	62	59
4	Caret navigation	8		pre 3		none	**L**	56	61	49	43	45
5	Pinned sites	9		5	**L**	2		45	36	47	60	66
6	Sync	10	**L**	4		4		56	33	47	62	56
7	Session restore (automatically)	10	**L**	pre 3		1		16	28	28	62	120
8	Crash/security protection	8		20	**L**	1		6	13	37	110	88
9	Malware protection	9	**L**	3		1		9	13	41	73	118
10	Outdated plugin detection	none	**L**	3	**W**	10		15	56	74	77	32
11	Do not track	9		4		23	**L**	21	28	43	85	77
12	Themes	none	**L**	pre 3	**W**	3		134	60	35	21	4
13	Experimental features	none	**L**	pre 3	**W**	8		72	76	51	43	12
14	Multiple users	none	**L**	pre 3	**W**	12		110	50	40	37	17
15	Apps	none	**L**	16		9	**W**	77	63	51	47	16
16	Developer tools	8		4	**L**	1		51	40	44	51	68
17	Personalized new tab	9		13	**L**	1	**W**	48	61	67	49	29
18	Click-to-Play	none	**L**	8		8	**W**	31	54	75	53	41
19	Print preview	pre 8		pre 3		9	**L**	19	21	58	92	64
20	Per-site security configuration	11	**L**	3	**W**	5		11	33	87	75	48
21	Web translation	none		none		5	**W**	50	66	59	55	24
22	Spell Checking	10	**L**	pre 3		1		20	35	44	78	77
23	Built-in PDF viewer	none	**L**	19		8	**W**	11	28	42	80	93
24	Sandboxing	pre 8		none	**L**	1		54	43	59	51	47
25	RSS reader	8		pre 3		none	**L**	119	65	34	22	14

Table 4: Feature release dates and survey results. **W** and **L** denote wins and losses, respectively.

"A meaningful margin" was defined as more than one release cycle, that is, when it took the competitors more than one version to include the feature after it was initially introduced. For example, "Personalized new tab" was introduced in Chrome 1. At the time the most recent versions of Internet Explorer and Firefox were 7 and 3, respectively. The feature was released in Internet Explorer 9 and Firefox 13, meaning that this was a meaningful margin. Had this feature been released in Internet Explorer 8 or Firefox 3.5 it would not have counted as a meaningful margin, despite being later than Chrome 1. Furthermore, Firefox lagged Internet Explorer in the release of the feature in a meaningful margin (Figure 10). So in this case Chrome received a "win" and Firefox received a "loss". All the release versions and their identification as wins or losses are shown in Table 4.

Note that the definition of the release margin is based on releases of new versions, and not on absolute time. This gives an advantage to browsers that are released infrequently. For example, any innovations included in Chrome versions 2 to 9 — a span of nearly two years — and included in Internet Explorer 9 would not be considered to have a significant margin, because Microsoft did not release any versions during all this time. Consequently our results may be considered to be conservative.

4.1.3 Feature Importance Survey

We created an online survey that lists and explains the 25 selected features, and asks the participants to evaluate the importance of each feature relative to other listed features on a discrete scale of 1 (least important) through 5 (most important). The features were listed in the same random order as in Table 3. The intended audience were people who spend many hours a day on the world wide web. The survey was published on Reddit (sub-reddit /r/SampleSize) [23] and on CS Facebook groups of the Hebrew University and Tel-Aviv University in Israel. 254 people answered the survey, and the distribution of results is shown in Table 4. The statistical analysis was performed on all of the participants.

4.1.4 Statistical Analysis

Opinion surveys like the one we conducted are commonly analyzed by calculating the average score received by each entry, and considering these to be the averages of samples of different sizes and unknown variance. Then a test such as Welch's t-test is used to check whether or not they are significantly different. However, this approach implicitly assumes that the scale is quantitative, meaning that the difference between 1 and 2 is the same as between 2 and 3, 3 and 4, and 4 and 5. But given that these numbers represent subjective levels of importance, this is not necessarily the case. Moreover, different people may use the scale differently. Therefore both the averaging and the statistical test are compromized.

Another problem with human users is that some of them are hard to please, and always use only the bottom part of the scale, while others are easy to please, and use the top part of the scale. But checking our data we found that most respondents actually used the full scale from 1 to 5, with an average near 3. This implies that we do not need to perform adjustments to the data to compensate for such different behaviors [36].

Nevertheless, comparing average scores is still not justifiable. We therefore use an analysis method due to Gilula and others where brand A is judged to be superior to brand B if the distribution of opinions about it dominates the distribution of opinions about B in the stochastic order sense. Mathematically this is expressed as $\forall s : F_A(s) \leq F_B(s)$, where F_A and F_B are the cumulative distribution functions of the opinions regarding A and B, respectively. Graphically, the plot of F_A is lower and to the right of the plot of F_B, and it accumulates more slowly. In simple terms this means that for each level of opinion 1 to 5 the probability that A receives a score of at least this level is higher than the probability that B receives such a score. However, in many cases the distributions do not dominate each other (and their graphs cross each other). It is then necessary to adjust the data by grouping brands and/or score levels together until dominance is achieved [28, 30, 29, 35, 31].

In our case the brands are the features of the browsers. But we don't really care about ranking the individual features. Rather, we want to rank sets of features. For example, we can take the set of features that were Chrome "wins", and compare it to the set of features that were Chrome "losses". If the first set turns out to be more important to users, then this testifies that Chrome project managers chose wisely and invested their resources in prioritizing the more important features first.

To perform these calculations we used the Insight for R v0.4 software package which implements this approach[1]. Given the adjusted (collapsed) data, we also calculate the polarity index. This is the ratio of users who considered features important (levels 4 and 5) to the rest (levels 1 to 3). A polarity index less than 1 indicates that the balance is skewed towards not important, while polarity index higher than 1 indicates that user opinion is skewed towards most important. Unlike average scores, the polarity index has a direct quantitative meaning and therefore the indexes of different brands can be compared to each other.

[1]The software and statistics advice were kindly provided by professor Zvi Gilula.

Rank	Browser	No. of wins	Importance score dist.					Polarity index
			1	2	3	4	5	
1	Chrome	6	0.16	0.20	0.24	0.23	0.17	0.67
2	Firefox	5	0.27	0.22	0.23	0.20	0.09	0.40
3	Internet Explorer	0	N/A	N/A	N/A	N/A	N/A	N/A

Table 5: Comparison between Chrome and Firefox "wins".

Rank	Browser	No. of losses	Importance score dist.				Polarity index*
			1	2	3-4	5	
1	Firefox Internet Explorer	6 13	0.17	0.16	0.45	0.22	2.03
2	Chrome	5	0.18	0.16	0.44	0.22	1.94

Table 6: Comparison between Chrome and Firefox/Internet Explorer "losses".

4.2 Results

In order to analyze which browser released the most important features earlier than the competitors we identified the "wins" and "losses" of each browser, and compared them in different combinations. Thus we performed as analysis of the "wins" of different browsers, and analysis of their "losses", and a specific analysis of the "wins" versus the "losses" of Chrome. Our results show that Chrome received a "win" in 6 features and Firefox in 5 features. In contrast, Internet Explorer did not receive any "wins", and 15 features did not have a "winner". Chrome received a "loss" in 5 features, Firefox in 6 features, and Internet Explorer in 13 features. Here only one feature was not ascribed as a "loss" to any of the browsers (the "web translation" feature).

These results already show that Chrome tended to release new features ahead of the other browsers, and that Internet Explorer lagged far behind. However, they still do not indicate whether the features released early by Chrome were indeed the more important ones. The following paragraphs provide this analysis by comparing the distributions of importance scores given to the sets of features that were "wins" and "losses".

4.2.1 Wins

The results of comparing the the user opinions regarding the feature sets where each browser "won" is shown in Table 5. A stochastic order of the response levels was present without any adjustments, with Chrome ranked first and Firefox second. Since Internet Explorer did not have any "wins" it was ranked last. The Polarity Index of Chrome and Firefox were 0.67 and 0.40, respectively. While both are smaller than 1, the features in which Chrome received a "win" were still more important to the end user, since the Polarity index was higher. The direct quantitative meaning is that for Chrome users considered the "winning" features to be important $\frac{2}{5}$ of the time, whereas for Firefox they considered them to be important only about $\frac{2}{7}$ of the time.

4.2.2 Losses

Given limited resources the developers of a browser cannot do everything at once, so the implementation of select features must be delayed. Under such circumstances it is best to delay those features that will not be missed by many users, namely those that are considered less important. Therefore, a lower ranking and a lower polarity index are favorable when comparing feature sets which are "losses".

In order to achieve dominance the ranking algorithm united opinion levels 3 and 4 (Table 6). Even so

Rank	Class	No. of features	Importance score dist.				Polarity index
			1-2	3	4	5	
1	Losses	5	0.33	0.18	0.27	0.22	0.96
2	Wins	6	0.36	0.24	0.23	0.17	0.67

Table 7: Comparison between Chrome "wins" and "losses".

Firefox and Internet Explorer showed the same trends and could not be distinguished one from the other, so they were clustered together. The result after these adjustments was that Firefox and Internet Explorer were ranked on top and Chrome was ranked lower. This means that the features in which Firefox and Internet Explorer received a "loss" were more important to the end users. However, it should be noted that the differences in the distributions were marginal at best. The Polarity Index could not be calculated in the regular way due to the unification of levels 3 and 4. The results given in the table are therefore the ratio of levels 3 to 5 to levels 1 and 2, making them higher than in other comparisons. They are close to each other, but still Chrome is a bit lower.

4.2.3 Chrome Wins and Losses

Finally, we compared the features that Chrome "won" with those that it "lost". In order to achieve a stochastic order the algorithm clustered levels 1 and 2 together. Interestingly the "losses" won, meaning that they were considered more important (Table 7). The Polarity Index of the "wins" and the "losses" were 0.67 and 0.96, respectively, meaning the features which Chrome released ahead of its rivals were considered important to the users about 40% of the time, whereas those in which it lagged behind were considered important nearly 50% of the time. Thus the prioritization used in developing Chrome was better than that of its rivals (as shown in the two previous analyses), but it was not perfect.

5 Discussion

5.1 Summary of Results

We tested the performance of three browsers, Chrome, Firefox, and Internet Explorer, using a wide array of commonly used benchmarks and across a long period of time. The results, summarized in Table 2, show that Chrome had an advantage over its rivals in most cases. But note that we did not test all aspects of browser technology, and relied on benchmarks. This is a threat to validity. For example, it is possible to conduct a more detailed study of compatibility issues [27] to try and quantify the problems that may occur with each browser.

In addition we compared the release dates of 25 specific features. Then features had a "winner", meaning that they were released by one browser ahead of the others by a meaningful margin. All but one also had a "loser", that is a browser that lagged behind by a significant margin. The relatively low fraction of features that had a "winner" (and the fact that 7 features were excluded from the study because they did not have a "winner" nor a "loser") indicates that the development of each browser is not isolated from its rivals. As a result, some features are released at about the same time. On the other hand, some browsers still managed to release a fair number of innovative features: Chrome and Firefox received 6 and 5 "wins", respectively. Internet Explorer on the other hand did not receive any "wins" and had the most "losses", 13. Chrome and Firefox had 5 and 6 "losses", respectively.

Chrome achieved better results in five technical performance tests: BrowserMark 2.0, PeaceKeeper, Browserscope Security, CSS3 Tests, and HTML5 Compliance. Firefox achieved better results only in the start-up times test. Interestingly, Chrome start-up times results may indicate that Chrome suffers from a

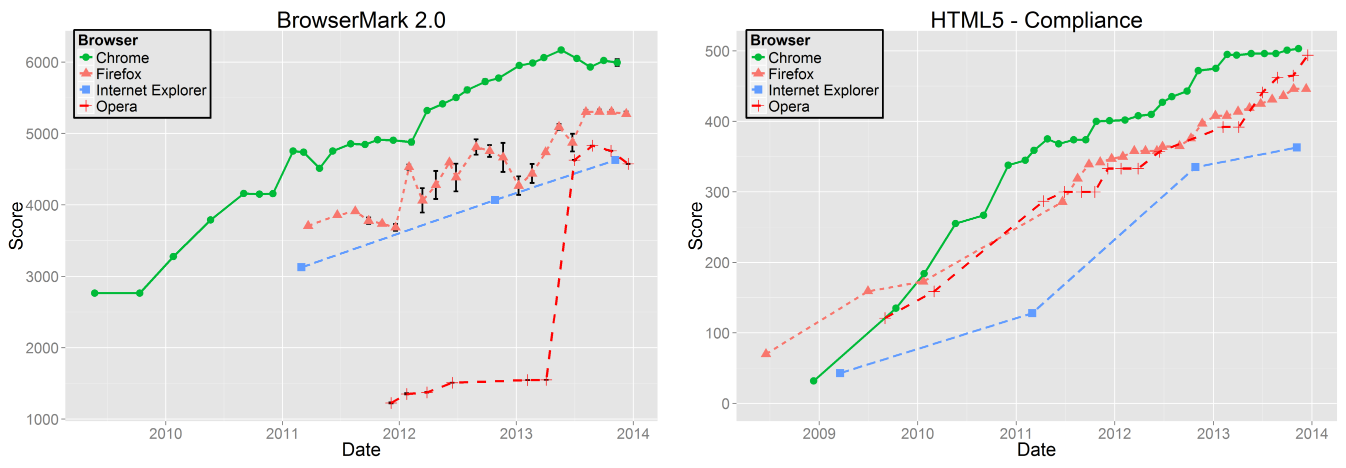


Figure 14: Sample Opera results overlaid on the previous results.

feature creep impacting its start-up times. Internet Explorer achieved better results only in SunSpider. The HTML5 Benchmark was very inconsistent in its results, and furthermore, variations between repetitions were surprisingly high (Figure 8).

Although Chrome and Firefox received similar numbers of "wins" the feature importance survey showed that features in which Chrome "won" were more important to the users than features in which Firefox "won". Likewise, features in which Chrome "lost" were less important to users than the features in which Firefox and Internet Explorer had "lost", but in the case of losses the difference was marginal. Furthermore, Chrome "losses" were actually more important to users than its "wins".

Ideally a browser should release the most important features to users first, and in case it has to lag in the release of certain features they should be of less importance to users. The results indicate that Chrome project managers were somewhat better at this than the project managers of competing browsers. This means that they generally made better choices than their rivals. However, they did not manage to focus on only the important features, and when they lagged in feature release, these features were sometimes actually more important to users.

5.2 The Importance of Marketing

A drawback of the work reported so far is its focus on purely technical aspects of browsers. We did not check the marketing aspect of the browsers, hence, we cannot separate the technical superiority from the brand name. For example, according to [24] an important aspect of Chrome's rise was "the great promotional efforts produced by Google" in the shape of promotional videos released on the web. Examples of 7 such videos are given, including the "Chrome speed tests" video released in May 2010 that went viral; at this time Chrome was just beginning its rise in market share, and the video may have contributed to its momentum. All 7 were released by April 2012, when Chrome had already overtaken Firefox but was still second to Internet Explorer.

In addition, other smaller browsers like Opera and Safari were not tested originally. Therefore, we cannot say if Chrome is technologically superior or inferior to them. If either of the untested browsers are technologically superior to Chrome than we cannot attribute market share to technological superiority, and must place a greater emphasis on marketing.

To lay this concern to rest we conducted a study of the performance of Opera using the same benchmarks as in Subsection 3.2. We focused on Opera and not on Safari for two reasons. First, Opera has a reputation for being an innovative and technologically advanced browser. Second, Safari is specifically targeted for

Apple platforms, and therefore is not really part of the same desktop market as the other browsers we are studying.

Not all versions of Opera were tested, as many of the benchmarks did not run properly on early versions. The results were that Opera performance was generally inferior to that of Chrome (two examples are shown in Figure 14). In some benchmarks, notably BrowserMark and Browserscope Security, its scores were actually lower than for all other browsers for many years. The sharp improvement in BrowserMark shown in Figure 14 is probably due to the move to using WebKit (and thus the same rendering engine as Chrome) in version 15 [25]; similar improvements were also seen in some other benchmarks in this version. In other benchmarks, such as CSS3 Test and HTML5 Compliance, Opera's results were similar to those of Firefox throughout. The only benchmark in which Opera was the best browser for a considerable period was CanvasMark, but this period only started in 2012 (and performance dropped in version 15).

Given the results for Opera we can say that Chrome was generally superior to all competing browsers from the technical perspective. In addition it may well have been superior from the marketing aspect. In this case the rise of Chrome was most probably the result of a win-win situation where technology considerations and marketing appeal both pointed in the same direction.

5.3 Implications Regarding Software Development

In the late 1990s the browser war between Internet Explorer and Mozilla (later Firefox) was portrayed in colors of a race between proprietary software and open source software. Chrome is a unique combination of both. It was initially developed within Google, but then it was largely turned into an open source project. An open question is whether it was really turned over to the open source community, or remains largely under Google control, both in terms of code contributions and in terms of management. Thus An interesting direction for further work is to dissect the sources of advances made in Chrome (or rather Chromium), and to see how many of them can be attributed to developers outside Google.

Be that as it may, our results regarding Firefox and Internet Explorer already provide evidence for the potential superiority of large-scale open-source projects. Up to 2009 Firefox was quickly gaining market share at the expense of Internet Explorer, and our benchmark results indicate that it appears to have had superior performance for most of them (this conclusion is restricted, however, by the fact that we did not measure Internet Explorer 6 and 7 which were the current versions during the early Firefox years). It also appears to have been more innovative, as reflected by having some "wins" in early introduction of new features, and much less "losses" than Internet Explorer. This is an important result, as it demonstrates that a large open-source project can in fact prioritize features better than a competing product developed in-house by a leading software firm.

Another sometimes contentious aspect of software development is the use of agile methodologies with a rapid release cycle as opposed to heavier plan-based methodologies with large-scale infrequent releases. Tabulating the browser version release dates indicates that Chrome and Firefox transitioned to rapid development methods, releasing a new version every 4-8 weeks (Figure 2). This meant that there were more releases and each release contained fewer new features, leading to more focus in the work on each new release. At the same time, with rapid releases the development teams could respond more quickly to their competitors' released features which they considered to be important, and also respond quickly to user feedback and requests.

6 Conclusions

We tested the technical performance of the three major browsers and compared the release times of 25 features. Overall it seems that the browsers became better over time, as most of the tests that were examined showed a clear improvement trend, and all the browsers evolved and received better results. It is also apparent that the release rate of versions became more frequent over the years.

In conclusion, the cumulative evidence we have collected indicates that Chrome's rise to dominance is indeed consistent with technical superiority over its rivals and with insightful management of feature selection. However, we still cannot say that it is *the result* of technical superiority alone, as marketing and the Google brand probably also played an important role. Studying the marketing campaign may well be a worthwhile effort.

Acknowledgments

Many thanks to Zvi Gilula for his explanations about statistical procedures and for providing the software used in the analysis. This work is a greatly extended version of a class project done with Kobi Atiya, who contributed to the conceptual design and initial results. The Opera measurements were performed by Amir Massarwi.

References

[1] URL: http://www.w3.org/People/Berners-Lee/WorldWideWeb.html.

[2] Chached version of http://www.netscape.com/newsref/pr/newsrelease591.html. URL: http://xml.coverpages.org/netscapeCode980401.html.

[3] URL: https://web.archive.org/web/20110623034401/http://weblogs.mozillazine.org/ben/archives/009698.html.

[4] URL: http://www-archive.mozilla.org/roadmap/roadmap-02-Apr-2003.html.

[5] URL: http://website-archive.mozilla.org/www.mozilla.org/firefox_releasenotes/en-US/firefox/releases/1.0.html.

[6] URL: http://mozilla.github.io/process-releases/draft/development_overview/.

[7] URL: http://blog.mozilla.org/channels/2011/07/18/every-six-weeks/.

[8] URL: http://blog.chromium.org/2008/09/welcome-to-chromium_02.html.

[9] URL: https://code.google.com/p/chromium/wiki/ChromiumBrowserVsGoogleChrome.

[10] URL: http://blog.chromium.org/2010/07/release-early-release-often.html.

[11] URL: http://gs.statcounter.com/.

[12] URL: https://www.webkit.org/perf/sunspider/versions.html.

[13] URL: https://www.webkit.org/blog/2364/announcing-sunspider-1-0/.

[14] URL: https://www.webkit.org/perf/sunspider-1.0.2/sunspider-1.0.2/driver.html.

[15] URL: https://www.webkit.org/perf/sunspider-0.9.1/sunspider-0.9.1/driver.html.

[16] URL: http://blogs.msdn.com/b/ie/archive/2010/11/17/html5-and-real-world-site-performance-seventh-ie9-platform-preview-available-for-developers.aspx.

[17] URL: http://digitizor.com/2010/11/17/internet-explorer-9-caught-cheating-in-sunspider-benchmark/.

[18] URL: http://www.kevs3d.co.uk/dev/canvasmark/.

[19] URL: http://peacekeeper.futuremark.com/faq.action.

[20] URL: http://www.browserscope.org/security/about.

[21] URL: http://css3test.com/.

[22] URL: http://html5test.com/about.html.

[23] URL: http://www.reddit.com/r/SampleSize/.

[24] URL: http://www.makeuseof.com/tag/7-awesome-google-chrome-promo-videos/.

[25] URL: https://dev.opera.com/blog/300-million-users-and-move-to-webkit/.

[26] Hal Berghel. “Who Won the Mosaic War?” In: *Comm. ACM* 41.10 (1998), pp. 13–16.

[27] Shauvik Roy Chaudhary, Mukul R. Prasad, and Alessandro Orso. “X-PERT: Accurate Identification of Cross-Browser Issues in Web Applications”. In: *Intl. Conf. Software Engineering*. 35. 2013, pp. 702–711. DOI: 10.1109/ICSE.2013.6606616.

[28] Zvi Gilula. “Grouping and Association in Contingency Tables: An Exploratory Canonical Correlation Approach”. In: *J. Am. Stat. Assoc.* 81.395 (1986), pp. 773–779. DOI: 10.2307/2289009.

[29] Zvi Gilula and Abba M. Krieger. “Collapsed Two-Way Contingency Tables and the Chi-Square Reduction Principle”. In: *J. R. Stat. Soc. B (Methodological)* 51.3 (1989), pp. 425–433.

[30] Zvi Gilula, Abba M. Krieger, and Yaakov Ritov. “Ordinal Association in Contingency Tables: Some Interpretive Aspects”. In: *J. Am. Stat. Assoc.* 83.402 (1988), pp. 540–545.

[31] Zvi Gilula and Benjamin Yakir. “An Insightful Comparison of Brands on an Arbitrary Ordinal Scale”. In: *J. Market Research Soc.* 40.3 (1998), pp. 249–264.

[32] Patrick Meenan. “How Fast is Your Website?” In: *Comm. ACM* 56.4 (2013), pp. 49–55. DOI: 10.1145/2436256.2436270.

[33] Barry Phillips. “Designers: The browser war casualties”. In: *Computer* 31.10 (1998), pp. 14–16. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=722269.

[34] Gregor Richards et al. “Automated construction of JavaScript benchmarks”. In: *ACM SIGPLAN Notices* 46.10 (2011), pp. 677–694. URL: http://dl.acm.org/citation.cfm?id=2048119.

[35] Ya’acov Ritov and Zvi Gilula. “Analysis of contingency tables by correspondence models subject to order constraints”. In: *J. Am. Stat. Assoc.* 88.424 (1993), pp. 1380–1387. DOI: 10.2307/2291280. URL: http://www.tandfonline.com/doi/abs/10.1080/01621459.1993.10476421.

[36] Peter E. Rossi, Zvi Gilula, and Greg M. Allenby. “Overcoming Scale Usage Heterogeneity: A Bayesian Hierarchical Approach”. In: *J. Am. Stat. Assoc.* 96.453 (2001), pp. 20–31. DOI: 10.1198/016214501750332668. URL: http://www.tandfonline.com/doi/abs/10.1198/016214501750332668.

[37] Steve Sounders. “High-Performance Web Sites”. In: *Comm. ACM* 51.12 (2008), pp. 36–41. DOI: 10.1145/1409360.1409374.

[38] Ronald J. Vetter, Chris Spell, and Charles Ward. “Mosaic and the world wide web”. In: *Computer* 27.10 (1994), pp. 49–57. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=318591.

[39] Paul Windrum. “Leveraging technological externalities in complex technologies: Microsoft’s exploitation of standards in the browser wars”. In: *Research Policy* 33.3 (2004), pp. 385–394. URL: `http://www.sciencedirect.com/science/article/pii/S0048733303001434`.

[40] David B Yoffie and Michael A Cusumano. “Judo strategy. The competitive dynamics of Internet time.” In: *Harvard Business Review* 77.1 (1998), pp. 70–81. URL: `http://europepmc.org/abstract/MED/10345393`.