Front-Page Performance

Image compression & optimization

Richard Fussenegger
Dept. of Multimedia Technology
Salzburg University of Applied Sciences
Puch bei Hallein, Austria
rfussenegger.mmt-m2012@fh-salzburg.ac.at

***Abstract*—Images make up the majority of data that has to be transferred over the internet to render a website. In this paper I describe how it is possible to automatically compress and optimize images within a web application without significant quality loss. High compression rates and intelligent optimization help to save Bytes and accelerate the delivery of web pages. To evaluate the savings that are possible with today's technics a program was created which generates a test-set from the world's top websites, optimizes all images and creates a summary of the savings. The generated statistic is very important to illustrate how easy it would be to optimize the images and even integrate the process into an existing web application.**

***Index Terms*—WWW, performance, image compression, image optimization, VP8, WebP, JPG, PNG, GIF**

I. Introduction

More than 60 % of the Bytes that are transferred over the Internet for rendering a web page are related to images (Souders, Interesting Stats 2013). Moreover the HTTP Archive's statistics show, that the total size of images has grown more than 30 % in only one year (Souders, Compare Stats 2013). These observations support the assumption that the total size of images will increase more over the next years.

This might be related to the fact that companies like Apple with their Retina display technology have increased the need for extremely detailed pictures with high resolutions and designers who want to meet these requirements. But the reason isn't really important; the fact that they are increasing in size all over the web calls out for optimization techniques in order to keep up with fast delivery of many images to many clients.

New problems arise with the ongoing growth of mobile users. They have to wait much longer for big files to download because of slower networks and often can't even afford them because of metered connections. But even on fast connections big images tend to slowdown the rendering process, simply because there are often many pictures that should be displayed.

Another difficulty related to automated optimization and compression is that the application itself during its runtime only has little knowledge about the domain, context or content of the uploaded image. Is it a photo, icon or maybe text? While some websites know exactly what they are dealing with (e.g. Flickr mainly handles photos) others don't (e.g. Wikipedia). Knowing what should or could be optimized and compressed helps to implement specific software for exactly that use case. If implementing a more general approach the only thing software can rely on is the files extension (type). JPEG[1] for photos and Portable Network Graphics (PNG) or Graphics Interchange Format (GIF) for computer generated images. This assumption isn't possible with Google's new image format WebP. It was designed to replace all other formats, thus making it impossible to find out what it might represent. Only solution to this problem is that user's uploading images have to specify what kind of image they are uploading (e.g. the Google's WebP executable *cwebp* has some predefined presets which are perfect: photo, picture, drawing, icon and text).

II. Compression & Optimization

There are various compression techniques and all of them depend on the images format. On the web the available formats are limited to GIF, JPEG and

[1] JPEG is the actual name and not an abbreviation. It derives from "Joint Photographic Experts Group" who developed the standard.

PNG. The aforementioned new format WebP from Google isn't supported by all clients yet but promising. One of the biggest problems if it comes to image compression is patents. Compression algorithms are almost always occupied with them, using any—even if the sources are available—are there for almost impossible because one has to face a costly trail.

A. Graphics Interchange Format (GIF)

The GIF format is the oldest format used on the web. There were a lot of discussions related to the Lempel–Ziv–Welch (LZW) lossless compression algorithm (Lempel, Ziv and Welch 1984) that was utilized by this format. Not because the algorithm was inefficient, the patents were the problem; they ran out back in 2006 (Parbel 2006). Because of that many organizations worked on developing an alternative, which was presented in 1996 with the Portable Network Graphics format. PNG has more colors and a better compression; this is why the best optimization technique for GIF images is simple: *convert to PNG* (World Wide Web Consortium 2006)

Animated GIF images aren't that easily optimized, simply because the official PNG standard doesn't include animated images and the Animated Portable Network Graphics (APNG) format that was developed by Mozilla developers back in 2004 was rejected by the PNG development group back in 2007 (Roelofs 2007) in favor of keeping their own Multiple-Image Network Graphics (MNG) format which ensures that animated and still images are strictly separated. The APNG format has little browser support and the MNG format has more or less none. The best optimization technique for animated images is the usage of so called "*CSS sprites*" together with JavaScript or CSS3 animations. All of these techniques are lacking proper benchmarks and are part of ongoing discussions (Manian, et al. 2013).

B. Portable Network Graphics (PNG)

As already mentioned before, PNG was developed to replace GIF as the primary format on the web. The main goal was to create a lossless image format with full True Color and alpha support (RGBA color space with 4×2^8 Bit). A combination of the Lempel–Ziv–Storer–Szymanski (LZSS) lossless compression algorithm (Storer and Szymanski 1982) and Huffman coding (Huffman 1952)—better known as zlib (gzip, deflate)—is used to compress the resulting data. Before compressing one of five filtering methods is applied. Each method predicts the value of each byte of the image based on the corresponding byte of the pixel.

Most programs aren't saving PNGs with the best selection of pre-compression filters and sometimes don't apply any compression. Using software to re-compress the file and save it with optimized usage solves this issue. Other techniques involve the reduction of used colors plus dithering if appropriate. Usage of alpha transparency is something that can bloat PNG files, it should only be used if really necessary.

Creating drop-in replacements for LZSS and/or the Huffman coding could improve the compression further, but nobody ever tried that. Most interesting new algorithm—again from Google and the WebP project—is Zopfli. A true drop-in for zlib (gzip, deflate) (Alakuijala and Vandevenne 2013).

C. JPEG

The presently most used format on the web isn't GIF anymore as it was in the past, it is JPEG. Although this isn't quite accurate, as JPEG is only a standard which describes how to compress image data, but not how that compressed that should be stored. Almost all JPEG images on the web are JPEG File Interchange Format (JFIF) files, which was introduced back in 1991 by Eric Hamilton.

This is an important distinction at this point. While GIF uses LZW and PNG deflate to compress their data, JPEG itself is the compression algorithm (although it also uses Huffman coding at some point during the compression, this is no must but fast and because of patent issues the most used algorithm). Another fact that makes JPEG unique at this point is the lossy compression, while GIF and PNG are lossless. In order to optimize JPEG files the recommended way is trimming of unnecessary image profiles, meta data and lowering the quality further (with information loss).

III. Automated Optimization

To illustrate the impact automated optimization can have on the size of images a program has been developed that fetches all images references in *img*-tags and from stylesheets references in *link*-tags. The test-set were the Alex top 100 websites of this world. After downloading all images to the local file system the following optimization tasks were executed:

- File type check → animated GIF files were not optimized
- Converted GIF files to PNG with ImageMagick[2]
- Automatically optimize the resulting PNG or the original JPEG file with imgmin[3]
- PNG files were also optimized with pngquant 2[4] and the result was compared to the imgmin result, the smaller file was then used
- The optimized file was converted to WebP

The last step was to generate a statistic of the achieved savings. This simple automated process was able to save 43.72 % with the usage of imgin and pngquant 2 (including the conversion of GIF to PNG). An impressive 68.99 % savings compared to the original file sizes was possible through conversion of the optimized files to Google's WebP format (the WebP result should be easy to improve even further, the problem is the lack of browser support). The full statistic is available online via the authors Github site at https://github.com/Fleshgrinder/se-research-seminar-1.

IV. Conclusion

Choosing the best compression method and applying simple optimization techniques can greatly improve the size of the images used on a website, without any or only little quality loss. The results of the program are more than promising and an optimized version that uses intelligent optimization techniques, improved calls to the existing tools or even extended libraries (e.g. Zopfli to replace gzip in PNG images) to compress the images could go much further.

[2] http://www.imagemagick.org/
[3] https://github.com/rflynn/imgmin
[4] https://github.com/pornel/improved-pngquant

References

[1] Alakuijala, Jyrki, and Lode Vandevenne. *Data compression using Zopfli.* Google, Inc., 2013.

[2] Huffman, David A. "A Method for the Construction of Minimum-Redundancy Codes." *Proceedings of the I.R.E.*, September 1952: 1098–1102.

[3] Lempel, Abraham, Jacob Ziv, and Terry A. Welch. "A technique for high-performance data compression." *Computer*, June 1984: 8–19.

[4] Manian, Divya, Tim Branyen, Nirzar Pangarkar, Sindre Sorhus, and Christian Schaefer. *Performance Evaluation of the Loading Spinner.* 2013. https://github.com/h5bp/lazyweb-requests/issues/103.

[5] Parbel, Matthias. *GIF ist jetzt frei von Patentrechten.* October 2, 2006. http://heise.de/-167490.

[6] Roelofs, Greg. *News and History of the PNG Development Group from 2007.* April 20, 2007. http://www.libpng.org/pub/png/png2007.html.

[7] Souders, Steve. *Compare Stats.* June 4, 2013. http://www.httparchive.org/compare.php?&r1=Jun%201%202013&s1=All&r2=Jun%201%202012&s2=All.

[8] —. *Interesting Stats.* June 1, 2013. http://httparchive.org/interesting.php?a=All&l=Jun%201%202013#bytesperpage.

[9] Storer, James, and Thomas Szymanski. "Data compression via textual substitution." *Journal of the ACM*, October 1982: 928–951.

[10] World Wide Web Consortium. *PNG versus GIF.* November 24, 2006. http://www.w3.org/QA/Tips/png-gif.