QUALCOMM®

Qualcomm Research

Uplink Data Compression

December 2014

Qualcomm Technologies, Inc.

Qualcomm Technologies, Inc.

5775 Morehouse Drive

San Diego, CA 92121

U.S.A.

Table of Contents

Figures

Tables

Introduction

The number of people using Smartphones is increasing globally, resulting in increased data consumption over HSPA networks. In conjunction with this rise in smartphone adoption, is an increase in the users' expectations of the quality of service they require of their smartphones. Today, users have come tonot only expect 'always-on' connectivity, but in particular expect a consistent quality of service experience, irrespective of whichcell they are in or how busy the cell is.

One way to help ensure, better user experienced involves over the air data traffic compression. Today, compression is performed at the application layer for most downlink traffic. For example, many websites send HTML documents in a compressed format and transmit all multimedia data (e.g. images, video and audio) in an encoded, and compressed, form. . On the uplink most of the data is typically protocol information such as IP/TCP/HTTP headers, and these are quite compressible. There are several benefits to compressing uplink data , including:

- By reducing the amount of data sent over the air, the UE consumes less of the fixed uplink bandwidth available to the base station as well aslowerstransmit power.
- From a user experience point of view, the severity of impacts due to low uplink bandwidth or limited transmit power is reduced. In particular, the user experiences faster web browsing response time in such types of scenarios.
- The shorter transmit durations and lower transmit power lead to reduced uplink interference. From a network point of view the reduction in uplink interference improves capacity and data rates for all UEs, including legacy ones.
- Alternatively, one can think of compression enabling the UE to send more data for a given bandwidth, hence improving the perceived speed of connection.

In the solution described in this document, the compression and decompression operations are entirely contained just above the current RLC layer. Thus, the only entities affected by this feature are the UE and RNC. All other entities such as the operating system, middleboxes (e.g., proxy servers, NATs, firewalls), servers are unaffected and are not even aware that data undergoes compression/decompression in flight.

The algorithms have been optimized such that the decompressor operation is straightforward, without requiring much computation. This ensures that the feature can be implemented as a software only upgrade at the RNC.

Motivation

There are several components of data on uplink that are candidates for good compression, such as TCP/UDP/IP headers, DNS requests and HTTP requests.

HTTP is the protocol used for retrieving webpages. Opening a webpage typically consists of the web browser sending tens of HTTP GET/POST requests. Each GET request is made to obtain an object such as

an HTML document, image, video, javascript or CSS file. These objects constitute various aspects of the website, and are processed by the browser to render the final webpage.

For example, the following is a GET request identified while loading the www.amazon.com web page:

```
GET / HTTP/1.1
Host: www.amazon.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:17.0) Gecko/20100101
Firefox/17.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Cookie: x-wl-
uid=1nm5D3WRA2mfidzfIEB+fgNN3svOpy/jEBIHq+h8CEk1kt1Cc2DgpNSAnsFxLQwL
5hgfY+3MipSY=; session-id-time=2082787201l; session-id=184-2472291-4052047;
csm-hit=535.90|; ubid-main=183-3799177-9039917; session-
token=N9MiwGi+ROWIfFDs0xTrsA51G5cgeauxP0guon1LbsyU6THBQWb7XrrnNAR9
wH6whoEYhZHJq5wRt8CTvuMl+eyIEVmpA3heAV8ijMKMW2mn7S29jSZhknM9/iOsu
q0AH1FO63UFXvvbDEf9n6z1taIQ9lNHwpkbaKwWmwTx20hF68aX7ac/qYxHVzWfbMl
oQx0S1lfHKqVpIqAdZX6eX5MsbEp8haGEfK+FI5p6EczKicv1iYtf9PRTcLdDd4QO8ZW
mzp+sudM=
Cache-Control: max-age=0
```

This GET is requesting the main HTML document for www.amazon.com. The request consists of various HTTP headers such as Host, User-Agent, Accept, Accept-Language, Cookie and others. Some of these HTTP headers, such as User-Agent, Accept-Language and Accept-Encoding do not change over time; and hence carry the same value in subsequent GET requests. The Cookie header, which is often the longest, is used by servers to identify the user over time and hence it is not uncommon for it to appear with the similar value across several GET requests. The Host header identifies the location from which an object is requested, and this may also stay the same across GET requests for sites that host all or a majority of objects in one place.

Thus, it is expected that there is significant redundancy across the stream of GET requests made in the course of downloading a webpage. Figure 1 quantifies the compressibility, using this algorithm, of uplink data across a broad range of websites as measured on a prototype implementation. The metric shown, compression factor, is defined as the ratio of the total number of bytes to total number of bytes after compression.

TCP/UDP/IP headers are compressible due to the repetitive nature of some of their fields. For a particular flow, the source and destination IP addresses and port numbers are repeated, without modification, on all packets belonging to the flow. Several other fields, such as TCP window and TCP urgent pointer are also often identical across packets.

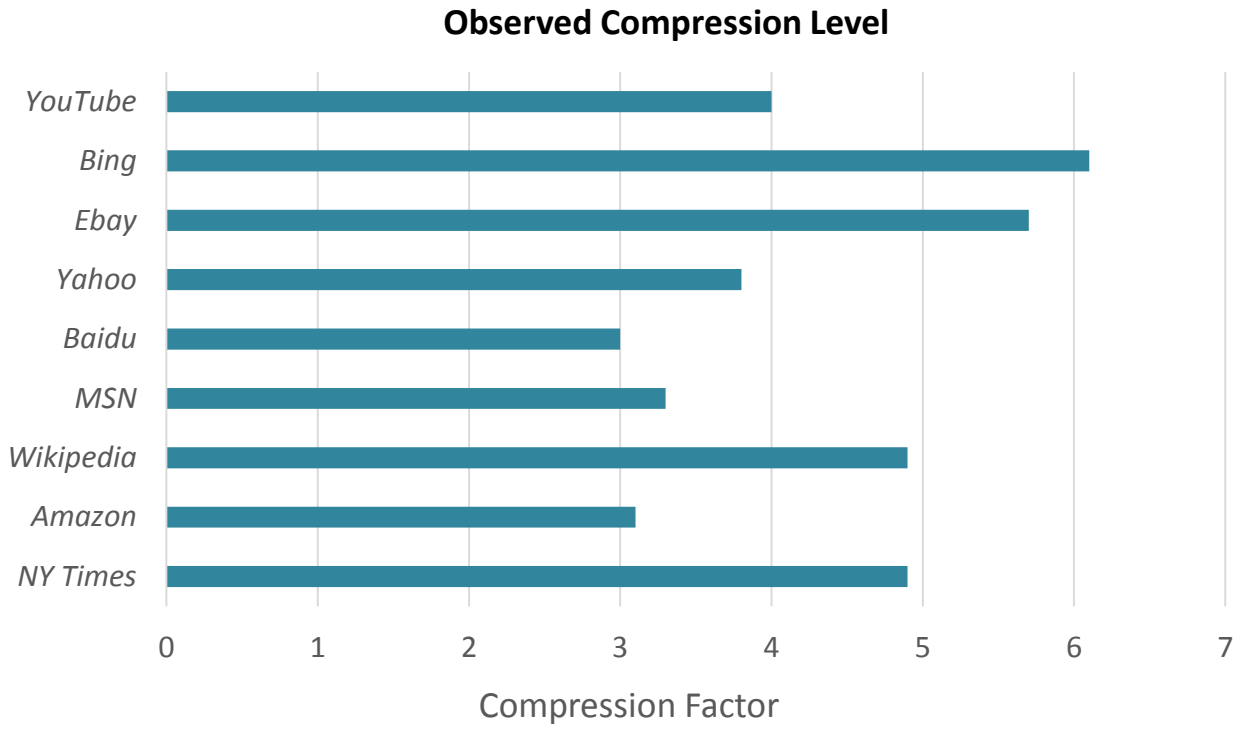

Figure 1 : Compression Factor for various popular websites

Architecture and Algorithm

As shown in Figure 2, the designed uplink data compression algorithm runs in the modem, just above the RLC layer. The RNC in the RAN runs the decompression algorithm. This architecture ensures zero impact to other entities such as operating system, middleboxes (e.g., proxy servers, NATs, firewalls), servers, and NodeB (in RAN). The compression algorithm operates on all uplink traffic passing through the modem, and does not need any information from the operating system or application. The algorithm ensures that it attempts compression only on flows that are compressible.

This architecture also allows the algorithm to take advantage of the functionalities of the RLC layer. The algorithm relies upon the RLC layer's properties of in-sequence delivery and error/loss-free delivery. In order to guarantee these characteristics, the RLC layer is operated in acknowledged-mode (AM). In addition, each compressed data unit carries a checksum that allows the decompressor to detect if there has been an error.

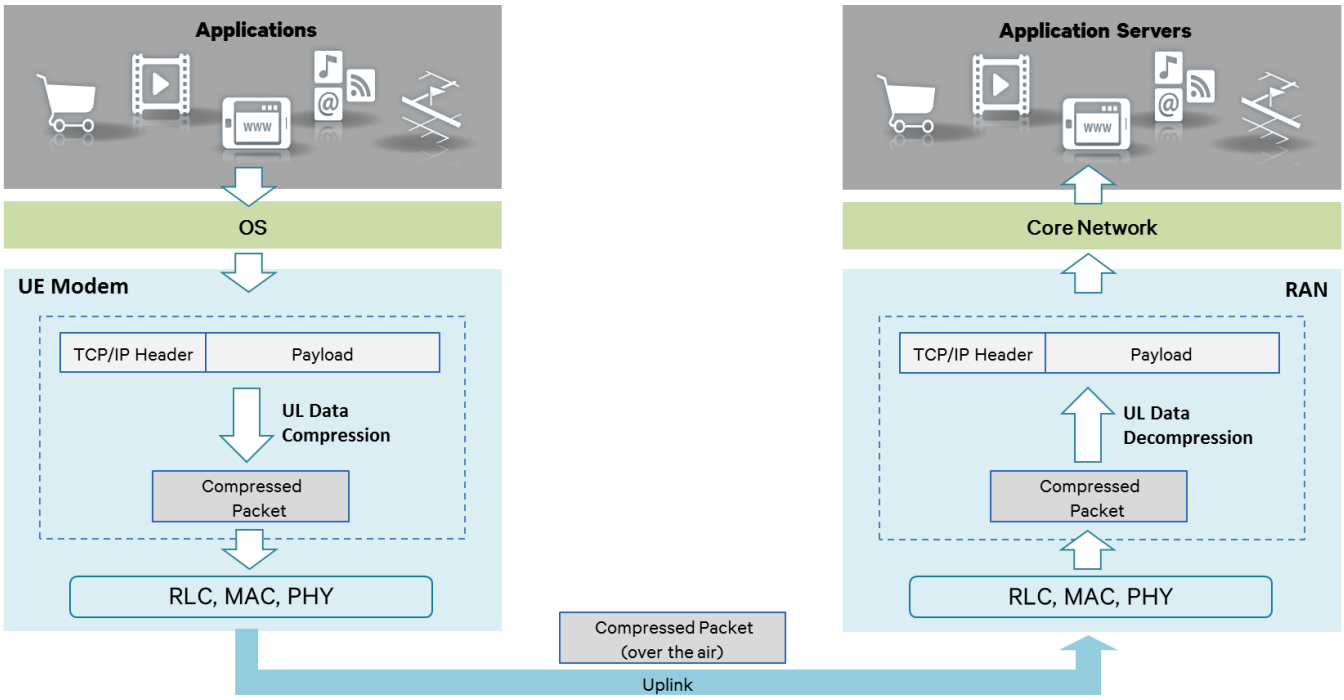

Figure 2 : Architecture of Uplink Data Compression

The algorithm maintains memory at the compressor and de-compressor so that it may keep track of the contents of past data packets. The compressor searches for strings of bytes in the current data packet that have occurred in past data packets stored in memory. Matching strings are then replaced by pointer metadata. Thus, reduction in packet size is achieved by replacing occurrences of long strings by short pointer metadata.

Upon receiving a compressed packet, the decompressor searches for pointer metadata and replaces the pointers with the strings of bytes that they refer to.

This mechanism needs the memories of the compressor and decompressor to be in sync, i.e., the compressor memory prior to compressing a packet should be the same in content as the decompressor memory prior to decompressing the packet. This property can be achieved if all packets arrive at the decompressor in the same order as they were processed by the compressor. This is guaranteed by the RLC characteristics of in-order delivery, error/loss-free operation.

Situations where these properties are not satisfied are the cases of unrecoverable loss leading to RLC reset, and UE mobility across RNC boundaries. In the case of RLC reset, the algorithm flushes the memories of the compressor and decompressor so that they are in-sync starting from the next packet. In case of mobility across RNC boundaries, a simple approach is for the RNC to disable the compressor/de-compressor prior to the transition and enable them post transition.

The user benefits of the described solution were measured on a prototype platform developed by Qualcomm Research. The network benefits were obtained through system simulation of users in a multi-cell scenario. Compressor/de-compressor memory size was set to 4kilobytes.

Error! Reference source not found. shows the reduction in page load time that results from UL compression, for a few popular websites. Since compression reduces the amount of data sent in the uplink, the HTTP requests are received and processed sooner by the webserver. Thus, the webpage loads faster. This benefit increases as the uplink data rate becomes more restricted (smaller), e.g. in conditions of lower power headroom and/or uplink grant. The gain varies across websites depending on the number and size of objects that need to be downloaded to render the page. Heavier sites such as www.nytimes.com see more gain than lighter ones such as www.amazon.com.

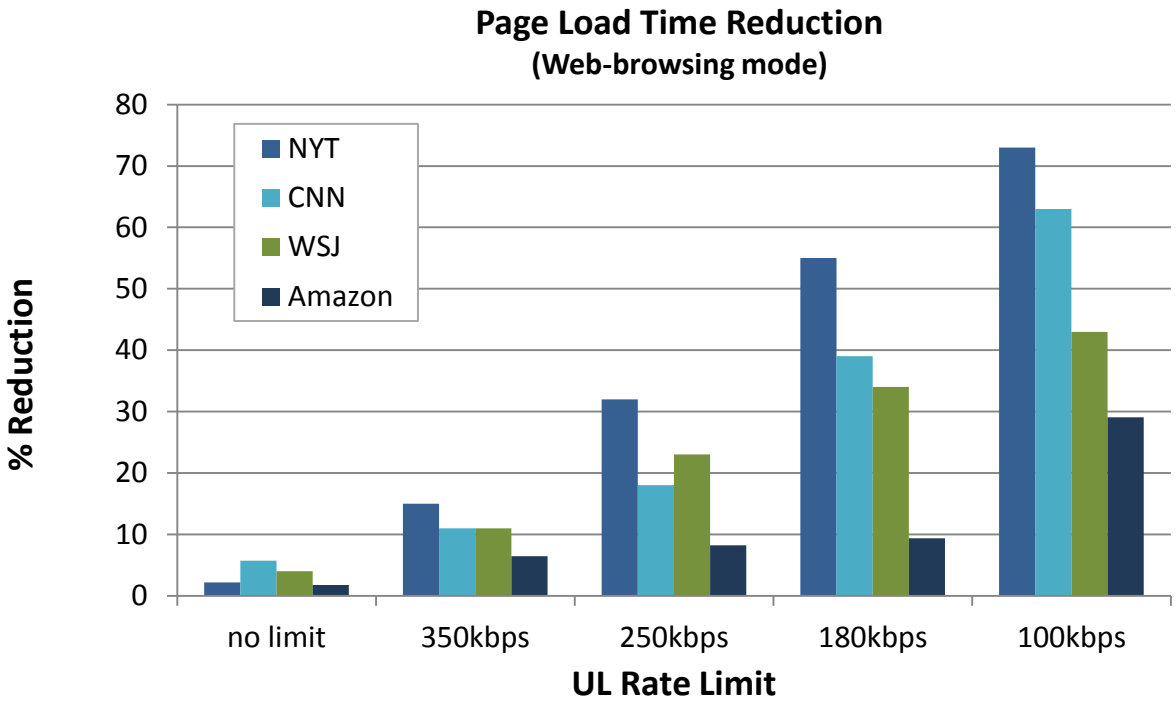

Figure 3 : Reduction in page load time due to compression

Figure 4 illustrates the reduction in page load time (news articles) using two popular android applications. Similar to Figure 3, the trend of increasing gain as the uplink data rate reduces is confirmed.

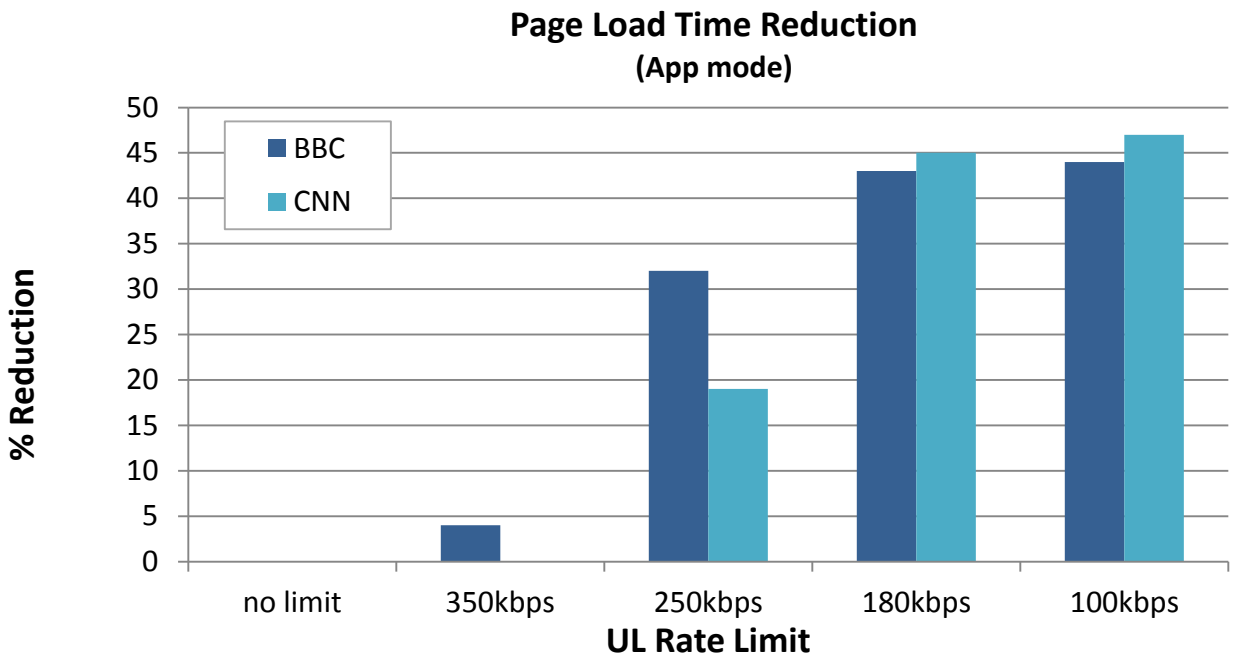

Figure 4 : Reduction in load time due to compression, using two popular news applications

Compression also reduces the amount of energy consumed at the UE when loading a webpage. Figure 5 shows the reduction in energy consumption measured over the time duration from user click to completion of the page load, for various values of UE pilot transmit power. Energy saving increases as uplink becomes more restricted. Field data from a metropolitan area suggests that at least 10% of time the pilot power is 15dBm or higher, leading to at least 45% reduction in energy consumption for browsing.

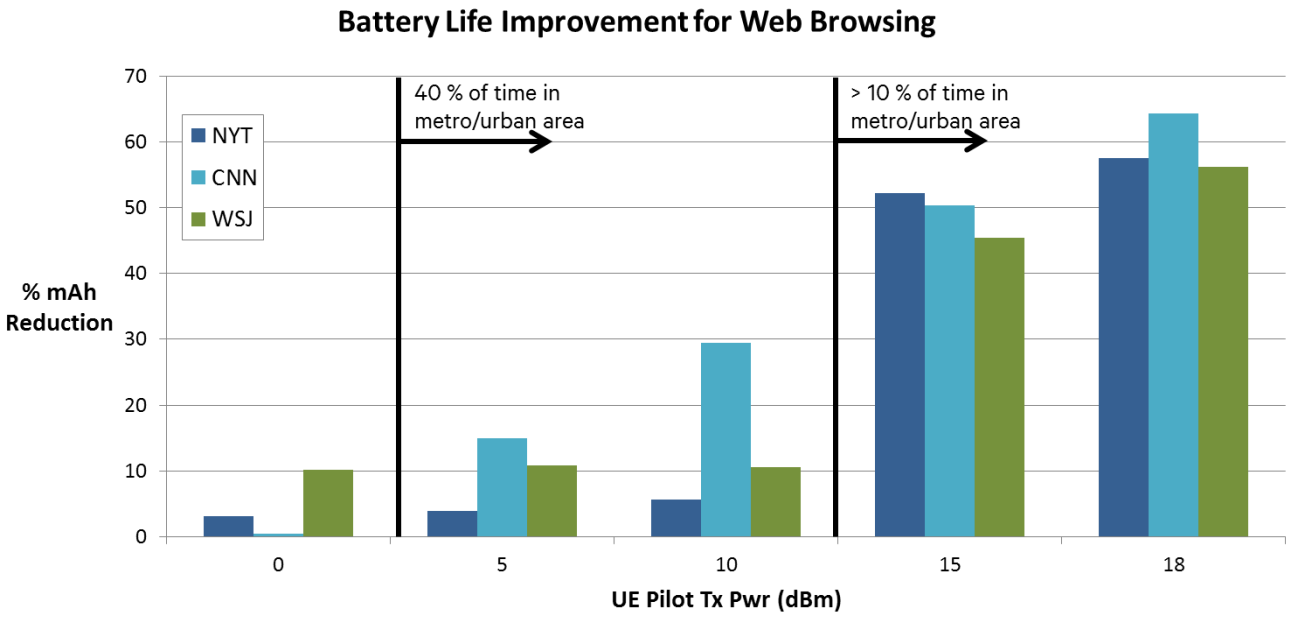

Figure 5 : Reduction in energy consumption, for web browsing, due to compression

Table 1 shows the benefit to legacy UEs from use of uplink compression by compression capable UEs. To quantify the gains, a 57-cell wrap around model simulation was performed. Each cell consisted of three types of UEs: full-buffer UEs that were not compression capable, web-browsing UEs that were not compression capable and compression capable web-browsing UEs. The number of full-buffer UEs per cell was fixed at 4. The number of web-browsing UEs was varied to obtain different traffic ratios between web-browsing and full-buffer UEs. To understand the impact of various degrees of penetration of the feature, we varied the fraction of web-browsing UEs that are compression capable. The amount of data generated by each web-browsing UE was small enough that the UE was not uplink bandwidth limited. Activating compression on web-browsing UEs reduced the UEs' contribution to uplink interference. This enabled the base station scheduler to allocate higher grant to the full-buffer UEs and in turn resulted in an increase in their throughput.

For the simulations, the UE locations were chosen randomly, a typical 3GPP channel mix was used with the EDCH TTI length set to 2ms and an HSUPA RoT target of 6dB. ISD was set to 1km.

Table 1: Throughput gain for legacy full-buffer UEs due to enabling compression on web browsing UEs

Throughput gain [%] for full-buffer UEs		Penetration of UL Data Compression UEs			
		25%	50%	75%	100%
Web browsing as a fraction of total traffic	1/3	9.5	19	28.6	38.2
	1/2	19.1	38.3	57.4	76.6

Conclusion

In this paper, we presented an overview of the design and benefits of compressing uplink data traffic in HSPA networks. Several benefits were measured on a prototype implementation. From a user experience standpoint, there is significant reduction in webpage load times, both for browsing and in-app loading. The user also enjoys longer battery life due to reduced energy consumption for page loads. Further, simulations show that even the users that do not adopt this feature see a benefit of increased throughput due to more network resources that are made available to them. From a network standpoint, the increased availability of resources can also be interpreted as an increase in cell capacity.

It is worth noting that this feature is relatively straightforward to implement. The entities affected by this feature are only the UE and the RNC. All other entities in the data path are unaware that data undergoes compression. In terms of protocol stack, the feature sits just above the RLC layer with minimal interaction with RLC, and does not need any cross layer information from the operating system or application. The decompression algorithm is designed to minimize computation. Since the RNC has many UEs connected to it, the overall memory and processing requirements on it is higher than that on an individual UE. Thus, an efficient decompression algorithm is crucial.