

# DELTA 'X' Reducing Size of an Android Application Update

Tushar Badlani

Department of Information Technology  
Genba Sopanrao Moze College of Engineering,  
Pune, India

Agasti Bhambure

Department of Information Technology  
Genba Sopanrao Moze College of Engineering,  
Pune, India

Sarang Dhere

Department of Information Technology  
Genba Sopanrao Moze College of Engineering,  
Pune, India

Saurabh Gupta

Department of Information Technology  
Genba Sopanrao Moze College of Engineering,  
Pune, India

**Abstract**— The number of applications (or apps) in the Android Market exceeded 600,000 in 2014 with more than 20 billion total downloads. The introduction of new features and correction of bugs and security vulnerabilities make it usual for mobile application developers to release new version of an application every month. Delta encoding based techniques that significantly reduce update traffic by transferring only the changes (or patches) between two versions of an application. The smartphone user can then download a smaller patch. We propose to use delta encoding algorithms and to download only the difference between two versions of an app. The DELTA 'X' algorithm unpacks APK and computes differences between decompressed application files, which allow it to produce much smaller patches.

**Keywords**— Delta Encoding, Android Application update, Data compression.

## I. INTRODUCTION

One of the most widely used mobile OS these days is Android. Android Inc. was founded in Palo Alto of California, U.S. by Andy Rubin, Rich miner, Nick sears and Chris White in 2003. Later Android Inc. was acquired by Google in 2005. After original release there have been number of updates in the original version of Android.

Android is a powerful Operating System supporting a large number of applications in Smart Phones. These applications make life more comfortable and advanced for the users. Hardware that supports Android is mainly based on ARM architecture platform. Main advantage and also a disadvantage of android OS, it is an open source.

### Applications:

These are the basics of Android applications:

1. Android applications are composed of one or more application components (activities, services, content providers, and broadcast receivers)

2. Each component performs a different role in the overall application behavior, and each one can be activated individually (even by other applications)

3. The manifest file must declare all components in the application and should also declare all application requirements, such as the minimum version of Android required and any hardware configurations required

4. Non-code application resources (images, strings, layout files, etc.) should include alternatives for different device configurations (such as different strings for different languages)

Advantages of domain:

1. Multi-Tasking
2. Android is an open source OS on the kernel LINUX
3. Deep integration with the third-party application system and can easily co-operate with them
4. Highly customizable

## II. RELATED WORK

Delta encoding is a well-known traffic reduction method. In 1997, Jeff Mogul and colleagues showed that delta encoding could reduce HTTP traffic by eliminating redundancy between the cached file copy and the new version to be downloaded. At that time, delta encoding enabled approximately 85 percent byte savings for cached files. Targeting a specific file format during delta encoding can also improve the compression rate and further reduce traffic. Google's Courgette, which encodes patches for the Chrome browser, benefits from exploring the specifics of the transferred binary executable files. The resulting patches are 10 times smaller relative to those possible with other delta encoding techniques. DELTA in turn targets Android Application Packages (APKs) and explores their internal structure to achieve smaller patch sizes.

Inmobile devices, over-the-air wireless downloads based on delta encoding are a popular way to distribute operating system updates. Besides Google Smart Application Update, which focuses on applications distributed through

Google Play, developers can use Update Direct for Android to update their Android applications. Applications compatible with Update Direct include a new library that lets the application update itself using a method based on delta encoding. Although, Update Direct reduces applications update traffic, applications can't be distributed through Google Play, and the mechanism doesn't support the Google App Engine. In contrast, DELTA reduces update traffic, and developers can distribute their applications through Google Play at no additional cost.

A survey conducted by the Ericson Mobility Report shows that monthly data traffic per Smartphone is 900 Mb. This will be 3500 Mb/Month by 2020. Each android device contains average of 35 applications. Android applications are got updated average of after 29 days. So total download data traffic is more than 2 gbytes per device for a month. Application update traffic is more than a 138 pbyte. This is nothing but huge data traffic [7].

### III. PROPOSED WORK

Total number of smartphones in India is over 67 million, and android share in the Indian market is over 90%.. Android has a strong grip in Indian market by launching more and more productive strategy plans. Recently Google launches project 'android one' which basically delivering a budget android phones for Indian people. In android one project Google guarantees for the android updates for next 2 years. Such strategies help Google to grow in Indian market.

Android have a huge app store consisting of more than 1 million applications. These applications are on average size of 6 Mb. In the game section, average game size is 40 MB.

So in order to reduce the whole data traffic, we use Delta Encoding method is used which we discussed in this paper. In science, engineering, and mathematics, the Greek letter delta ( $\Delta$ ) is used to denote the change in a variable. The term delta encoding, refers to several techniques that store data as the difference between successive samples (or characters), rather than directly storing the samples themselves. The first value in the delta encoded file is the same as the first value in the original data. All the following values in the encoded file are equal to the difference (delta) between the corresponding value in the input file, and the previous value in the input file.

In this paper a new mechanism for application update is proposed, named DELTA X. It provides substantial reduction in traffic produced by application updates. Delta ++ was the previous encoding Method for computing the difference or "diff", between the two files. This difference is utilized to create the advanced version from the previous version of the file. A smart phone application is updated by downloading only the difference of old version and the new one and applying the delta patch in smart phone. The key difference of DELTA++ with Google Smart Application Update is the decompression of Android APK package and compression is performed on each modules of the APK. Google's method doesn't perform this. Experimental results prove that application updates is reduced in size by 77% in average with DELTA++ whereas 55% reduction in size on

average for Google Smart Application Update or DELTA. Such a reduction in use of network bandwidth comes with a trade-off. When using DELTA++, more time is required to provide the application patch on the smart phone due to the increased patch complexity. This delay is accepted for updates for smart phone applications because the users don't want an update immediately when it is released. This disadvantage is been further reduced as in DELTA X reduces the deployment time further. Additional use of battery is found to be negligible. As a study says, if less data traffic is used by a smartphone, its battery usage is also reduced. Implementation of DELTA X, evaluation of the additional energy used for DELTA ++, a comparison between DELTA++, DELTA or Google Smart Application Update and the estimation for large scale savings achieved by full-scale deployment of DELTA X etc. are performed in this project. The DELTA X application is an advanced version of DELTA++ as it overcomes the disadvantages of DELTA++.

The DELTA 'X' method overcomes some of the drawbacks of the DELTA++ like the major disadvantage of the deployment time of the DELTA++ and the size of the patch is also reduced in DELTA X.. The DELTA 'X' method keeps track of the deleted and the updated files in the updated application.

### THE DELTA 'X' METHOD

It is proved previously that DELTA++ (Delta Encoding for Less Traffic for Apps) is useful to reduce application update traffic. DELTA++ is based on delta encoding tool, bsdiff [] and it enables savings in data centres and mobile networks. A new DELTA X method is introduced and by implementing this, further reduces the transmitted package size and attains greater savings. Patch size is computed using delta differencing algorithm which mainly depends on the total difference between two files. The patch size is influenced by the use of compression techniques. Two files may be slightly different but, their compressed versions may have great difference on binary level. This is due to the ways through which they are passed during compression. Similarly happens in the case of APK application package, too. It is basically a compressed archive of all files belonging to an Android application. The concept of DELTA++ is to calculate difference between the application files of APK instead of compressed APK packages themselves. Original DELTA method produces delta difference of old version APK file of application with the new version. The delta patch is generated on the server by using bsdiff delta encoding tool. The bspatch tool is used to deploy the patch in the smart phone.

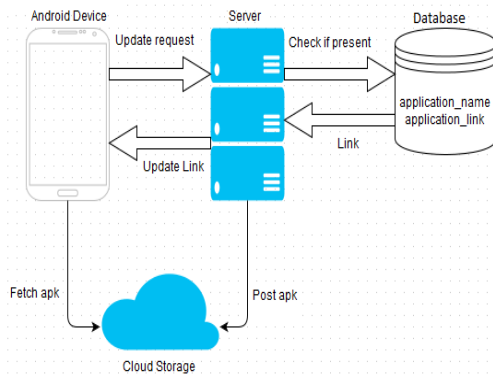


Figure 1 Delta X architecture

DELTA works similar to Google Smart Application Update and it will not unpack the APK file. DELTA++ is improved on DELTA by exploiting the specific structure of APK package and decompressing it. Thus, patch of much smaller size will be produced in this way. The DELTA++ method introduces the manifest file which contains the information about the application. Like the DELTA++ method, Delta X is also divided into two: 1. patch computation and 2. patch deployment. Patch computation is performed on the server side of the data centre. It need to be done only once for patch version of each application. Patch deployment is carried out on the user's android smart phone and it is repeated when an application is updated. The procedure for DELTA X patch is followed:

1. The old version and the new version APK packages are decompressed.
2. To obtain names, SHA-1 hash digests and paths of each file of both APK packages, the manifest files of both are traversed.
3. The files of new version are flagged as NEW (if the file is Present only in the new version, not in the old version), UPDATED (if both versions have same file but SHA-1 sum is different), SAME (if both versions contain the file but deleted in the new version) or DELETED (if old version contains the file but deleted in the new one).
4. The latest version files marked as NEW are copied into the constructed patch.
5. The files marked as UPDATED are the latest version of the input of bsdiff delta encoding algorithm which determines the difference of the old version and the new one. Then the computed difference is copied into the constructed patch. In some cases, the difference of small files may be greater than their individual sizes. This is due to the overhead along with the creation of delta file. Here, the new file is again marked as NEW and it is copied into the patch.
6. If the files have no change they are marked as SAME and are kept untouched.
7. PatchManifest.xml file is created and it is included in the patch. It acts as a description and includes information about the application version that can be updated using this patch. It also provides information such as NEW files included in the patch and delta differences determined between UPDATED

files. PatchManifest.xml file also contains information about files marked as DELETED.

8. The APK package of the current version is loaded by using the application info class.
9. Using the Manifest.xml in the patch, the unwanted files of the old version are deleted.
10. All the differences in the patch are applied, Thus updating the file.
11. In the final stage, the compression of the patch is done. Now, the patch is compressed and ready to be sent to the Android device.

Now, let's see how to deploy DELTA X patch in the Android device.

1. The received patch is decompressed into any temporary directory.
2. All NEW files of patch are copied into the old version of application. Thus, the old and the new version of application contain the same files.
3. Finally, the Android Package Installer, the built-in application is used to install the resulting APK package thus completing update of application.

DELTA X has been implemented as server side application. It is responsible to construct patches and serve them by request. A compressed APK is created on the server and then deployed.

#### IV. RESULT ANALYSIS

The analysis of the results is done with the help of experimental analysis. The experimented results are analyzed on the basis of the 2 factors:

1. Data Consumption
2. Battery Usage.

##### 1. Data Consumption:

As the graph shown below, it takes 42% of the actual application size when you update using the current mechanism (Google Smart Application Update). It takes 19% of the total app size in Delta++ and it takes only 15% of the total size using Delta X. So, if we take an example of an app of 100 MB, Google's Smart Application Update would take about 42 MB and whereas it will take 19 and 15 MB respectively for Delta ++ and Delta X. As a result. If you take a cellular data cost of 10paise/10 KB, it will save an approximately **Rs.100** per 100 MB download.

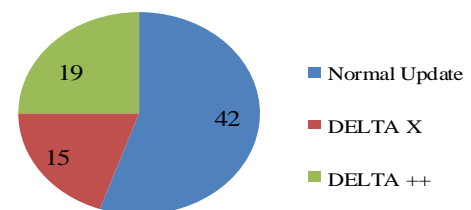
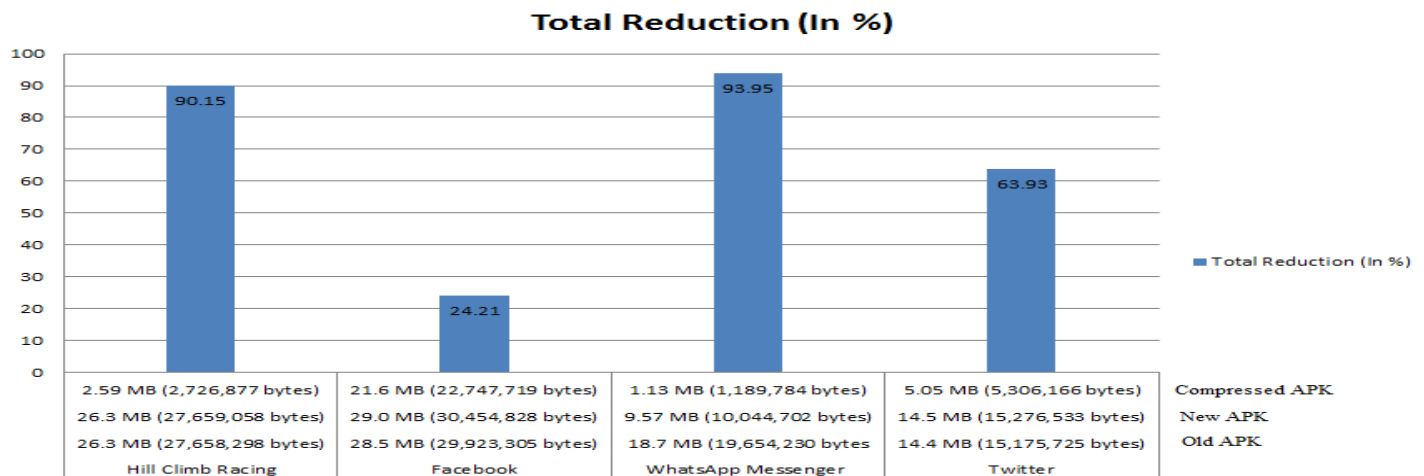


Figure 2 Cellular Traffic Reductions

## 2. Battery Consumption:

The Application Update does not affect the battery usage by much. The usage of battery in Delta X is negligible. IT uses the same amount of battery as the previous version DELTA++.

Experimental Analysis was conducted on some apps of different sizes to check the result and analyse them. It was found out that the DELTA 'X' method reduces an approximately 70 percent of the total size of the application. The experimental Results show that the data reduction in the updates is around 15 percent more than the Google Smart Application Update. It was also found out that if there are a lot of changes in the application like the complete GUI changes or any other drastic change in the application, the data reduction by DELTA 'X' method is not much.



It was estimated that annual application update traffic in India can be reduced significantly with the help of DELTA X. Accordingly the experimental results, the updated size of application is dependent on the various compression techniques used by the developers. Bandwidth is reduced as we are compressing the whole apk and then deploying it. This will reduce the load on the network. Estimated traffic reduction in India can be 2 gbytes per month and approximately 136 pbytes per annum. Surely DELTA X method is very beneficial in reducing the size of an android application update.

Apple iPhone also have limited but consistent users. Apple users in India are more than 4 million. But reason behind considering apple iPhone is, apple also have a huge app store. Here average application size is 26 mb and in game section, it is average of 60 mb. Another reason to consider apple iPhone is that it is our future work. We can expand our work for the apple products so as to reduce total data traffic. We can work on the DELTA 'X' method for iPhone which will be more useful as compared to android

## REFERENCES

1. Android operating system. Source code. URL: <http://source.android.com>
2. N. Samteladze and K. Christensen, "DELTA: Delta Encoding for Less Traffic for Apps," Proceedings of IEEE Conference on Local Computer Networks, pp. 212-215, October 2012.
3. N. Samteladze and K. Christensen, "DELTA++: Reducing the Size of Android Application Updates," submitted to IEEE Internet Computing in February 2013.
4. <http://www.webenza.com/android-vs-ios-smartphone-users-in-india-2014-infograph/>
5. <http://dazeinfo.com/2012/10/20/increasing-size-of-apps-could-be-a-major-challenge/>
6. <http://www.appbrain.com/stats/stats-index>
7. Ericsson Mobility Report, November 2014.
8. Google Play Store. Top free apps, 2012. URL: [https://play.google.com/store/apps/collection/topselling\\_free](https://play.google.com/store/apps/collection/topselling_free)
9. H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin, "A First Look at Traffic on Smartphones," Proceedings of the ACM SIGCOMM Conference on *Internet Measurement*, pp. 281-287, November 2010.
10. K.-K.Yap, T.-Y. Huang, M. Kobayashi, Y. Yiakoumis, N. McKeown, S. Katti, and G. Parulkar, "Making Use of All the Networks Around Us: A Case Study in Android," Proceedings

- of the 2012 ACM SIGCOMM workshop on Cellular *networks: operations, challenges, and future design*, pp. 19-24, 2012.
11. K. Lee, I. Rhee, J. Lee, S. Chong, and Y. Yi, "Mobile Data Offloading: How Much Can Wi-Fi Deliver?," *Proceedings of the 6th International Conference*, p. 26, 2010.
  12. M. Welsh, "Data Compression in Chrome Beta for Android," The Chromium Blog, March 5, 2013. URL: <http://blog.chromium.org/2013/03/data-compression-inchrome-beta-for.html>
  13. M. Stolikj, P. Cuijpers, and J. J. Lukkien, "Efficient Reprogramming of Wireless Sensor Networks Using Incremental Updates and Data Compression," 2012.
  14. J. Seward, "The bzip2 and libbzip2 Official Home Page," 2013. URL: <http://www.bzip.org>.
  15. S. Musil, "Google Play Enables Smart App Updates, Conserving Batteries," CNET News, August 16, 2012. URL: [http://news.cnet.com/8301-1023\\_3-57495096-93/google-play-enables-smart-app-updates-conserving-batteries/](http://news.cnet.com/8301-1023_3-57495096-93/google-play-enables-smart-app-updates-conserving-batteries/).
  16. Google Mobile Blog, 2012. URL: <http://googlemobile.blogspot.com>.