Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN : 2348-4470

p-ISSN : 2348-6406

International Journal of Advance Engineering and Research Development

Special Issue on Recent Trends in Data Engineering

Volume 4, Special Issue 5, Dec.-2017

Data Compression using Wireless Distributed Computing

Ivan Pillay[1], Bhushan Mali[2], Siddhesh Pande[3], Shubham Soni[4].

[1]Department of Computer Engineering, AISSMS IOIT
[2]Department of Computer Engineering, AISSMS IOIT
[3]Department of Computer Engineering, AISSMS IOIT
[4]Department of Computer Engineering, AISSMS IOIT

Abstract - *In this new era of technology, smartphone is used as a multi-purpose device. Also due to development in hardware in recent years it has become possible to perform multiple tasks simultaneously or perform complicated task just on a smartphone. On an average, people replace their smartphone every 20 months. which implies that there is lot of computational power available at the user end which is either idle or used infrequently. We propose an application ofdistributed data compression, which utilizes these available devices to execute its own tasks. The process of data compression is offloaded to nearby devices which also conserve resources on user's device. This application distributes the task of data compression among nodes based on their resource constraints. Thus high capacity devices receives larger chunk of data as compared to others.*

Keywords– *Data Compression, Distributed Computing, Smartphones, Wireless local area network.*

I. INTRODUCTION

Recently we have seen how fast technology is growing especially in mobile market. Devices are getting smaller, cheaper and more robust. Smartphones are used for tasks which were at one point expensive or required more physical interaction with it such as to-do lists. Modern day smartphones are equipped with processors and memory comparable with a traditional desktop system. They are also capable of interacting with devices over the network, decode different media format files and so on. Smartphones come on an average of 1 GHz processor, 1 GB Ram and 2400mAh battery which are enough for executing graphic and computational intensive applications.

These specifications enable a user to do their tasks primarily using smartphones. As a result there is an increase in data generated at user's end. Modern smartphone operating systems typically occupy quarter to half of available storage on device. The file size of image and videos captured by on device camera, graphic intensive games as well as other applications occupy significant portion of a user's device. Some of these files even though important but may not be used frequently and there might such files in large volumes. Thus in order to reduce storage consumption we can use data compression. Since data compression is computationally intensive task we propose data compression using distributed architecture.

We will exploit the resources available nearby user to execute our compression task. Most of smart phones come equipped with Wi-Fi which we will use to create a temporary network of nearby devices. Task offloading will depend on each device's resources such as battery and free space. Device with higher capacity gets higher chunk of original file for compressing. Where as if a device does not enough resources it will not be allocated any chunk of file. This distribution will help the user's device to save some computational power and energy which can be used by other applications.

II. RELATED WORK

This[5] paper discusses load balancing techniques for wireless devices. Their proposed Adaptive LoadBalancing (ALB) algorithm works by initializing all available nodes Ng. A master node is selectedi.e. N1 that has knowledge of all other nodes including channel conditions and available energylevels. The master node then decides how data should be routed or processed by selecting nodesthat have good processing power as well as their corresponding channel condition. ALB algorithmdistributes tasks based on available energy levels of the selected nodes i.e. nodes with good channelconditions will get more share of the task than the nodes with poor conditions.

This[7] algorithm isused in spectrum sensing technology. This paper presents compression of textual data on mobile devices. It uses few algorithms and compare the results among them to analyse suitable algorithm for different types of textual data. It uses Burrows-Wheeler transform, Move-to-front, Arithmetic coding techniques to compress data. It also demonstrates the usage of these algorithms on mobile phones.

This [10]paper discusses a scalable framework for wireless distributed computing.The algorithm divides a huge database small enough that can be stored in each of the nodes. This reduces the total number of access-requests required by the nodes to access the Database. All the node processes on the data that is locally available and share its results with other nodes. The other nodes then use this intermediate values for their operation thus reducing the computational time required to process entire data-set.

II. SYSTEM MODEL

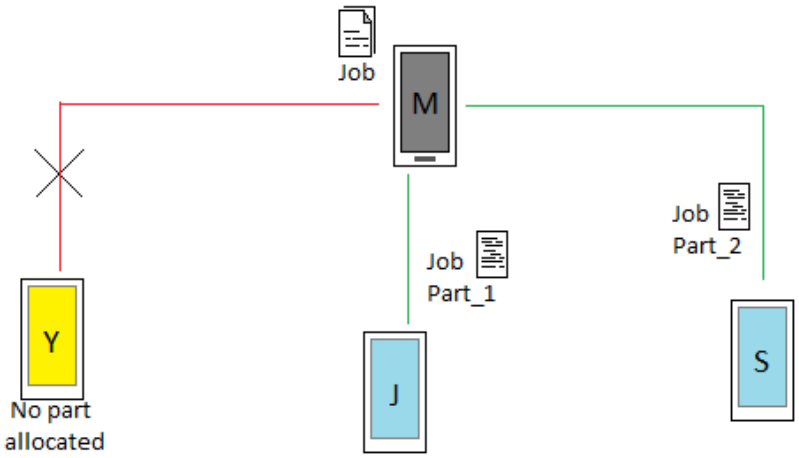

Figure 1. System Architecture

As seen in Figure 1, the Master device **M** creates a network to which all nearby Slave devices join using Wi-Fi Direct. Slave Devices automatically detect **M**. **M** needs to confirm connection request sent by each device, this is a part of Wi-Fi Direct security. Upon finalizing peer count, the distribution for each device is calculated based on their free space and battery level. Parts of file are send to each of the device in the peer group as per allocated size. Device **Y** will not be allocated any task due to insufficient resources hence compression job will be divided among **J** and **S** only.

Peer devices i.e., **J** and **S** upon receiving data, compresses it then sends the result back at source address i.e., to Master device**M**. **M** after confirming data reception from each peer device concatenates all the parts to create a single compressed file. All of the temporary files created at peer devices are deleted after operation. Since all the devices are connected via Wi-Fi Direct, they can still be connected to any other wireless network including Wi-Fi simultaneously.

III. ALGORITHMS

A. Move to Front

This is an encoding algorithm i.e., it makes data sorted in a way which is easier to compress as compared to original data. This algorithm works as follows:

//Encoding.
1. Initialize character set **S**index list **I**.
2. C =read next character from input stream.
3. Store index of C occurring within **S** in **I**.
4. Move C at front of the set **S**.
5. Repeat steps 2-4 until end of file.

//Decoding
1. Obtain index list **I**. and initialize char set **S**.
2. C = read next character from **S** at indexfrom **I.**
3. Output C.
4. Move C at front of the set **S**.
5. Repeat steps 2-4 until entire **I** is traversed.

B. Burrows Wheeler Transform

Burrows wheeler is a coding algorithm and is more suitable for text data. It works as follows:

//Encoding
1. Create an empty table **T**.
2. Prefix data with START character and suffix data with END character.
3. Add original data as record in **T**.
4. Rotate left all characters in data by one place and add the result to **T**.
5. Repeat step 4 until next character i.e., second character is START.
6. Sort **T**.
7. Output last column and index of original data in **T**.

//Decoding
1. Create an empty table **T**.
2. Obtain BWT output **B** and Index **I**.
3. Add **B** in last column of **T**.
4. Sort **T**.
5. Prepend **B** in **T**.
6. Repeat steps 4-5 until record length becomes equal to original data length.
7. Output record at **I**.

C. Lempel-Ziv-Welch

This is a compression algorithm that will be applied on result obtained from above mentioned coding algorithms.This algorithm assigns fixed bit code to each character and assigns additional codes to sequences encountered as they are observed. It buffers input data until the buffer + next byte does not exist in dictionary, it then outputs the code for buffer (excluding data).

//Encoding
1. Create a dictionary **D** and add all symbols from charset in it.
2. Search for longest sequence **S** in **D** that match current input set.
3. Store the code for S and delete S from input set.
4. Add **S** appended by next symbol to the dictionary.
5. Repeat steps 2-4 until end of input stream.

//Decoding algorithm is same as encoding.
1. Create a dictionary **D** and add all symbols from charset in it.
2. Search for longest sequence **S** in dictionary that matches with input read from encoded data stream.
3. Store the code for **S** and remove **S** from current input set.
4. Add **S** appended by next symbol in encoded data to dictionary.
5. Repeat steps 2-4 until end of stream.

IV. DATA PROCESSING COST

A. Data Transmission

Let's consider total n devices present in the system. The whole file F is divided into n-1 chunks of data F1, F2, F3 …Fn-1. The offloader sends partitioned data to rest of the other devices available. The time required to transmit i^{th} chunk to i^{th} device is:

$$t_d = \frac{F_i}{B_i}$$ *--where B is bandwidth*

B. Data Compression

The time required to execute compression algorithm on i^{th} device is given as:

$$t_c = \frac{F_i}{C_i}$$ *--where C is time required to execute algorithm.*

C. Total Cost

$$T = t_d + t_c$$
$$= \sum_{i=0}^{n} \frac{F_i}{B_i} + \sum_{i=0}^{n} \frac{F_i}{C_i}$$
$$= \sum_{i=0}^{n} F_i \left(\frac{1}{B_i} + \frac{1}{C_i} \right)$$

IV. CONCLUSION AND FUTURE WORK

Thus we have studied compression algorithms and how computational power available from nearby devices can be exploited. This application will be useful for users who carry most of their data on smartphone only. Wireless Distributed Computing can be used to conserve resources on master device which then can be used by other applications. Since all these operations are done locally there is no need for Internet or an access point. This application can be further extended in future to distribute other tasks (that can be divided into sub tasks) as well without relying on external resources such as Internet. Allocating resources in multihop network is also possible enabling the user to take advantage of multiple networks available nearby.

V. REFERENCES

[1] Hong Quy Le, Hussein Al-Shatri and Anja Klein, "Optimal Joint PowerAllocation and Task Splitting in Wireless Distributed Computing",in SCC2017; 11th International ITG Conference on Systems, Communications andCoding, 2017.
[2] G. Massari, M. Zanella and W. Fornaciari, "Towards Distributed MobileComputing", in Mobile System Technologies Workshop (MST), 2016.
[3] J. Ziv and A. Lempel, "On the Complexity of Finite Sequences", in IEEETransactions on Information Theory, 1976.
[4] Sergio De Agostino, "Lempel-Ziv Data Compression on Parallel and DistributedSystems", in Data Compression, Communications and Processing (CCP),2011.
[5] Mohammed M. Alfaqawi et al, "Adaptive Load Balancing Algorithm ForWireless Distributed Computing Networks", in Intelligent Systems Engineering(ICISE), 2016.
[6] Gerardo Calice, Abderrahmen Mtibaa, Roberto Beraldi et al, "Mobile-to-mobileopportunistic task splitting and offoading", in Wireless and Mobile Computing,Networking and Communications (WiMob), 2015.
[7] Eko Darwiyanto, Heru Anugrah Pratama and Gia Septiana, "Text datacompression for mobile phone using burrows-wheeler transform, move-to-frontcode and arithmetic coding", in Information and Communication Technology(ICoICT), 2015.
[8] Jouni Siren, "Burrows-Wheeler Transform for Terabases", in Data CompressionConference (DCC), 2016.
[9] J. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression", in IEEE Transactions on Information Theory,vol 23, 1977.
[10] Songze Li, Qian Yu, Mohammad Ali Maddah-Ali and A. Salman Avestimehr, "A Scalable framework for Wireless Distributed Computing", in IEEE/ACMTransactions on Networking, 2017.

