Image Processing by Map Reduce Using Serialization

Dr.Parashuram Baraki[1], Dr.V.Ramaswamy[2] & Mr.Fairoz.B[3]

[1]*Professor and Head, Computer Science & Engineering Department, Hirasugar Institute of Technology, Belagavi , Karnataka, India,*

[2]*Professor and Head, Computer Science & Engineering Department, S R C, Kumbakonam, SASTRA University, Tamil Nadu, India*

[3]*Technical Head, Product Design and Development Team, Vidvath Technologies Private Limited, Karnataka, India.*

Abstract- Nowadays, the sizes of photograph collections are increasing dramatically and accomplishing petabytes of records. Such big volumes cannot be analyzed on personal pc within a reasonable time. Therefore, processing of present day picture collections calls for dispensed computing. This paper offers a MapReduce Image Processing framework (MIPr), which affords the ability to apply allotted computing for photograph processing. MIPr is primarily based on MapReduce and its open supply implementation Apache Hadoop. MIPr offers diverse kinds of image representations in Hadoop inner layout and the input/output equipment for integration of photograph processing into Hadoop facts workflow. The picture codecs in the MIPr framework are based totally at the famous photo processing libraries. Furthermore, the MIPr includes the excessive-stage Image processing API for builders who are not familiar with Hadoop. This API lets in to create sequential capabilities that process one picture or a set of associated photographs. The MIPr framework applies such functions to the huge quantity of photographs in parallel. In addition, MIPr consists of MapReduce implementations of popular photograph processing algorithms, which may be used for dispensed picture processing with none software program development. The MIPr framework notably simplifies picture processing in Hadoop allotted environment.

Keywords- Image processing, mapreduce, hadoop,Serialization

I. INTRODUCTION

HIPI is an photo processing library designed to be used with the Apache HadoopMapReduce parallel programming framework. HIPI allows efficient and high-throughput image processing with MapReduce fashion parallel programs usually achieved on a cluster. It provides a solution for the way to keep a large series of photos on the Hadoop Distributed File System (HDFS) and make them available for efficient distributed processing. HIPI also presents integration with OpenCV, a popular open-source library that carries many computer vision algorithms (see covar example software to research extra about this integration). HIPI is evolved and maintained by using a growing number of builders from around the sector.

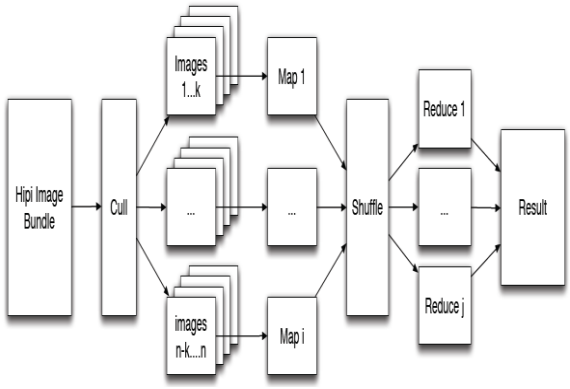

Fig.1: System Design

This diagram shows the organization of a typical MapReduce/HIPI program:

The number one input object to a HIPI software is a HipiImageBundle (HIB). A HIB is a set of images represented as a single document on the HDFS. The HIPI distribution consists of several beneficial tools for creating HIBs, inclusive of a MapReduce software that builds a HIB from a listing of pics downloaded from the Internet. The first processing stage of a HIPI application is a culling step that allows filtering the photographs in a HIB primarily based on a ramification of person-defined conditions like spatial resolution or criteria related to the photo metadata. This functionality is carried out via the Culler class. Images which can be culled are in no way completely decoded, saving processing time. The pics that continue to exist the culling degree are assigned to person map duties in a manner that tries to maximize facts locality, a cornerstone of the HadoopMapReduce programming model. This functionality is accomplished thru the HibInput Format elegance. Finally, character photos are offered to the Mapper as objects derived from the HipiImage summary base class along side an associated HipiImage Header object. For example, the Byte Image and Float Image training make bigger the HipiImage base elegance and provide get admission to to the underlying raster grid of picture pixel values as arrays of Java bytes and floats, respectively. These lessons

offer a number of beneficial functions like cropping, color space conversion, and scaling. HIPI additionally consists of assist for OpenCV, a popular open-source pc vision library. Specifically, image classes that extend from Raster Image (such as Byte Image and Float Image, discussed above) can be converted to OpenCV Java Mat items the usage of workouts within the OpenCVUtils class. The OpenCVMatWritable elegance affords a wrapper around the OpenCV Java Mat class that can be used as a key or value object in MapReduce programs. See the covar example software for more designated facts about a way to use HIPI with OpenCV. The records emitted by using the Mapper are accrued and transmitted to the Reducer according to the integrated MapReduceshuffle algorithm that attemps to reduce community visitors. Finally, the person-described reduce tasks are done in parallel and their output is aggregated and written to the HDFS.

II. IMPLEMENTATION

To demonstrate how the hadoopmapreduce framework canbe extended to work with image data for distributed imageprocessing.

Distributed Image Processing Problem Statement

Easy to apprehend and implement, and at the same time, it is computationally high priced. Have the belongings that the new value to be calculated for a pixel depends not only on that pixel's unique fee but also at the values of surrounding pixels. An image may be considered as made of a dimensional array of pixels. We can partition the photograph records into subsets and function on the information in parallel with the aid of distributing subsets to extraordinary map duties.

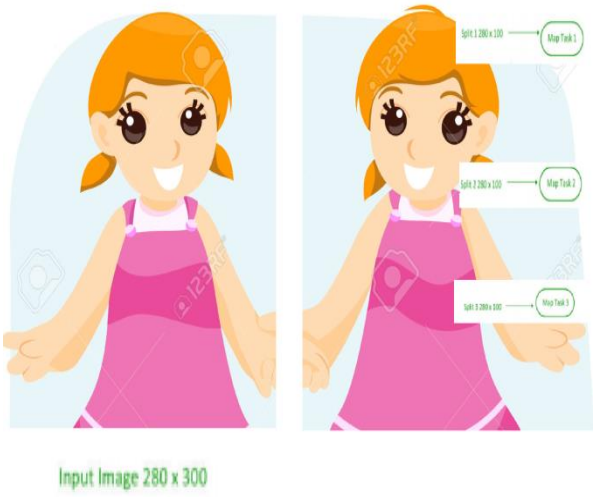

Fig.2: Distributed Sobel Edge Detection algorithm

Algorithm

Convolution at a single pixel

1. Create kernel h indexed from 0 to m-1 horizontally and 0 to n-1 vertically and populate it with kernel coefficients
2. Compute kernel half width, m2 = floor(m/2)
Compute kernel half height, n2 = floor(n/2)
3. sum = 0
for k = -n2 to n2 loop
for j = -m2 to m2 loop
sum = sum + h(j + m2, k + n2) f(x - j, y - k)
end loop
end loop
g(x, y) = sum

Convolution of an image ignoring the borders

1. Create kernel h indexed from 0 to m-1 horizontally and 0 to n-1 vertically and populate it with kernel coefficients
2. Compute kernel half width, m2 = floor(m/2)
Compute kernel half height, n2 = floor(n/2)
3. Create an M x N output image, g
4. for all pixel co-ordinates, x and y, loop
g(x, y) = 0
end loop
5. for y = n2 to N-n2-1 loop
for x = m2 to M-m2-1 loop
Compute g(x, y) using previous algorithm
end loop
end loop

III. IMPLEMENTATION

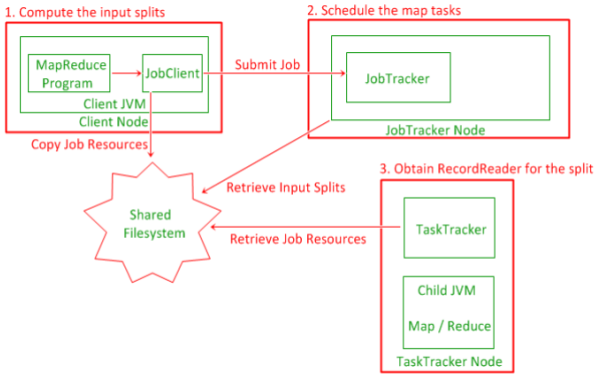

Fig.3: implementation

Serialization

How to serialize a BufferedImage the usage of Hadoop's serialization framework?

1. Serialize RGB color facts Drawback: Color conversion takes region if the default version does not in shape the photograph ColorModel.
2. Serialize raw pixel facts as a byte array Drawback: Requires conditional code based on image type.

Three. Serialize in a favored photograph layout using ImageIO

Advantages:

1) Do now not need to deal immediately with underlying pixel information representation.
2) Intermediate photo codecs may be specified using custom properties.
3) Default PNG format used gives lossless zip compression.

Output Format

Responsible for,

1. Writing output for the process to the report machine (getRecordWriter)
2. Checking for validity of output specification of activity (checkOutputSpecs)

Record Writer

A Record Writer is used to write the output key-price pairs to an output document.

Image Output Format

Image Output Format is designed much like the Multiple Output Format summary class, which lets in you to put in writing facts to more than one files whose names are derived from the output keys and values.

Image Output Format affords a Record Writer implementation for writing the picture to the file system.

Key: image course, based on which the output document call is generated.

Value: photograph and its metadata, written to record system using Image IO in the desired output layout.

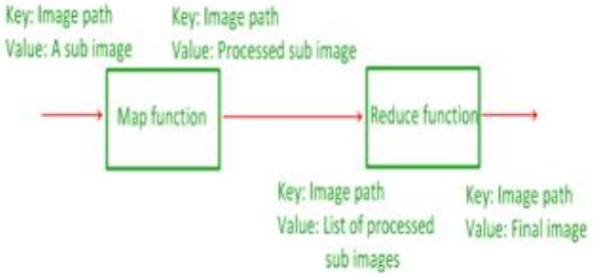

Fig.4: Map and Reduce function

The map characteristic is an implementation of the Sobel facet detection algorithm applied to the input sub picture. The lessen function uses the metadata associated with each picture split to combine them into the very last processed output image.

IV. EXPERIMENTAL RESULTS

Fig.5: Input image-1

Fig.6: Out Put Image-1

Fig.7: Input image-2

Fig.8: Out Put Image-2

V. CONCLUSION

This paper has defined our Hadoop Image Processing Framework for implementing big scale image processing packages. The framework is designed to abstract the technical details of Hadoop's powerful MapReduce device and provide an smooth mechanism for users to method huge image datasets. We provide software program equipment for storing pix in the numerous Hadoop record formats and efficaciously gaining access to the Map Reduce pipeline. By supplying interoperability between unique image facts kinds we allow the person to leverage many distinctive open-source image processing libraries. Finally, we provide the method to hold photograph headers in the course of photo manipulation manner, retaining useful and precious information for photo processing and vision packages. With those features, the framework affords a brand new degree of transparency and ease for developing huge-scale picture processing packages on pinnacle of Hadoop'sMapReduce framework. We display the energy and effectiveness of the framework in terms of performance enhancement and complexity discount. The Hadoop Image Processing Framework must significantly make bigger the population of software program builders and researchers easily able to create large-scale photograph processing programs.

VI. REFERENCES

[1]. J. Dean, S. Ghemawat, "Mapreduce: Simplified data processing on large clusters", Commun. ACM, vol. 51, no. 1, pp. 107-113, Jan. 2008.

[2]. S. Ghemawat, H. Gobioff, S.-T.Leung, "The google file system", SIGOPS Oper. Syst. Rev., vol. 37, no. 5, pp. 29-43, Oct. 2003.

[3]. J. Lin and C. Dyer, "Data-intensive text processing with mapreduce,"Synthesis Lectures on Human Language Technologies, vol. 3, no. 1, pp.1–177, 2010.

[4]. A. McKenna, M. Hanna, E. Banks, A. Sivachenko, K. Cibulskis,A. Kernytsky, K. Garimella, D. Altshuler, S. Gabriel, M. Daly, andM. A. DePristo, "The Genome Analysis Toolkit: A MapReduce frameworkfor analyzing next-generation DNA sequencing data," Genome Research, vol. 20, pp. 1297–1303, 2010.

[5]. K. Wiley, A. Connolly, J. Gardner, S. Krughoff, M. Balazinska,B. Howe, Y. Kwon, and Y. Bu, "Astronomy in the cloud: usingmapreduce for image co-addition," Astronomy, vol. 123, no. 901, pp.366–380, 2011.

[6]. S. Horaczek, "How many photos are uploaded to the internetevery minute?" http://www.popphoto.com/news/2013/05/how-manyphotos-are-uploaded-to-internet-every-minute,2013.

[7]. B. White, T. Yeh, J. Lin, and L. Davis, "Web-scale computer visionusing mapreduce for multimedia data mining," in Proceedings ofthe Tenth International Workshop on Multimedia Data Mining, ser.MDMKDD '10. New York, NY, USA: ACM, 2010, pp. 9:1–9:10.[Online]. Available: http://doi.acm.org/10.1145/1814245.1814254

[8]. R. Pereira, M. Azambuja, K. Breitman, and M. Endler, "An architecturefor distributed high performance video processing in the cloud," inCloud Computing (CLOUD), 2010 IEEE 3rd International Conferenceon, July 2010, pp. 482–489.

[9]. Z. Lv, Y. Hu, H. Zhong, J. Wu, B. Li, and H. Zhao,"Parallel k-means clustering of remote sensing images based onmapreduce," in Proceedings of the 2010 International Conference n Web Information Systems and Mining, ser. WISM'10. Berlin,Heidelberg: Springer-Verlag, 2010, pp. 162–170.[Online].Available:http://dl.acm.org/citation.cfm?id=1927661.1927687

[10]. C. Zhang, H. De Sterck, A. Aboulnaga, H. Djambazian, and R. Sladek,"Case study of scientific data processing on a cloud using hadoop," inHigh Performance Computing Systems and Applications, ser. LectureNotes in Computer Science, D. Mewhort, N. Cann, G. Slater, and T. Naughton, Eds. Springer Berlin Heidelberg, 2010, vol. 5976, pp.400–415.

Authors Profile

Dr.V.Ramaswamy obtained his Ph.D. degree from Madras University, in 1982. He is working as Professor and Head in the Department of Compute Science and Engineering, SASTRA University, Srinivasa Ramanujan Centre, Kumbakonam, Tamil Nadu. He has more the 30 years of teaching experience including his four years of service in Malaysia. He is guiding many research scholars and has published many papers in national and international conference and in many international journals and authored one book. He has visited many universities in USA and Malaysia. He is a life member of Indian Society for Technical Education, India. His research interest includes Fuzzy logic, Cryptography, Data Mining, Image Processing, Video Processing and Pattern Recognition.

Dr. Parashuram Baraki obtained his Ph.D in Computer Science & Engineering from Jain University Bangalore Karnataka, under the guidance of Dr.V.Ramaswamy and currently working as Professor and Head in the Department of Computer Science & Engineering, Hirasugar Institute of

Technology, Nidasoshi, Belagavi, Karnataka, India. He has more the 17 years of teaching experience and published many papers in national and international conference and in many international journals. He is a life member of Indian Society for Technical Education, India. His research interest includes Image Processing, Video Processing and Pattern Recognition.

Mr. Fairoz B obtained his Masters Degree M.Tech in Computer Science & Engineering from Visvesvaraya Technological University, Belgaum , Karnataka and currently working as Technical Head in the Product Design and Development Team, Vidvath Technologies Private Limited, Davangere, Karnataka, India . He has more the 3 years of IT Industry experience and have worked on many real time projects and well research projects. He is a Udemy Certified Android Application Developer. His research interest includes Machine Learning, Big Data and Hadoop, Artificial Intelligence , Image Processing and Video Processing.