International Journal of Communication and Information Technology

E-ISSN: 2707-6628
P-ISSN: 2707-661X
www.computersciencejournals.com/ijcit
IJCIT 2021; 2(1): 19-29
Received: 17-11-2020
Accepted: 21-12-2020

Zachary Bosire Omariba
Department of Computer Science, Egerton University, Kenya

Corresponding Author:
Zachary Bosire Omariba
Department of Computer Science, Egerton University Kenya

A survey on big data processing: A case of jpeg 2000-image compression

Zachary Bosire Omariba

DOI: https://doi.org/10.33545/2707661X.2021.v2.i1a.25

Abstract
Big Data is actually the current big thing in computing as the quantity of computing data generated in the globe is growing exponentially from dissimilar areas for diverse causes. This survey presents firstly an overview of Big Data history, paradigms, and characterizations. Secondly, the technologies associated with Big Data processing. A case study of JPEG-2000 image compression is presented through simulations to underscore the importance of data compression in big-data processing. Finally the challenges faced in Big Data processing forms the basis of future research.

Keywords: big-data, data processing, jpeg-2000, image compression

Introduction
The rapid development of Big Data remains an elephant in the house that has attracted extensive attention from governments, academia, and industry in the whole world. Big Data is actually the current big thing in computing as the quantity of computing data generated in the globe is growing exponentially from dissimilar areas for diverse causes. Big Data is defined by its size, comprising a large, complex and independent collection of data sets, each with the potential to interact [1, 2]. Big Data, broadly defined, comprises information available from billions, even trillions of records generated by millions of people and stored in myriad sources throughout the cyber universe [3], and from the macro [4] Big Data is the bond that subtly connects and integrates the physical world, human society, and the cyberspace. The term "Big Data" often refers simply to the use of predictive analytics, user behavior analytics, or certain other advanced data analytics methods that extract value from data, and seldom to a particular size of data set. There is little doubt that the quantities of data now available are indeed large, but that's not the most relevant characteristic of this new data ecosystem. More generally, a data set can be called Big Data if it is formidable to perform capture, curation, analysis, and visualization on it at the current technologies [5]. For the purpose of this paper we define Big Data as exceptionally enormous data sets that may be analyzed computationally to disclose patterns, trends, and associations, especially relating to human-computer actions, interactions, and relations.
A study on the Evolution of Big Data as a Research and Scientific Topic shows that the term "Big Data" was present in research starting with the 1970s. The term Big Data has been around since 2005 when it was launched officially by O'Reilly Media in 2005. However, the usage of Big Data and the need to understand all available data has been around much longer.

Characterization of Big Data
There are variant sources of Big Data that exist today. Actually, data exist in large amounts in all walks of life. The main sources of Big Data are data information within the organization, sensory information in the Internet of things and interactive information in the Internet world [6]. These sources can be ranging from databases, blogs, email, search engines, digital music, social media sites, smart grids, industrial equipment, satellites, data warehouse applications, train tracks, and smartphone location. Big Data has been defined differently from 3V to 4V, 5V, and 6V by various researchers. For this study, Big Data can be described by the following characteristics 6V+C [1, 4, 6, 7, 8, 9].

Volume: Volumes of data have exponentially increased in recent times. The data volume is huge, especially the huge amounts of data generated by a variety of devices, the size of the

data is very large, much larger than the flow of information on the Internet, PB level will be the norm. It is not uncommon for businesses to deal with yotta-bytes of data and typically analysis is performed over the entire data set. This implies the need for large storage space and computing power requirement during processing.

Velocity: Big Data is not just about volume though; just as important is the rate of change of the data. The data related to perception, transmission, decision making, control open loop have very high requirements on real-time data processing. The late result is of little value. It is essentially different from traditional data processing. For a large volume of data which does not change very often, the analysis takes a number of hours or days to complete may be acceptable. But if the dataset is growing by terra-bytes per day, or the data is changing at a high rate of speed, the processing time of analysis becomes much more important. The velocity, therefore, deals with the place at which raw data streams in the database in various forms from various sources like the mobile devices, internet of things (IoT), and social networks [2].

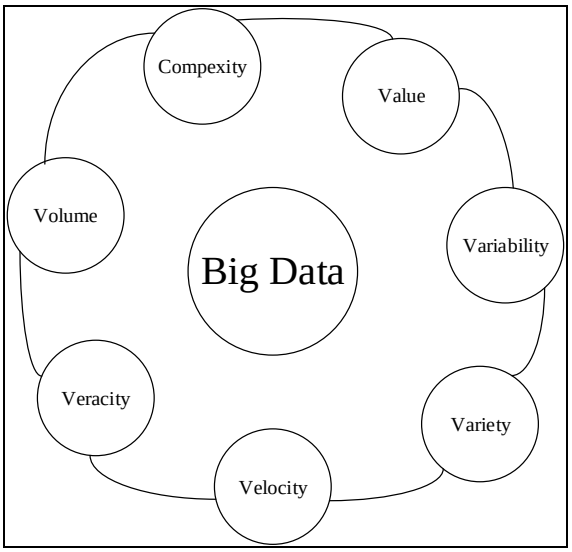


Fig 1: Big Data Characterization

Variety: Refers to the many different data and file types that are important to manage and analyze more thoroughly, but for which traditional relational databases are poorly suited [10]. Big Data may be collected via smartphones, sensors, or even social networks and comes in the form of text, audio, blogs, videos, pictures, images, or location information, which is not easy to put Big Data into a relational database [11]. Even with the same kind of data, encoding, data format or other aspects may be different. Thus Big Data includes structured, semi-structured, unstructured, and a blend of both.

Variability: Variability refers to the variation in the data flow rates [12], which keeps changing dramatically over time [13], and it is used to determine the internal variability in Big Data with multiple information shifts as time passes [14][15]. As data is generated from a myriad of sources, a challenge occurs on how to connect, match, cleanse, and transform data received from diverse sources, hence the variability of the Bid Data.

Veracity: When dealing with high volume, veracity, and variety of data, it is inevitable that at all that the data is going to be 100% error free, as there will be unimportant or dirty data. This means that in any data that is streamed there exists some biases, noise, and abnormality in the data that needs to be cleaned. However, determining how clean the data is for analysis to be performed is the big question. Basically, data does not need to be 100% perfect but need to be relatively close enough to enable one to obtain gainful and relevant insight. Thus verification or veracity of the data beyond a reasonable doubt is essentially dependent on the application.

Value: Big Data is having a high commercial value. It is the reason why more and more people are concerned about Big Data. But the density of the value is low. If you take the video for example, during the uninterrupted monitoring process, useful data may only be a few seconds. Implementing information technology infrastructure system to store Big Data is a costly venture and businesses are going to get a return on investment (ROI). Due to this value must be extracted from the data as it is pointless to build the capability to store and manage it. Data analysis to provide ROI is thus required to build a competitive advantage.

Complex: Big Data is not only complex but diverse in data types and representation. There is often a complex dynamic relationship between each data. The change of one data may have an impact on a lot of data. Dealing with a variety of structured and unstructured data greatly increases the complexity of both storing and analyzing Big Data.
However many researchers have defined the characteristics of Big Data differently by increasing the characteristics to 6Vs: volume, velocity, variety, variability, veracity and value [16]. Thus the term Big Data can be described differently but the characteristics are growing as time goes by.

Big Data Processing Technologies

Big Data refers to data streams of higher velocity and higher variety, the infrastructure required to support the acquisition of Big Data must deliver low, predictable latency in both capturing data and in executing short, simple queries; be able to handle very high transaction volumes, often in a distributed environment; and support flexible, dynamic data structures [3]. To take advantage of the cloud for Big Data analysis, data must first be in the cloud. To address the issue of uploading Big Data to the cloud, a number of approaches to WAN optimization have been established. The techniques include compression, data de-duplication or redundancy elimination, caching, and protocol optimization [17].

Table 1: Overview of big-data processing technologies

Technology	Description	Advantage	Limitations	References
Compression	Process of modifying, encoding or converting the bits structure of data in such a way that it consumes less space on disk	▪ Reduces storage size, ▪ Speeds up file transfer, ▪ Decreases costs of storage hardware, ▪ Decreases cost of network bandwidth. ▪ To get rid of unnecessary data ▪ Data compression is byte order independent ▪ Compressed data is read/written faster than original data	▪ Is a mathematically intense process ▪ Effects of errors in transmission ▪ Slower for sophisticated methods ▪ Need to decompress all previous data ▪ Takes time to compress ▪ Data compression process takes up valuable resources ▪ High Computational Latency ▪ Can't search compressed files ▪ Lossy algorithms	[18]
Data Redundancy		▪ Alternative data backup method ▪ Better data security ▪ Faster data access and updates ▪ Improved data reliability	▪ Possible data inconsistency ▪ Increase in data corruption ▪ Increase in database size ▪ Increase in cost	[19] [20]
Data Caching	Caching is a technique of storing frequently used data/information in memory, so that, when the same data or information is needed next time, it could be directly retrieved from the memory instead of being generated by the application	▪ Reduced execution time, ▪ Reduced run-time, ▪ Improved performance of small file transfers, ▪ Efficient query matching ▪ Security ▪ Reduced interaction with the under layer	▪ Picking the right cache size, ▪ Inconsistency may occur ▪ Threshold for use is high ▪ Stale data ▪ Complexity ▪ Expiration policy, ▪ And eviction policy	[21] [22][23] [24]
Protocol Optimization	Is using data science and computer algorithms to discover inefficiencies that would be almost impossible to discover with traditional research methods.	▪ Improves access efficiency ▪ Sets optimal parallelism and concurrency levels. ▪ Increase in throughput ▪ Reduces routing costs	▪ Optimization convergence problem ▪ Numerous decision variables	[25] [26][27][28] [29]

Data Redundancy Elimination

Data redundancy refers to a state created within a database data storage technology in which the same piece of data is held in two separate places. This often leads to many data anomalies and should be avoided through design by applying database normalization, proper use of foreign keys, and through the application of compression and decompression schemes. This goes a long way to prevent, reduce or minimize data redundancy, like unnecessary increase in the size of database and likelihood of data corruption, the likelihood of inconsistency of data and decreased efficiency of the database. The processes of getting candidate similar pairs, eliminating redundancy, and similarity computation are all finished within a MapReduce job which is employed on a filter-and-verification framework. In the first step, it generates signatures for all objects. If two objects are similar, they must share one or more common signatures. Based on this idea, it prunes large numbers of dissimilar pairs and the survived pairs are candidate pairs. In the verification step, it verifies the candidate pairs to generate the final answers [19]. In the map phase, the signature function will be applied on each object, and the map function will emit <key, value> pairs. In reduce phase, redundancy elimination strategy will be applied on each set where all objects share the same key or signature, and the reduce function will emit <key, value> pairs that are similar and can be considered as the same entity [19].

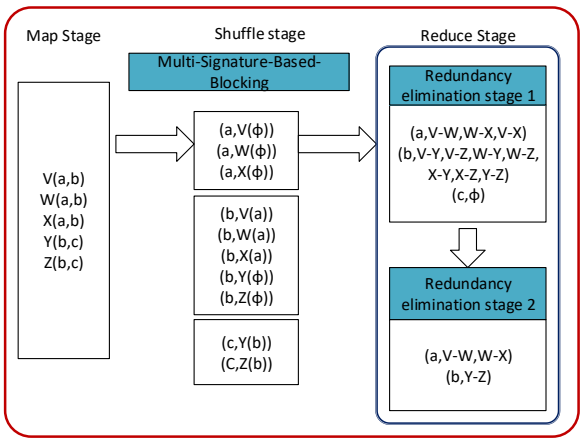


Fig 2: An example of redundancy elimination

Fig. 2 shows an example of the multi-sig-er method. From this method, we can find the processes of eliminating redundancy and similarity joining are done away with in the reduce phase. Blocking technique and eliminating redundancy strategy aim to reduce the number of candidate similar pairs and improve computing efficiency [19].

Data redundancy can be reduced through the process of data normalization. Database normalization is the process of efficiently organizing data in a database so that redundant data is eliminated [20]. This process can ensure that all of a company's data looks and reads similarly across all records.

By implementing data normalization, an organization standardizes data fields such as customer names, addresses, and phone numbers. Normalizing data involves organizing the columns and tables of a database to make sure their dependencies are enforced correctly. The "normal form" refers to the set of rules or normalizing data, and a database is known as "normalized" if it's free of delete, update, and insert anomalies. When it comes to normalizing data, each company has their own unique set of criteria. Therefore, what one organization believes to be "normal," may not be "normal" for another organization. For instance, one company may want to normalize the state or province field with two digits, while another may prefer the full name. Regardless, database normalization can be the key to reducing data redundancy across any company.

Data Caching

The data caching in computing is referred as the implicit of explicit practice of storing data and retrieving data from high performance store. A cache in this case is the hardware, or software component that serves requests for the stored data with high speed. The stored data in the cache is as a result of the data stored in the cache due to earlier computation or data stored elsewhere.

Caching leads to a reduction in time which is achieved by caching the working datasets in a cluster's memory while, eliminating the need of repeated read/write to/from a disk [21], improves performance of small file transfers [22][23], and ensures efficient query matching [24]. There are two instance of the cache: cache hit and cache miss. The cache hit occurs when the requested data is found in the cache, but if the requested data is not found in the cache that forms the cache miss. The more the cache hit rate, the higher the performance of the system, as the reading of the data from the cache is faster than reading the data from a slower data source. The cache hit rate or hit ratio is the percentage of accesses that result in cache hits. The big data caching framework is shown in Figure 3. The mobile device can save or fetch data to and from the data store or local data base, and also can load data to the server side application. These mobile devices can be laptops, personal digital assistants (PDAs), iPads, or smart phones. The server side application can also load data from the cloud store and the data is stored in the cache for processing.

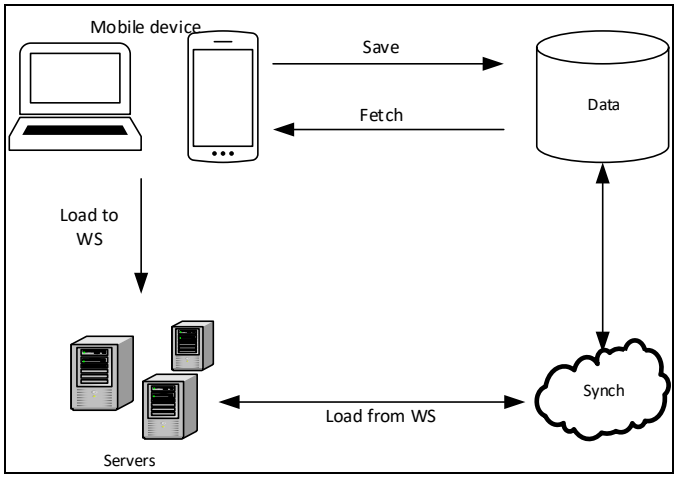


Fig 3: Big data caching framework

Protocol Optimization

There are various optimization protocols proposed in literature that are either TCP/IP-based, or UDP-based. For efficient transferring of big data over long distances, the use of appropriate transfer protocols is required. Current innovations aimed at tackling high-speed massive flow challenge in long fat networks include: Grid FTP, Globus, Tsunami, FP, and UDT [25]. For protocol optimization the distributed consensus algorithm can be run to select a leader to reduce the number of broadcasts, while allowing computation between encrypted data and non-encrypted data [26]. Protocol optimization improves access efficiency by utilizing prefetching, and caching technologies.

Compression

Compression of data is the shrinking of the volume of available data into technical (compressing file formats), or functional (shrinking data volume using econometric tools), so as to enable easier and faster processing of data signals above the minimum threshold [18]. In computer signal processing, data compression is referred to as bit-rate reduction, which involves the encoding of information using fewer bits than the original representation. Data compression should be done in a way that it mitigates the cost and time needed during analysis to decision making with minimal loss of information of minimal error. There are two types of data compression (lossy-compression, and lossless-compression). The lossy data compression involves the class of data encoding methods and standards that uses inexact approximations and data discarding to represent the content. This method of compression is used to reduce the size of data for storing, transmission, and or handling. Lossy-compression method is commonly used in video and audio compression, where a certain loss of information cannot be detected by most users. Sometimes when the data is being cleaned, there can be the elimination of noise, and this noise cannot be easily detected, which means that some data will be lost if the reversal process takes place. On the other hand, lossless data compression refers to the class of data compression methods, algorithms, and standards that allows the original data to be perfectly reconstructed from the compressed data. This means that in lossless compression we do not experience the loss of data after the reverse process is invoked. This method basically involves the reconstruction of data to its original format or set after

decompression is done as data is not lost.

Data are compressed because the raw (uncompressed file) data require much storage space, large transmission bandwidths, and long transmission times. With the present state of technology, the only solution is to compress data before their handling, storage and transmission. Then, at the receiver end, the compressed/coded data can be decompressed/decoded back to the original format.

Lossy-and-lossless-Image compression

There are basically two types of compression namely the lossy compression, and lossless compression. Some standards or algorithms offers both methods of compression combined while others offers only one form of compression method.

Case study: JPEG 2000- Image compression

In 2000 the (Joint Photographic Experts Group) committed in order to create a new compression technique that replaces their previous one (JPEG) the widely adopted by applications as a coding standard [30].

Description of JPEG 2000

The JPEG2000 standard allows for both lossless and lossy compression. But the lossy compression is more commonly used. Briefly, The JPEG2000 compression standard is composed of the stages shown in the flow graph in Figure 3.

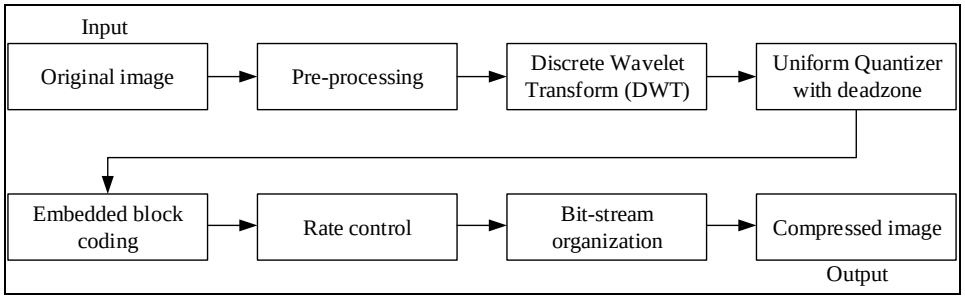


Fig 4: The JPEG2000 Compression Algorithm

Pre-processing

In the first stage, pre-processing is performed. Pre-processing actually contains three sub-stages, as shown in Figure 4. These steps must be performed so that the discrete wavelet transform can be properly performed.

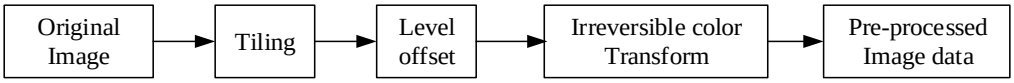


Fig 5: Sub-stages of pre-processing

The image to be encoded might be larger than the amount of memory available to the encoder. To solve this problem, JPEG2000 allows for optional tiling [31][32]. Tiling is required as it handles large datasets since it reduces the needs imposed on the processing platform (e.g. memory requirement, transmission bandwidth, and transmission time) [31][33]. In tiling, the input image is partitioned into rectangular and non-overlapping tiles of equal size (except possibly for those tiles at the image borders), as shown in Figure 5. Each tile is compressed independently using its own set of specified compression parameters.

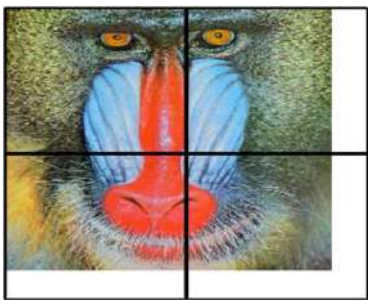


Fig 6: Example tiling of the 8-bit image

JPEG2000 expects its input sample data to have a nominal dynamic range centred about zero. This expectation is necessary since JPEG2000 uses high-pass filtering. The level offset pre-processing stage ensures that this expectation is met. If the original *B*-bit image sample values are unsigned (non-negative) quantities, an offset of -2^{B-1} is added so that the samples have a signed representation in the range $-2^{B-1} \leq x[n] < 2B-1$. If the data is already signed (centred around zero), no adjustment is performed.

JPEG2000 compression is most commonly used to compress colour images. There are several ways to represent colour images numerically, for example, RGB, YC*b*C*r*, YCM, and HSB. However, colour images are most commonly represented in RGB format. In RGB format, the image is composed of three component planes, one each for the red, green, and blue colour components. When the discrete wavelet transformation is performed in JPEG2000, each colour layer is transformed independently. However, since Y, C*r* and C*b* colour components are less statistically dependent than R, G, and B colour components, they independently compress better. Therefore, in JPEG2000 an irreversible colour transform (ICT) is performed to convert RGB data into YC*r*C*b* data according to the following:

$$\begin{bmatrix} Y \\ Cr \\ Cb \end{bmatrix} = \begin{bmatrix} 0.300 & 0.587 & 0.115 \\ -0.170 & -0.332 & 0.500 \\ 0.500 & -0.420 & -0.081 \end{bmatrix} = \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

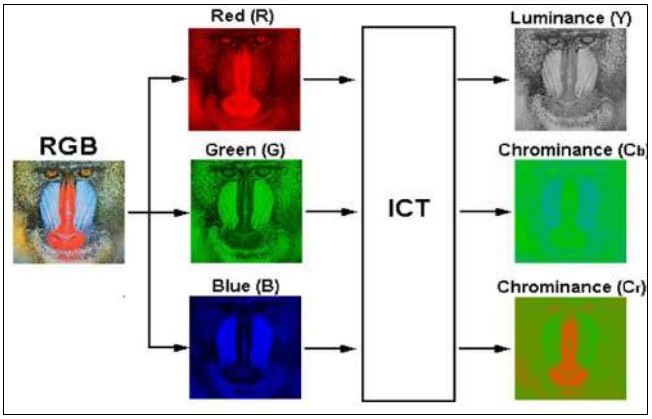


Fig 7: The irreversible colour transform (ICT) of an image.

Discrete Wavelet Transformation

JPEG2000 uses discrete wavelet decomposition (DWT) to decompose each image tile into its high and low sub-bands. The DWT is performed by filtering each row and column of the pre-processed image tile with a high-pass and low-pass filter [32][34]. Because this process results in double the number of samples, the output from each filter is down sampled by 2 (every other value is removed) so that the sample rate remains constant. Also, it does not matter if the rows or the columns of the component matrix are filtered first. The resulting DWT is the same.

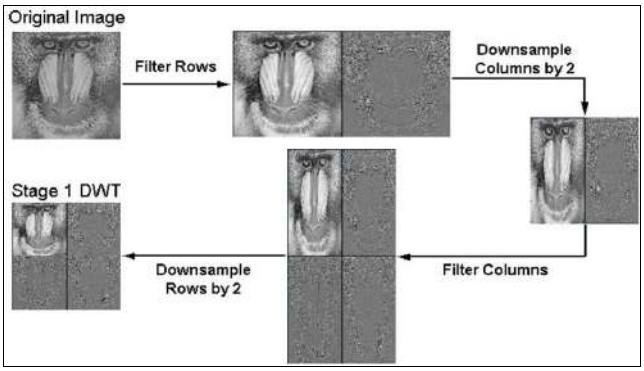


Fig 8: The discrete wavelet decomposition DWT process for the Y component of the baboon image.

In JPEG2000, multiple stages of the DWT are performed. The number of stages performed is implementation dependent [32].

Figure 6 shows the Stage 1 DWT for the Y component of the original pre-processed image tile. The four quadrants are defined as follows:

LL: low sub-bands for row and column filtering
HL: high sub-bands for row filtering and low sub-bands for column filtering
LH: low sub-bands for row filtering and high sub-bands for columns filtering
HH: high sub-bands for row and column filtering

In stage 2 (Figure 9), the same process is repeated with the LL1 sub-bands. Only the LL sub-bands is further transformed because the high sub-bands rarely contain any significant samples. Finally, in Stage 3 the DWT is repeated a third time. Again, only the LL2 sub-bands from Stage 2 is further filtered, as Figure 10 shows. JPEG2000 supports from 0 to 32 stages. For natural images, usually between 4 to 8 stages are used.

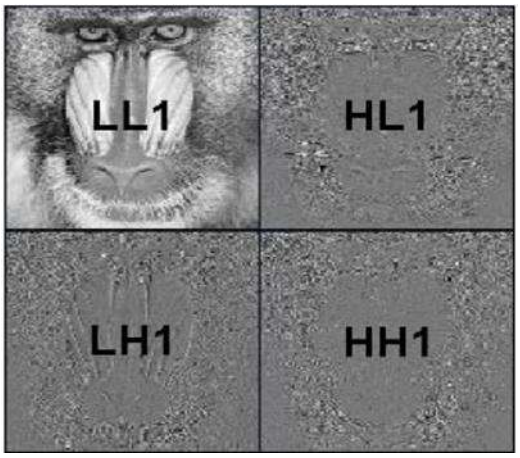


Fig 9: Stage 1 DWT of the 8-bit image tile

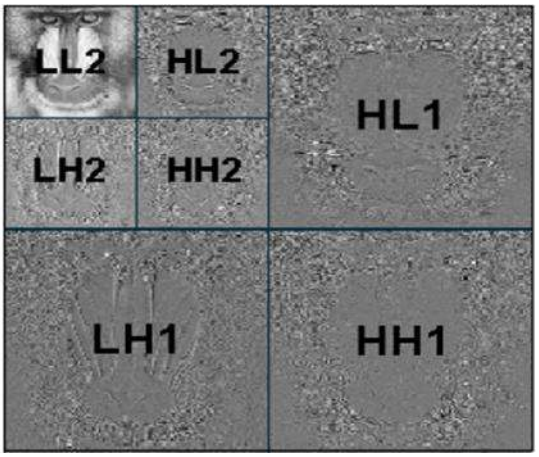


Fig 10: Stage 2 DWT of the 8-bit image tile

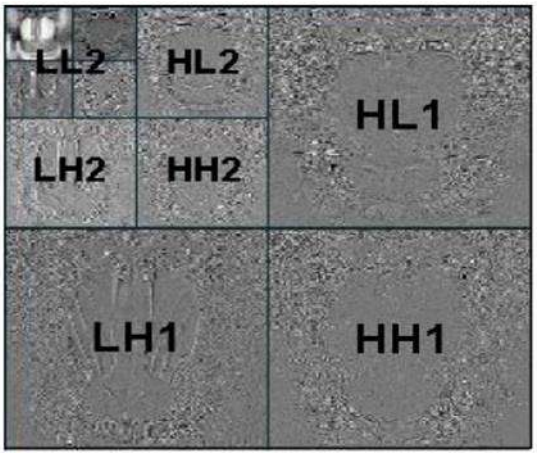


Fig 11: Stage 3 DWT of the 8-bit image tile

Quantization

The quantization is scalar and can be of three types: single (SQ), multiple (MQ) and bi-level (BQ) [35]. With SQ each wavelet coefficient is quantized once, the produced bit-stream not being signal-to-noise-ratio (SNR) scalable. With MQ a coarse quantizer is used and this information coded. A finer quantize is then applied to the resulting quantization error and the new information coded. This process can be repeated several times, resulting in limited SNR scalability. BQ is essentially like SQ, but the information is sent by bit-planes, providing general SNR scalability. The wavelet coefficients are quantized using a uniform quantizer with dead-zone as shown in Figure 11. For each sub-bands b, a

basic quantizer step size _b is used to quantize all the coefficients in that sub-bands according to:

$$q = sing(y)\left\lfloor \frac{|y|}{\Delta_b} \right\rfloor \quad (2)$$

Where *y* is the input to the quantizer, sign(*y*) denotes the sign of *y*, _b is the step size, and *q* is the resulting quantizer index. Dead-zone means that the quantization range about 0 is 2_*b*. This ensures that more zeros result.

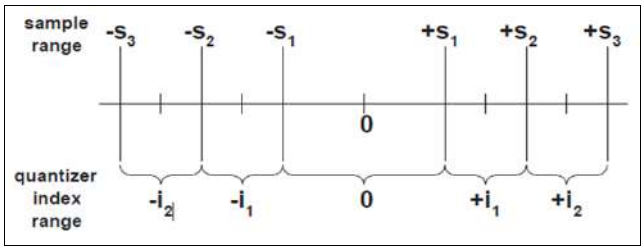


Fig 12: Dead-zone quantizer structure

Embedded Block Coding

In JPEG2000, before coding is performed, the sub-bands of each tile are further partitioned into relatively small code-blocks (e.g. 64*x*64 or 32*x*32 samples) such that code blocks from a sub-bands have the same size [30][34]. Code-blocks are used to permit a flexible bit-stream organization.

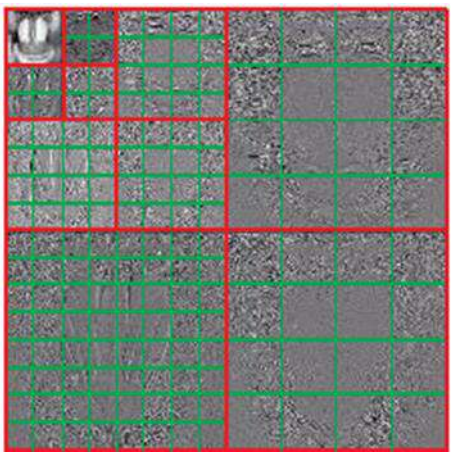


Fig 13: Example division of sub-bands into code-blocks

Code-blocks are then coded a bit-plane at a time starting from the Most Significant Bit-Plane to the Least Significant Bit-Plane (if some MSB-planes contain no 1s, the MSB-plane is set to the topmost bit-plane, with at least one 1, the number of bit-planes which are skipped is then encoded in a header.)

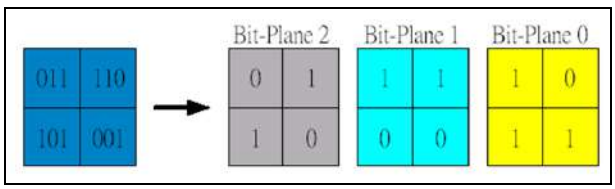


Fig 14: Example of bit plane representation

Each coefficient bit in the bit-plane is coded in only one of the

Three Coding Passes

1. Significance Propagation: If a bit is insignificant (=0) but at least one of its eight neighbors is significant (=1), then it is encoded. If the bit at the same time is a 1, its significance flag is set to 1 and the sign of the symbol is encoded.
2. Magnitude Refinement: Samples which are significant and were not coded in the significance propagation pass.
3. Clean-up: It codes all bits which were passed over by the previous two coding passes insignificant bits. It is the first pass for MSB plane.

At the same time as embedded block coding is being performed, the resulting bit-streams for each code-block are organized into quality layers. A quality layer is a collection of some consecutive bit-plane coding passes from all code-blocks in all sub-bands and all components, or simply stated, from each tile. Each code-block can contribute an arbitrary number of bit-plane coding passes to a layer, but not all coding passes must be assigned to a quality layer. Every additional layer successively increases image quality. Figure 14 shows an example of how the encoded data for each bit-plane in each code block can be organized into quality layers. Note the relationship between the amount of important information in a code-block and how many quality layers are present in that code-block.

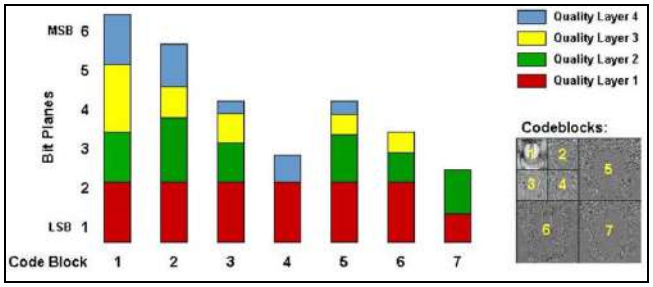


Fig 15: Example quality layer distribution for stage 2 DWT with sub-bands containing only one code-block

Rate Control

Rate control is the process by which the code-stream is altered so that a target bit rate can be reached. Once the entire image has been compressed, a post-processing operation passes over all the compressed blocks and determines the extent to which each block's embedded bit-stream should be truncated in order to achieve the target bit rate. The ideal truncation strategy is one that minimizes distortion while still reaching the target bit-rate. Each code-block, *Bi*, contains a finite number of truncation points, *Zi* + 1, having lengths $L_i^{(z_i)}$ such that

$$0 = L_i^{(0)} \le L_i^{(1)} \le L_i^{(2)} \le \cdots \le L_i^{(z_i)} \quad (3)$$

The overall reconstructed image distortion can be represented as a sum of the distortion contributions from each of the code-blocks, where *D*(*z*)*i* is the distortion contributed by a code-block. Since the code blocks are compressed independently, any bit-stream truncation policy can be used. If the overall length of the final compressed bit-stream is constrained by *Lmax* then any set of truncation points can be selected such that

$$\sum_i L_i^{(z_i)} \leq l_{max} \quad (4)$$

And the overall distortion is minimized

$$\sum_i D_i^{(z_i)} \quad (5)$$

- Which code-blocks are included in the packet
- The number of most significant all zero bit-planes skipped by the entropy encoder for each newly included code-block
- The number of included coding passes for each code-block
- The length of included coded data for each code-block, potentially zero.

In bit-stream organization, the compressed data from the bit-plane coding passes are first separated into packets. One packet is generated for each precinct in a tile. A precinct is essentially a grouping of code blocks within a resolution level [34]. Precincts divide a resolution level of a component into rectangles of size (*Px*, *Py*) containing 2*Px*, 2*Py* samples. Si precincts cannot overlap code-blocks and must have dimensions that are exact powers of 2, the precinct size restricts the subordinate code-block partitions. An example of precinct partitioning is shown in Figure 14.

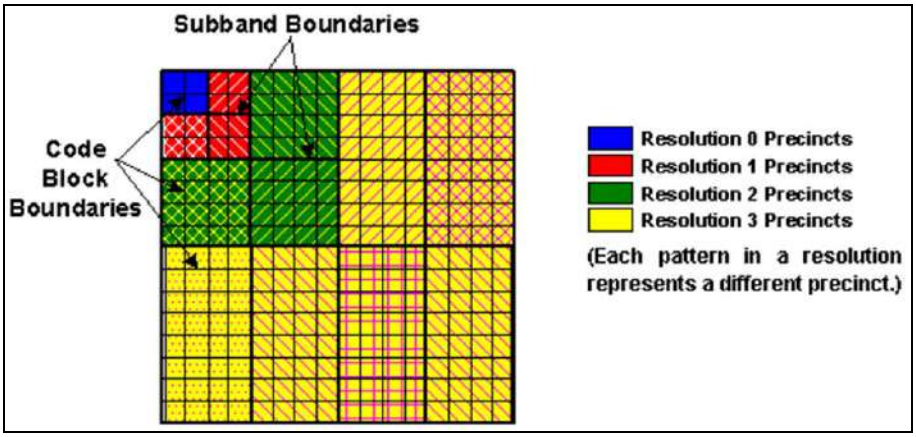


Fig 16: Example precinct partitioning

As previously stated, each precinct generates one packet, even if the packet is empty. A packet is composed of a header and the compressed data. The header contains:
The organization of the code-stream can be seen in Figure 16
Then, the packets are multiplexed together in an ordered manner to form one code-stream. In Part 1, there are five built-in ways to order the packets, called progressions, where position refers to the precinct number:
Quality: layer, resolution, component, position
Resolution 1: resolution, layer, component, position
Resolution 2: resolution, position, component, layer
Position: position, component, resolution, layer
Component: component, position, resolution, layer.

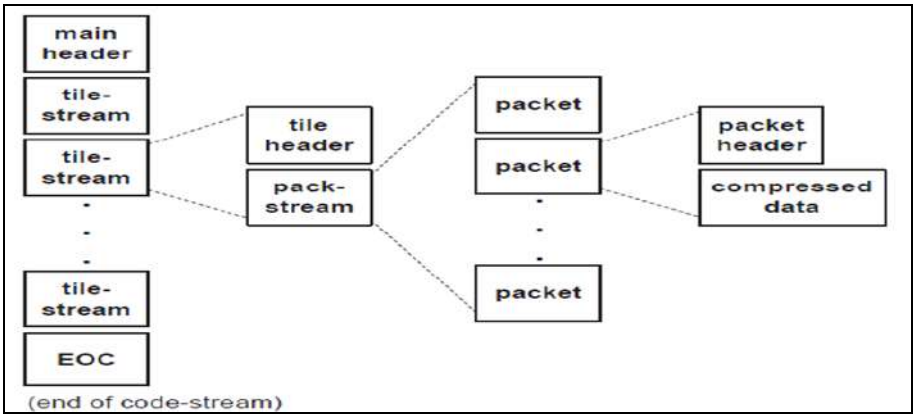


Fig 17: Code-stream organization

The sorting mechanisms are ordered from most significant to least significant. For example, in the case of quality progression, packets are ordered first by layer, second by resolution, third by component, and fourth by position [35]. It is also possible for the progression order to change arbitrarily in the code-stream. Additionally, in Part 2 of the standard, it is possible to specify user-defined progressions at the expense of additional overhead.
This new standard by the JPEG2000 working group was hoped to create a standard which would address the faults of current standards (JPEG):
Superior compression performance: At high bit rates, artifacts become nearly imperceptible, JPEG 2000 has a small machine-measured fidelity advantage over JPEG. At lower bit rates (e.g., less than 0.25 bits/pixel for grayscale images), JPEG 2000 has a significant advantage over certain modes of JPEG: artifacts are less visible and there is almost no blocking [31]. The compression gains over JPEG are

attributed to the use of DWT and a more sophisticated entropy encoding scheme.

Multiple resolution representation: JPEG 2000 decomposes the image into a multiple resolution representation in the course of its compression process. This representation can be put to use for other image presentation purposes beyond compression as such.

Progressive transmission by pixel and resolution accuracy: These features are more commonly known as *progressive decoding* and *signal-to-noise ratio (SNR) scalability*. JPEG 2000 provides efficient code-stream organizations which are progressive by pixel accuracy and by image resolution (or by image size) [35]. This way, after a smaller part of the whole file, has been received, the viewer can see a lower quality version of the final picture. The quality then improves progressively by downloading more data bits from the source.

Choice of lossless or lossy compression: Like the Lossless JPEG standard, the JPEG 2000 standard provides both lossless and lossy compression in a single compression architecture. Lossless compression is provided by the use of reversible integer wavelet transform in JPEG 2000.

Error resilience: Like JPEG 1992, JPEG 2000 is robust to bit errors introduced by noisy communication channels, due to the coding of data in relatively small independent blocks.

Flexible file format: The JP2 and JPX file formats allow for handling of colour-space information, metadata, and for interactivity in networked applications as developed in the JPEG Part 9 JPIP protocol.

High dynamic range support: JPEG 2000 supports any bit depth, such as 16- and 32-bit floating point pixel images, and any colour space.

Side channel spatial information: Full support for transparency and alpha planes.

Simulation results

Table 2: Simulation parameters

Parameter	Value
Programming Language	Java (Jdk1.8)
Library	Java Advanced Imaging (JAI)
IDE	Netbeans
Environment	Windows10 i74510u Intel CPU
Output Extension	Jp2
Compression Type	Lossy
Type Of Progression	Layer
Encoding Rate	1.01 (Bit Per Pixel)
Wavelet Filters	J2kimagewriteparam.Filter_97

First experiment

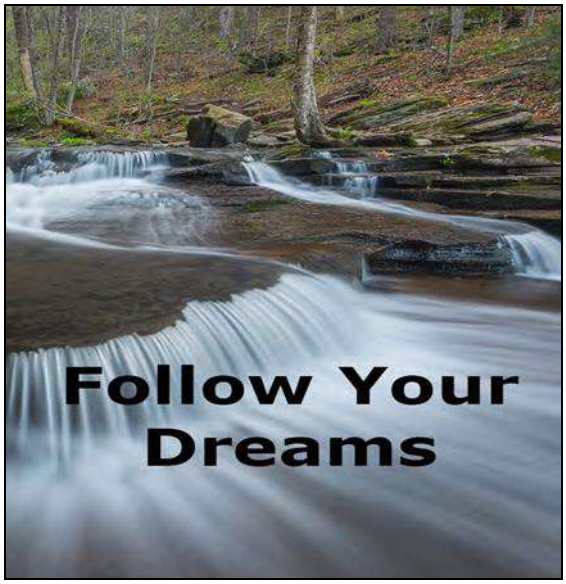
We compress different files of different formats and compare the resulting file's size with the size after jepg2000 compressing as shown in Table 3.

Table 3: Experiment 1 results

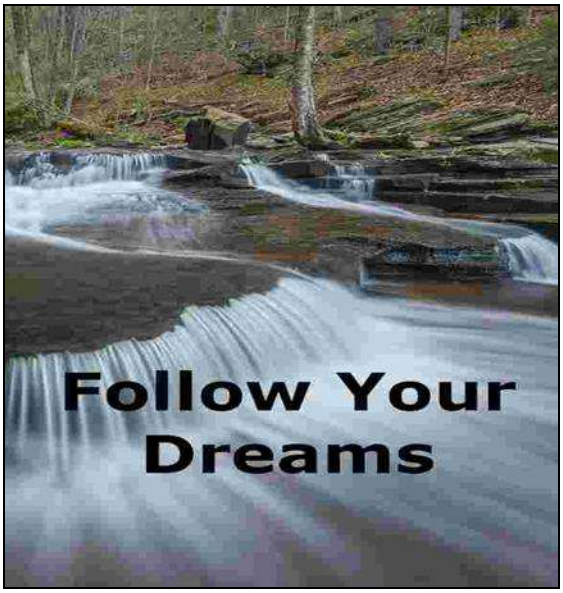
Orignal Format	Orignal size	Size after JPEG2000 compression	Compression ratio
Png	127 KB	78,6 KB	1,6
JPEG	15,9 MB	2,88 MB	5,52
TIFF	7,82 MB	1,53 MB	5,1

Second experiment

We compressed the same file (.png) in JPEG and JPEG2000, we set the compression quality to (0.1) which is good according to java library and we observe the quality, Figure 17 shows the difference in quality.



(a) JPEG2000



(b) JPEG

Fig 18: The difference in quality (JPEG2000, JPEG)

Challenges of Big Data Management and Analytics

There is an explosive growth trend on the amount of data that is processed, transferable, or stored within a given time period [37]. The conventional data processing systems and protocols are likely to reach a tailback, unable to complete operational tasks in a timely manner. As more and more fields involve Big Data problems, ranging from global economy to social administration and from scientific researches to national security, we have entered the era of Big Data [5]. Data processing has encountered various open research issues namely data staging, data analysis, data security and distributed storage system [38]. With the emergence of new technologies for Big Data, a few attempts of data processing can be found adapted to them as noted by [7]. The challenges of Big-Data processing revolves around privacy, visualization, integration, and data-mining [36], and other challenges include data inconsistence and

incompleteness, scalability, and timeliness.

Security and privacy concepts in big data are some of the most pertinent issues as they have become increasingly significant associated with big data in all application domains in the modern world [39]. To improve the effective and growing big data uptake must be taken seriously by every organization to prevent breaches or other security incidences of over sensitive information. Data privacy should thus entail data security, access control, and information security. Clearly, privacy is an issue whose importance particularly to customers, is growing as the value of Big Data becomes more apparent. However, people do still care about what and how their personal information is used, especially if it could become disadvantageous or harmful to them [40]. Wherever, big data problems are handled, difficulties lie in data capture, storage, searching, sharing, analysis, and visualization. Purposely data visualization entails to represent knowledge more intuitively and effectively by using different graphs. To convey information easily by providing knowledge hidden in the complex and large-scale data sets, both aesthetic form and functionality are necessary [5]. But it is particularly difficult to conduct data visualization because of the large size and high dimensionality of big data.

To provide a single view of data distributed over different sources, data integration combines data views [15], fundamentally to get people collaborate and share data. However there is no clarity on the complexity of data integration and means of integration. Typically Big data integration is a process of collecting data from multiple sources and storing using different technologies to provide a unified view. Data can be in different forms, such as structured, unstructured and semi-structured. Integrating different types of data has become a complex task in case of merging different systems or different applications. Many challenges are associated with data integration, such as enabling real-time data access, increasing performance and scalability, and overlapping of the same data, which attracts further research.

Conclusion

Big data is the common phenomena in the recent past as data is growing in big magnitudes by day. This growth of data means that the data requires to be compressed in a way to occupy smaller space in memory and during transmission. This survey presented the big-data processing from the perspective of technologies involved and presented a case study of JPEG-2000 compression algorithm and compared it with JPEG. The simulation results show that JPEG-2000 performs better than JPEG and other compression standards. This calls for other advanced compression algorithms to process big data as data is kept growing. Future research will involve big data security, visualization and scalability.

References

1. Elena Geanina MV. ULARU, Florina Camelia PUICAN, Anca APOSTU, Perspectives on Big Data and Big Data Analytics, Database Syst. J., vol. III, no. 2012;4:1-79.
2. Inoubli W, Aridhi S, Mezni H, Maddouri M. An experimental survey on big data frameworks, Futur. Gener. Comput. Syst 2018;86:546-564. doi: 10.1016/j.future.2018.04.032.
3. Sowmya SKRR. Data Mining with Big Data, in 2017 11th International Conference on Intelligent Systems and Control (ISCO) 2017, 246-250, doi: 10.1109/TKDE.2013.109.
4. Jin X, Wah BW, Cheng X, Wang Y. Significance and Challenges of Big Data Research, Big Data Res 2015;2:59-64, doi: 10.1016/j.bdr.2015.01.006.
5. Chen CLP, Zhang C. Data-intensive applications, challenges, techniques and technologies : A survey on Big Data, Inf. Sci. (Ny) 2014;275:314-347, doi: 10.1016/j.ins.2014.01.015.
6. Zhang X, Xu F. Survey of research on big data storage, Proc. - 12th Int. Symp. Distrib. Comput. Appl. to Business, Eng. Sci. DCABES 2013, 76-80, doi: 10.1109/DCABES.2013.21.
7. García S, Ramírez-gallego S, Luengo J, Benítez JM, Herrera F. Big data preprocessing : methods and prospects 2016, 1-22, doi: 10.1186/s41044-016-0014-0.
8. Colombo P, Ferrari E. Privacy Aware Access Control for Big Data : A Research Roadmap, Big Data Res 2015;2(4):145-154, doi: 10.1016/j.bdr.2015.08.001.
9. Russom P. Big Data Analytics, 2011. [Online]. Available: tdwi.org.
10. Lovalekar S. Big Data: An Emerging Trend in Future 2014;5(1):538-541.
11. Hashem IAT, Yaqoob I, Anuar NB, Mokhtar S, Gani A, Ullah Khan S. The rise of big data on cloud computing: Review and open research issues, Inf. Syst 2015;47:98-115, doi: 10.1016/j.is.2014.07.006.
12. Gandomi A, Haider M. International Journal of Information Management Beyond the hype : Big data concepts, methods, and analytics, Int. J Inf. Manage 2015;35(2):137-144 doi: 10.1016/j.ijinfomgt.2014.10.007.
13. Gepp A, Linnenluecke MK, Neill TJO, Smith T. Big data techniques in auditing research and practice: Current trends and future opportunities, J Account. Lit 2017;40:102-115, doi: 10.1016/j.acclit.2017.05.003.
14. Ur Rehman MH, Chang V, Batool A, Ying T. Big data reduction framework for value creation in sustainable enterprises, Int. J Inf. Manage 2016;36:917-928, doi: 10.1016/j.ijinfomgt.2016.05.013.
15. Siddiqa A *et al.*, A survey of big data management : Taxonomy and state-of-the-art 2016;71:151-166, doi: 10.1016/j.jnca.2016.04.008.
16. DN, PT, Isah Haruna. Research Opportunities and Challenges in Big Data BigDat2015 Lessons 2015.
17. Ahuja SP, Moore B. State of Big Data Analysis in the Cloud 2013;2(1):62-68, doi: 10.5539/nct.v2n1p62.
18. Bradlow ET, Gangwar M, Kopalle P, Voleti S. The Role of Big Data and Predictive Analytics in Retailing, J Retail 2017;93:79-95, doi: 10.1016/j.jretai.2016.12.004.
19. Yan C, Song Y, Wang J, Guo W. Eliminating the Redundancy in MapReduce-based Entity Resolution 2015, 1233-1236, doi: 10.1109/CCGrid.2015.24.
20. Shin D. Demystifying big data: Anatomy of big data developmental process, Telecomm. Policy 2016;40(9):837-854, doi: 10.1016/j.telpol.2015.03.007.
21. Qasha R, Wen Z, Cała J, Watson P. Sharing and performance optimization of reproducible workflows in the cloud, Futur. Gener. Comput. Syst 2019;98:487-502, doi: 10.1016/j.future.2019.03.045.
22. Comput JPD, Arslan E, Pehlivan BA, Kosar T. Big data transfer optimization through adaptive parameter

tuning, J Parallel Distrib. Comput 2018;120:89-100, doi: 10.1016/j.jpdc.2018.05.003.
23. Vahi K, Rynge M, Juve G, Mayani R, Deelman E, Del Rey M. Rethinking Data Management for Big Data Scientific Workflows, in 2013 IEEE International Conference on Big Data 2013, 27-35.
24. Lu F, Shi Z, Gu L, Jin H, Yang LT. An adaptive multi-level caching strategy for Distributed Database System, Futur. Gener. Comput. Syst 2019;97:61-68, doi: 10.1016/j.future.2018.11.050.
25. Yu S, Brownlee N, Mahanti A. Comparative Performance Analysis of High-speed Transfer Protocols for Big Data, in 38th Annual IEEE Conference on Local Computer Networks 2013, 292-295.
26. Yin J, Zhao D, RWA. Data Confidentiality Challenges in Big Data Applications, in IEEE International Conference on Big Data (Big Data) 2015(8), 2886-2888.
27. Yi JH, Deb S, Dong J, Alavi AH, Wang GG. An improved NSGA-III algorithm with adaptive mutation operator for Big Data optimization problems, Futur. Gener. Comput. Syst 2018;88:571-585 doi: 10.1016/j.future.2018.06.008.
28. Yildirim E, Arslan E, Kim J, Kosar T. Application-level optimization of big data transfers through pipelining, parallelism and concurrency, IEEE Trans. Cloud Comput 2016;4(1):63-75, doi: 10.1109/TCC.2015.2415804.
29. Mohamed SH, El-Gorashi TEH, Elmirghani JMH. A survey of big data machine learning applications optimization in cloud data centers and networks, arXiv, vol. i 2019.
30. Marcellin MW, Gormish MJ, Bilgin A, Boliek MP. An Overview of JPEG-2000, in Proceedings of Data Compression Conference 2000, 523-544. [Online]. Available: http://citeseer.nj.nec.com/marcellin00overview.html.
31. Christopoulos C, Skodras A, Ebrahimi T. The JPEG still image coding system: an overview, IEEE Trans. Consum. Electron 2000;46(4):1103-1127.
32. Ayub S, Beg MS, Fook TT. JPEG and wavelet compression, in Proceedings of IEEE TENCON'02, 2000, 590-593.
33. Schelkens P, Munteanu A, Tzannes A, Brislawn C. JPEG2000 Part 10- Volumetric Data Encoding, in IEEE International Symposium on Circuits and Systems, 2006, 3874-3877.
34. Nguyen C, Redinbo GR. Fault Tolerance Design in JPEG Image Compression System, IEEE Trans. Dependable Secur. Comput 2005;2(1):57-75.
35. Santa-cruz D, Ebrahimi T. A Study of JPEG still image coding versus other standards, in 10th European Signal Processing Conference 2000, [Online]. Available: http://www.jpeg.org/public/wg1n1816.pdf.
36. Marjani M *et al.*, Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges, IEEE Access 2017;5:5247-5261, doi: 10.1109/ACCESS.2017.2689040.
37. Kaseb MR, Khafagy MH, Ali IA, Saad EM. An improved technique for increasing availability in Big Data replication, Futur. Gener. Comput. Syst 2019;91:493-505, doi: 10.1016/j.future.2018.08.015.
38. Hashem IAT, Yaqoob I, Anuar NB, Mokhtar S, Gani A, Ullah Khan S. The rise of big data on cloud computing: Review and open research issues, Inf. Syst 2015;47:98-115, doi: 10.1016/j.is.2014.07.006.
39. Khaloufi H, Abouelmehdi K, Beni-Hssane A, Saadi M. Security model for big healthcare data lifecycle, Procedia Comput. Sci 2018;141:294-301, doi: 10.1016/j.procs.2018.10.199.
40. Oguntimilehin A, Ademola EO. A Review of Big Data Management, Benefits and Challenges, J Emerg. Trends Comput. Inf. Sci 2014;5(6):433-438, [Online]. Available: http://www.cisjournal.org.
41. Raju Sake, P Mohammed Akhtar. Fitting of modified exponential model between rainfall and ground water levels: A case study. Int J Stat Appl Math 2019;4(4):01-06.