applied sciences

MDPI

Article

PySpark-Based Optimization of Microwave Image Reconstruction Algorithm for Head Imaging Big Data on High-Performance Computing and Google Cloud Platform

Rahmat Ullah * and Tughrul Arslan

School of Engineering, University of Edinburgh, Edinburgh EH9 3FF, UK; tughrul.arslan@ed.ac.uk
* Correspondence: rahmat.orakzai@ed.ac.uk

Received: 6 April 2020; Accepted: 9 May 2020; Published: 14 May 2020

Abstract: For processing large-scale medical imaging data, adopting high-performance computing and cloud-based resources are getting attention rapidly. Due to its low–cost and non-invasive nature, microwave technology is being investigated for breast and brain imaging. Microwave imaging via space-time algorithm and its extended versions are commonly used, as it provides high-quality images. However, due to intensive computation and sequential execution, these algorithms are not capable of producing images in an acceptable time. In this paper, a parallel microwave image reconstruction algorithm based on Apache Spark on high-performance computing and Google Cloud Platform is proposed. The input data is first converted to a resilient distributed data set and then distributed to multiple nodes on a cluster. The subset of pixel data is calculated in parallel on these nodes, and the results are retrieved to a master node for image reconstruction. Using Apache Spark, the performance of the parallel microwave image reconstruction algorithm is evaluated on high-performance computing and Google Cloud Platform, which shows an average speed increase of 28.56 times on four homogeneous computing nodes. Experimental results revealed that the proposed parallel microwave image reconstruction algorithm fully inherits the parallelism, resulting in fast reconstruction of images from radio frequency sensor's data. This paper also illustrates that the proposed algorithm is generalized and can be deployed on any master-slave architecture.

Keywords: Apache Spark; big data applications in imaging; cloud computing; medical image reconstruction; parallel algorithm

1. Introduction

Due to the availability of new data sources, there is a need to rethink existing architectures that enable big data analytics, especially in healthcare. Some sources include mobile phones (sensors), genomics, sensor networks, social media, internet of things, and medical imaging modalities that determine medical diagnostic and medical records. All these sources produce data at a rapid pace, which needs to be stored and processed somewhere in an efficient way [1]. Storing and analyzing such a huge amount of data is a challenge, especially when it does not conform to the traditional notion of data structures. Big data characteristics include huge volume, wide variety, high velocity, veracity, and value. The processing of these data requires new architecture, new tools, and new analytical methods [2]. Distributed and parallel computing architectures are preferred approaches to process such large scale unstructured and semi-structured data. Using technologies, such as Hadoop and Spark, to perform natural language processing on these data can significantly improve performance and create valuable information.

The image data processing is an important aspect of clinical diagnosis; for example, the correct analysis and interpretation of images are vital for the early detection of diseases [3]. The use of state-of-the-art devices for medical imaging, such as X-ray, magnetic resonance imaging (MRI), computerized tomography (CT), and positron emission tomography (PET), has greatly improved data acquisition process. These imaging modalities are being used for spatial representation of different biological tissues for different diagnosis. CT combines multiple x-ray scans to reconstruct the image representing a slice/volume of the imaged object. The advantages of CT imaging include its excellent resolution and contrast, along with being relatively faster than MRI. However, it requires larger dose of ionizing radiation, as well as being slower and more complex, compared to planar X-ray. X-ray and CT expose patients to ionizing radiation that increases the risk of cell mutation. Moreover, the equipment is costly and therefore not available at all hospitals [4]. Conversely, ultrasound is a low-cost solution, due to its cost-effectiveness and portable ability. The disadvantages of ultrasound is that it requires a good physical contact with the imaged tissue for a high wave impedance contrast to reconstruct high resolution image [5]. Photoacoustic imaging (PAI) is an emerging modality for imaging human tissues based on thermoelastic effect. This imaging modality used short laser pulse to reconstruct the optical absorption map of the human tissue. The main advantages of PAI is combining the high spatial resolution of ultrasound and high contrast of optical imaging. PAI is a multiscale imaging modality used in different preclinical and clinical applications, such as tumor detection [6] and functional imaging [7]. The downside of PAI are inherent artifacts in the reconstructed images caused by imperfect reconstruction algorithms [8].

The most common and one of the newest imaging modality is MRI. It is a tomographic technique based on the phenomenon of nuclear magnetic resonance. It produces high-quality images having excellent spatial resolution in the order of millimeters and is more suited for imaging soft tissues (e.g., distinguishing white and grey matter in the human brain) due to higher concentration of the hydrogen molecules. The disadvantages of MRI are its high cost of equipment and maintenance, system complexity and time needed for data acquisition [9]. A single scan of MRI takes about 5 min. Furthermore, MRI is not suitable in different situations, such as for patients with a heart pacemaker, as magnetic fields can upset the operations of pacemakers. In addition, heating can occur in some metallic implants [10].

Microwave imaging is an alternative imaging modality being investigated for different biomedical applications, such as breast cancer and brain imaging [11]. It is non-ionizing with emission power less than cellular devices radiation and less system complexity similar to ultrasound technology. Furthermore, it offers safe, low-cost, and portable method for medical imaging applications [12]. The downside of microwave imaging is it does not create images of high resolutions as compared to CT and MRI. However, due to its advantages, many research groups are exploiting improved antenna design, imaging setup optimization, tissue scattering modeling, development and optimization of image reconstruction algorithms, and in vitro and in vivo validation of these imaging systems. They have successfully designed and developed prototypes, as well as conducted trials on volunteers and/or patients, mainly targeting breast [13,14] and brain imaging [15–17].

Large-scale image processing applications, especially those involving multistage analysis, often indicate substantial inconsistency in processing times ranging from a few seconds to several hours [18]. Microwave techniques in medical imaging and remote sensing applications reconstruct the image of dielectric regions to retrieve the location, shape, and size of the targeted object. For many years, new optimization algorithms have been studied to address the compute-intensive inverse scattering problem associated with these techniques [19].

Field programmable gates arrays (FPGAs) and graphics processing unit (GPU)-based parallel approaches are commonly used to solve problems of image processing, especially in multi-modality image diagnoses, that require huge processing time [20,21]. Although, these approaches improve processing speed significantly, they are limited by non-scalability and higher cost. The use of parallel or cloud computing technology for image-guided diagnostics is a recent trend and has caught the

attention of many researchers in the field [22]. Some of the potential reasons to use cloud computing for medical image analysis are huge data storage, remote availability, faster processing demands, and scalability. In order to enhance the efficacy and accuracy of imaging modalities, medical imaging societies need to consider and embrace parallel approaches using high-performance computing (HPC) or cloud computing technologies as accelerators and use them as powerful tools for multi-modality imaging data analysis. Big data frameworks, such as Hadoop and Spark, are getting attention as it makes possible the scalable utilization of these parallel systems [23]. However, parallel implementation of algorithms on these frameworks faces several technical challenges depending on the level of their complexity. An inexpensive solution is to place the required data on the computational nodes to avoid network saturation. Hadoop and Map-reduce provide tools to address this issue; however, they are still not widely being integrated into medical image processing, despite having shown great success in online commerce, social media, and video streaming [24].

Data acquisition from sensors and its image reconstruction process using different algorithms and machine learning (ML) techniques is a recent trend and is gaining a lot of interest [25]. Consequently, to make it more efficient and scalable, healthcare entities are adopting big data frameworks, for instance, Apache Hadoop and Spark [26]. These are the most common big data processing frameworks being used for the implementation of the algorithm over distributed systems. The main goal is to process a huge amount of data on a cluster of hundreds or thousands of machines in parallel, which helps in the real-time decision-making process.

This paper aimed to contribute to this growing area of research by exploring the use of Apache Spark on HPC and Google Cloud Platform (GCP) for microwave imaging of head diagnostics. This work has made use of Eddie [27] provided by Edinburgh Compute and Data Facility. Eddie is the third generation HPC cluster running a Linux operating system. It consists of some 7000 Intel® Xeon® cores, with up to 3 TB of memory available per compute node. The cluster can be used to run jobs in parallel, taking advantage of its multiple compute core. It uses Open Multi-Processing (OpenMP) to run memory-intensive jobs and shared memory programs. The work provides two different implementations of the microwave-based image reconstruction algorithm. The sequential implementation is based on the general Python using NumPy array, while the parallel implementation is based on PySpark. The goal is to assess the effectiveness of using big data technologies for medical image reconstruction that involves multiple iterations on a massive amount of data points. The main contributions of this paper are as follows:

- The parallelism of the microwave image reconstruction (MIR) algorithm is first identified. Then, the computational model of Apache Spark is studied, and the parallel version of the algorithm is presented.
- A novel distributed approach, which adopts both data and algorithm in parallel, for efficient image reconstruction of head imaging is proposed. The imaging algorithm is optimized using PySpark on HPC clusters to improve the processing speed and make it real-time. The imaging system retrieves input data generated through radio frequency (RF) sensors and store in Eddie. An integrated imaging algorithm optimized through PySpark creates and saves images back to Eddie Storage.

The proposed parallel microwave image reconstruction (PMIR) algorithm is implemented on the PySpark version of Apache Spark. Experimental results indicate that the proposed PMIR algorithm fully inherits the parallelism of the MIR algorithm, resulting in the fast reconstruction of images from RF sensor's data. The proposed parallel algorithm performs an average of 28.56 times faster than sequential Python implementation.

The remaining part of the paper proceeds as follows: In Section 2, related literature on image processing on cloud and/or HPC is presented. Section 3 briefly describes image reconstruction, the microwave imaging reconstruction algorithm, Hadoop, and Spark, along with the Map-reduce programming model. In Section 4, data acquisition through RF sensors and the implementation of

the PMIR algorithm are presented in detail. Section 5 describes the experimental results. Finally, conclusions and future work are presented in Section 6.

2. Related Work

A considerable amount of work had been published on the use of HPC or cloud computing for healthcare big data processing in health. Most of the research focuses on images/data analysis acquired through traditional imaging modalities. This section highlights the most recent work on the use of HPC, cloud computing, and big data frameworks for medical image/data processing. An overview of the recent tools and techniques proposed for large scale image data processing using cloud and big data frameworks can be found in Table 1.

A cloud-based system to process and analyze MRI Scans was proposed in Reference [28]. The tool was integrated with a threshold-based brain segmentation method to analyze regions of interest. It enabled users to upload their data on a simplified web page. The proposed tool has several benefits, such as better speed due to scalability and automatic up-gradation of the system without bothering the end-users. Experimental data was used to test the performances and functionality of the system. However, no parallel computing techniques were taken into consideration. A smart cloud system is proposed to address the problems related to big data storage by providing infrastructure support to small hospitals with limited resources [29]. The confidence connected segmentation method was used to detect the tumor object. It allowed users to upload MRI or CT scan images for detecting the tumor in real-time.

Table 1. Techniques for large scale image data processing using high-performance computing (HPC)/Cloud.

Proposed Technique	Advantages	Limitations	Ref.
ASL-magnetic resonance imaging (MRI) Cloud	Cloud-based tool to process MRI data, image segmentation, better performance due to scalability	No parallel techniques implemented	[28]
Smart Cloud System	Cloud-based system for processing Neuroimaging data, image compression	No parallel techniques implemented, network congestion	[29]
cone-beam computerized tomography (CT) reconstruction	Faster reconstruction of image using Hadoop	No in-memory computation, no evaluation on computing cluster	[30]
Parallel Fuzzy c-Means Segmentation Algorithm	12.54 time speed improvement, in-memory computation	No generality evaluation	[31]
Muscle image segmentation algorithm	10 time speed improvement, in-memory computation, both data and model parallelism	No generality evaluation	[32]
Parallel computing approach to accelerate microscopy data	High throughput	Does not considered big data frameworks	[33]

A special issue on the use of cloud and parallel computing have been published in the Journal of X-Ray Science and Technology to highlight the state-of-the-art advances in image-guided diagnosis and therapy [22]. For instance, the use of parallel computing is investigated for modeling blood flow using computational fluid dynamics and smooth particle hydrodynamics in Reference [34]. An imaging application for neurological diseases and neuroimaging is presented in Reference [35]. An approach called "cloud-based medical image processing- as-a-service", utilizing the Apache Hadoop framework and HBase is proposed for efficient processing and retrieval of medical images [36]. Similarly, a cloud-based medical imaging system was designed and developed to reconstruct images for cancer patients using Proton Computing Tomography [37]. The proposed service uses Amazon Web Service (AWS) to process multi-client requests simultaneously.

A generalized flow for medical image classification was proposed based on Azure cloud by Roychowdhury et al. [38]. Similarly, Serrano et al. [39] stated that big data paradigms fit very well for the workflow needed for image reconstruction. The paper focuses on improving the image quality of CT scan through the iterative reconstruction algorithm. The work also evaluated the costs associated with different execution policies on a single node and distributed configurations. A processing pipeline for processing functional magnetic resonance imaging (fMRI) based upon PySpark was proposed in [30]. The proposed pipeline was evaluated on a single node, and the results revealed an improvement of fourfold. However, the efficiency was not evaluated on a computing cluster.

An ultra-fast, scalable, and reliable reconstruction method for four-dimensional cone-beam CT was developed using Map-reduce [40]. The work revealed the potential use of Map-reduce to solve large-scale imaging problems using cloud computing. A tool called "Java Image Science Toolkit (JIST)" was developed and integrated with AWS [41]. Similarly, a Spark-based image segmentation algorithm was proposed to accelerate the processing of agriculture image big data [31].The input images are converted into pixel data and then distributed to the computing platform. In order to improve the efficiency of iterative computing, membership degrees of a pixel are calculated and updated in parallel. The experimental results indicated that the proposed Spark-based fuzzy C-means algorithm can perform 12.54 times faster on ten computing nodes.

A distributed computing approach was implemented for the segmentation of skeletal muscle cell images using Apache Spark [32]. The authors aim to balance the load on available Spark cluster resources. A parallelized region selection algorithm based on the hierarchical tree was proposed, for the efficient segmentation of muscle cells. In the same vein, an architecture for medical image analysis based upon Hadoop and Spark was proposed in Reference [42]. Image classification was performed as a case study. The proposed architecture contains two parts of medical image big data processing relies on traditional imaging modalities, such as CT, MRI, and X-ray. Moreover, they reviewed existing algorithms used for image classification, compression, and prediction and proposed a theoretical framework based on traditional computing. A parallel computing approach to accelerate single-molecule localization microscopy data processing using ThunderSTORM on the HPC cluster was proposed in Reference [33]. The performance was compared to a desktop workstation, which indicated that data processing speed can be improved by accessing more HPC resources.

The aforementioned cloud or parallel computing-based services mainly focus on hosting the application on the cloud for experiencing the benefit of large storage or to ease image retrieval of X-ray, CT scan, or PET scan images. Moreover, the use of big data frameworks were also considered for image segmentation, including agriculture and medical images. However, they do not aim to deal with novel modalities for image reconstruction, such as microwave imaging for breast and head diagnostics. Moreover, none of them consider the performance improvement of image reconstructions using Apache Spark for head imaging, using radio frequency data.

3. Image Reconstruction and the MIR Algorithm

The microwave imaging system is getting attention of researchers in recent years for breast cancer and brain tumor/stroke detection. The microwave imaging system has low-cost components and is lightweight and non-ionizing as compared to other imaging modalities, such as X-ray, CT, and PET scans. Microwave imaging via space-time (MIST) algorithm and its extended/modified versions are most used for the purpose. The MIST beamforming was originally proposed for the detection of breast cancer in 2003 [43]. It had attracted many research groups throughout the world working on breast, brain or lung cancer/tumor detection. Many relevant papers have been published using this algorithm [44,45] and are evaluated on clinical practices for breast cancer detection [46]. The main advantage includes its understandability and high-resolution image. Moreover, it can detect small tumors up to 2 mm. Although MIST has all these advantages associated with its use, the time it takes to process input data and produce the output images does not lie in an acceptable range due to plenty

of iterative operations and accumulations of signal data. Therefore, it needs to be optimized before it is used for real-time imaging in clinical diagnosis.

3.1. Principle of MIR Algorithm

The main principle of these algorithms is to calculate the difference in the dielectric constant also called relative permittivity of the targeted human tissue. An array of antennas is set around the target tissue (breast or brain). The tissue is penetrated with signals and the backscattered signal are stored. In order to create the energy map of the targeted tissue, the energy value for each pixel in the imaging area is calculated and saved. Due to the distance between antennas and points, points correspond to signal shifts creating a signal delay in time. Therefore, delay values are calculated first to align the signals and get the energy for each pixel. The input signals are then processed to find out the location and size of malignant tissue. The position of maligned tissue is located by analyzing the energy level for all pixels based on the difference in dielectric properties of healthy and malignant tissue. Figure 1 illustrates the basic principle of the MIST beam-forming for a single focal point in the imaged object. The backscattered signals received on each antenna for a single focal point are time aligned and summed together to get the energy. The process is repeated for total number of points to create an image.

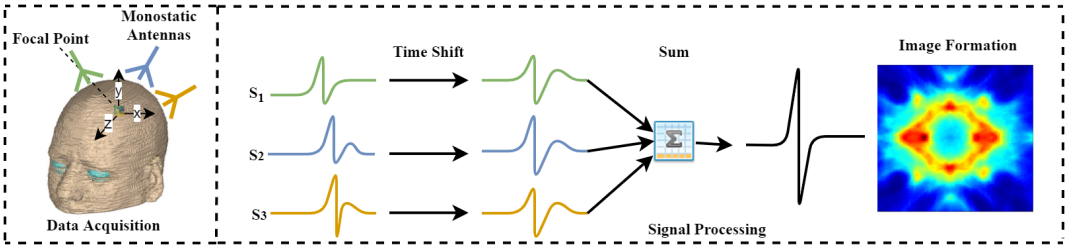

Figure 1. Microwave imaging via space-time (MIST) beam-forming process for a focal point. S_1, S_2, and S_3 are backscattered signal. These signals are first time shifted and then summed together to find the energy for the specified focal point.

To produce the energy map, the amplitude of backscattered signals is calculated for all pixels. In order to make sure that all the signals summed properly, delay shift values for all pixels are calculated. Since delay shift values are proportional to the distance between antennas and imaging points, the distance between antenna and pixel therefore needs to be calculated first. This depends on the total number of antennas and stored input points. The distance is calculated using the equation below [47].

$$\tau_n(\overrightarrow{r}) = \frac{\sqrt{(x - x_o)^2 + (y - y_o)^2} * N}{d}, \tag{1}$$

where x and y are the x, y coordinates of antennas, x_o and y_o are the position of pixel in the imaged tissue in Cartesian coordinates, τ_n is the propagation time of signal n, $\overrightarrow{r}$ is the focal point in the imaging object, d is the diameter, and N represents the total number of input points. Then, the intensity values are calculated as:

$$I[n] = \left| \sum_{i=1}^{N} A_n\left(\tau_n\left(\overrightarrow{r}\right)\right) \right|^2, \tag{2}$$

where A_n represents the signal of antenna A at location n. Algorithm 1 gives the pseudo-code for sequential Python-based implementation of the MIR algorithm. The algorithm first read sensors data and antenna's location in order to compute delay and pixel values. Two loops are used to get delay shift values for each signal based on all antennas. The loop is repeated to compute the matrix of pixels. The body of the loop is based on Equation (1). The number of delay shift samples is calculated by multiplying the total number of samples. Finally, delay shift values are stored and used to calculate the final values of energy.

Algorithm 1 MIR Algorithm

Input: Radio Frequency Sensors Data
Output: Brain Images

for $A_n \leftarrow 1$ to A_t **do**
  for x, y $\leftarrow 1$ to N_p **do**
    Compute $\tau_n\left(\overrightarrow{r}\right)$ using Equation (1)
  end for
end for
for u, v $\leftarrow 1$ to N_p **do**
  $E_{nxn} = 0$
  for $\forall$ S $\leftarrow S_s$ **do**
    $E_{ni} = 0$
    for do $A_n \leftarrow 1$ to A_t
      if S_s in range **then**
        $E_{ni} = E_{ni} + A_n + \tau_n\left(\overrightarrow{r}\right)$ as described in Equation (2)
      end if
    end for
    $E_{nxn} = E_{nxn} + (E_{ni} * E_{ni})$
  end for
end for

I_{nxn}= round (E_{nxn})

In order to calculate the matrix of pixel values, delay matrix and input signals are used as inputs. The two for loops are used to calculate the matrix of pixels. To calculate the energy for the next pixel, the value of the current pixel is set to zero. The second and third for loop is used to traverse all signal samples to make sure the signal range is valid. Backscattered signals from all antennas are first aligned and then summed in the loop body based on Equation (2). The 'Eni' represents the energy for all samples from all antennas separately. Consequently, it is set to zero after finishing all the antennas. Since the energy level is proportional to signal amplitude, irrespective of positive or negative, energy is therefore accumulated by taking the square of 'Eni'. Next, to store the results, "round" function is used to get rounded values, which improve the efficiency of the algorithm without affecting the quality of resulting images. Finally, the energy is stored in a two- dimensional matrix.

To optimize the algorithm, the intensive operation needs to be parallelized. The proposed parallelized version of the algorithm can be found in Section 4.

3.2. Apache Spark Framework

Recently, there has been renewed interest in using big data frameworks, such as Apache Hadoop and Spark, for large scale data/image processing. Apache Hadoop based on the Map-reduce programming model provides a relatively structured and easy approach to process big-data on commodity clusters. However, for computations intensive applications, Apache Spark shows better performance due to its in-memory processing. Apache Spark is an open-source framework originally developed by Berkeley's AMP Lab based on the Map-reduce programming paradigm [19]. The Map-reduce is a programming paradigm that allows users to break larger tasks into smaller tasks, run it in parallel, and integrate the output of each task into the final output. The Hadoop Distributed File System (HDFS) provides this capability. It is a distributed file system designed to run on commodity hardware. HDFS let users distribute data to multiple nodes across a cluster and take advantage of the parallel processing of Map-reduce.

A typical Map-reduce program contains three classes as shown in Figure 2. The driver class initializes a Spark Context (SC) object, which coordinates the independent set of processes running on a cluster. The SC can be connected to many kinds of cluster managers, such as standalone, YARN [48], or Mesos [49]. The purpose of these cluster managers is to allocate resources to multiple applications. Once created, it gets executors on different nodes, which are processes that run computations and store data for the application. Finally, it sends the application code (Python files in this work) to the executors. Spark programs can be run on Hadoop YARN, Kubernetes, Apache Mesos, and standalone

cluster mode, as well as in the cloud. It can access data from various sources including HDFS, Apache Cassandra, HBase, and many others.

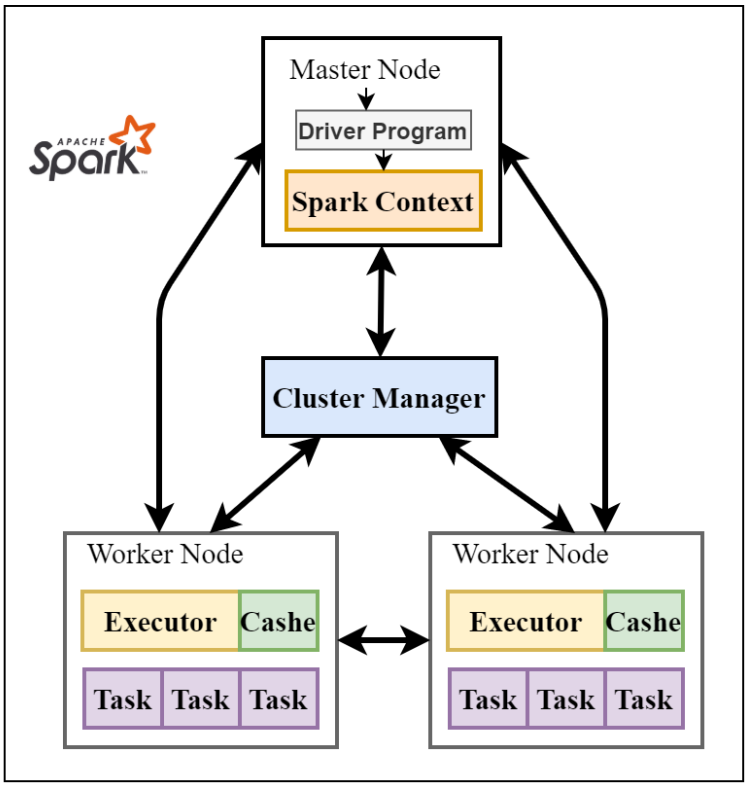

Figure 2. Overview of Spark framework. The master node contains driver program, which drives the application by creating Spark context object. Spark context object works with cluster manager to manage different jobs. Worker nodes job is to execute the tasks and return the results to Master node.

Apache Spark is different than Hadoop, as it has a distributed memory model and allows to store intermediate results in cashed memory in order to improve the performance. Zaharia and his team's paper [50] states that, while Spark was still currently a working prototype, the performance results were very encouraging. Even at that time, Spark could outperform ML workloads by a factor of 10. They crash a node to demonstrate that Spark could continue to function with fewer nodes and compared results with the Hadoop implementation. With Hadoop, each iteration would take over 2 min. In the case of Spark, the first iteration took almost 3 min because the data was cached for further iterations, but only took six seconds to complete each of the remaining iterations.

PySpark is a Python library that give access to Spark using Python programming language. It used Py4J library to handle communications with Spark. Advantages include easy integration with other languages, such as Java, Scala, and R. It uses the resilient distributed data set (RDD) to speed up the executions by performing in-memory computations. The RDD is the architectural foundation of Spark, which are read-only sets of data, spread over a cluster of different machines, being operated in parallel. Records are split into logical partitions and distributed to all machines in the cluster, allowing RDD to provide high-level programming interface by hiding the data partitions. These logical Partitions are the primary units of parallelism. Spark supports two types of operations to work on these partitions: transformations, such as "map", which creates new RDDs; and actions, such as "reduce", which returns a value to the driver program after performing an operation on the data set. The transformation operation creates new RDDs and, consequently, creates dependency between the old and the new RDDs.

In short, Apache Spark is an efficient and general-purpose engine for big data processing. It was initially developed specifically for data science but has evolved to support yet more use

cases, including real-time processing. The four core reasons to use Spark include its speed, usability, generality, and platform agnostic. Compared to Apache Mahout, which runs on disk instead of memory, benchmark tests show about 9-fold improvement. Furthermore, Spark is being used for both batch and stream processing [51].

4. Data Acquisition and Parallel Design and Implementation of MIR Algorithm

For this work, the input data is obtained by using an array of six antennas in a hat-like structure around the head. The signals are obtained using the mono-static radar mode operation; where the same antenna is used to transmit and receive signals. The antennas were arranged around the inner side of the absorber with an equal distance of 9 mm. The signals are generated by the Vector Network Analyzer (VNA) having a frequency range of 1 to 3 GHz. This operation is performed using HP 8753 VNA, with a dynamic range of up to 100 dB. Cylindrical scanning is performed to get the back-scattering signal s11, using six antennas on different locations covering the whole head. The data is stored in text format, showing S11 parameters along with a frequency range of 1–3 GHz. Six sets of data, one for each antenna, making 1206 points in total are then used as input to the algorithm for image reconstruction. Detailed information on the experimental setup, VNA used, and wearable device can be found in our previous work [52].

4.1. Identifying the Parallelism of MIR Algorithm

To identify the parallelism in the MIR algorithm, the sequential implementation is analyzed to find the time-intensive part, as well as the dependency among processes. The most compute-intensive part is the number of iterations required to construct a matrix of pixel values for an image, due to multiple iterations and massive input data. In the process, each iteration contains many parallel processes, creating an opportunity for parallel design and implementation. The number of iterations required for the reconstruction of one image is explained as follows:

Suppose P_1, P_2 ... P_n are the patients, and D_1, D_2...D_n denotes the number of times each patient is diagnosed and the data is stored. So, we have

$$C_r = \sum_{p=1}^{n} \sum_{d=1}^{n} P_i D_i A_n N_{ip}, \tag{3}$$

where N_{ip} denote input points stored, and P_i and D_i denote patients and diagnostics, respectively.

For instance, if the number of input points (data) for each antenna is 200 (minimum required data points for image reconstructions reported in Reference [53,54]) and 06 antennas: 200 × 6 × 100 × 100 = 12,000,000 iterations is required to reconstruct an image of 100 × 100 pixels.

The most time-consuming part of the algorithm is delay calculations and pixel value calculations. To make each of these calculations more efficient, there is a need to implement a parallel algorithm that executes these data on multiple nodes in the cluster using big-data frameworks, such as Spark or Hadoop Ecosystem. The signal samples and location of antennas are uploaded to the Eddie distributed file system, and stored in distributed manner, which is input to the MIR algorithm.

According to the MIR algorithm, explained in Section 3, in order to calculate pixel points from input data, delay values need to be calculated first. Since it needs to be calculated once only, its parallel implementation therefore does not have any substantial effect on the efficiency of the algorithm. The most intensive part is the pixels value calculation, which needs to be parallelized. During this process, pixel values in different subsets are independent of each other; hence, the calculation of each pixel value in different subsets on different machines in the cluster is a good fit for parallelism.

4.2. Design and Implementation of PMIR Algorithm

The proposed parallelized version of the MIR algorithm is designed and implemented using PySpark. A detailed flowchart is shown in Figure 3. The sensor data is stored on Eddie distributed

file system. The master node reads the input data and distributes it to all workers in the cluster with shared memory. This also includes broadcasting locations of antennas in antenna array to each worker node. To make each of the calculation more efficient, resilient distributed data sets (RDDs) are used. The input data received through RF sensors is first converted to an RDD and stored in the distributed memory. An empty matrix is created and broadcasted, which is shared by different worker nodes.

The map operation from map-reduce programming model is then used to calculate subset of values in parallel using Spark context object. The "map" operation computes the pixel value on each worker node in the cluster, and update the empty matrix, in parallel. Finally, reduce operation sums up the different subsets from different computing nodes in the cluster and returns it to the master node, until signal samples are in range. The master node is used to reconstruct the images using simple Python functions. The resultant images are stored back to the Eddie distributed file system.

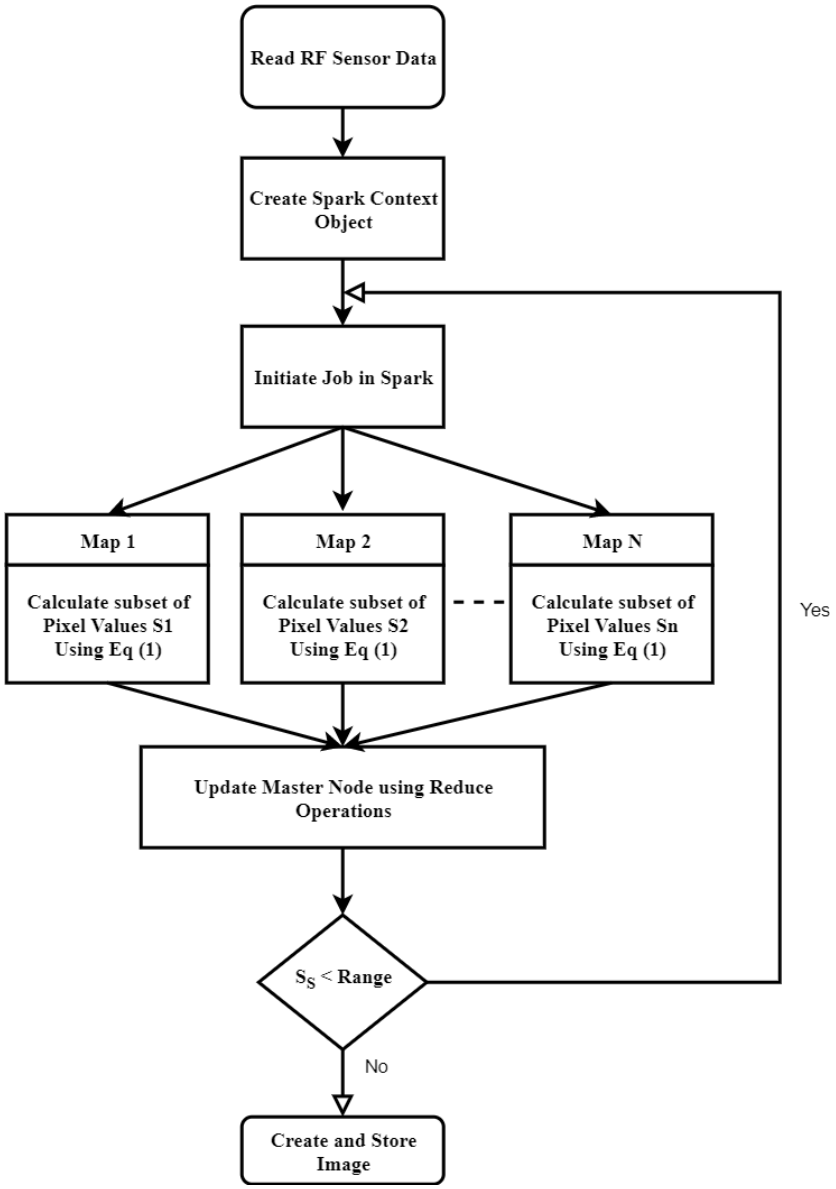

Figure 3. The parallelism of the microwave image reconstruction (MIR) algorithm. RF = radio frequency.

The parallel version of the MIR algorithm is implemented using Python programming language. The pseudo-code of the proposed parallelized MIR algorithm is shown in Algorithm 2.

The above process is repeated for all cases and the number of diagnostics performed.

Algorithm 2 PMIR algorithm

Input: Radio Frequency Sensors Data
Output: Brain Images

1: Read data from Eddie distributed file system
2: Copy the RDD for delay and Antennas location to each worker node
3: Set the variables z, distance, and energy to zero
4: **While** N_{ip} < range **do**
5: Calculate the subset of pixel points on each worker node in parallel based on Equation (2) using Map operation
6: Update the master node concurrently with a subset of pixel values using Reduce operation
7: **End While**
8: Reconstruct the image from the matrix of pixel values on master node
9: Save the image to the Distributed file system on Eddie

5. Experimental Evaluation

Using the PySpark based Map-reduce programming model, the parallel MIR algorithm for head imaging big data is implemented on Eddie. Eddie (see www.ecdf.ed.ac.uk) is the University of Edinburgh's research computing cluster. It has 7000 CPU cores. Each server (compute node) has up to 3 TB of RAM. It runs Open Grid schedule and Scientific Linux 7.x. An overview of Eddie architecture is shown in Figure 4.

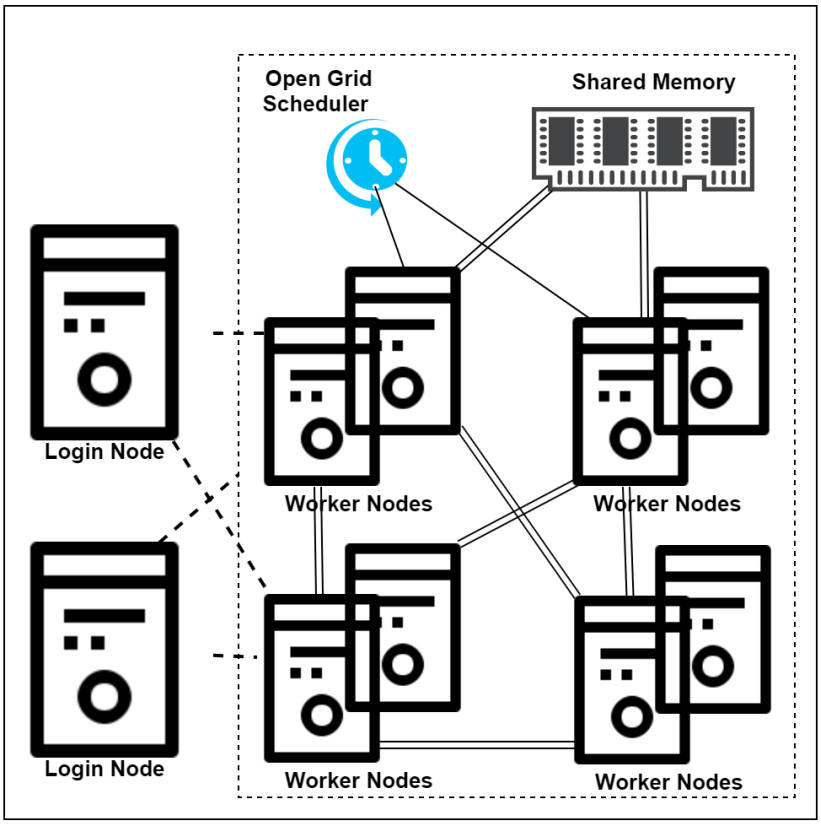

Figure 4. Overview of Eddie architecture. Login nodes lets users start an interactive session or submit batch jobs. The submitted jobs run on worker nodes connected through 10 Gbps Ethernet network depicted by double lines.

The login node has limited memory and lets users submit job scripts or start an interactive session. All these nodes are connected with a 10 Gbps Ethernet network. The Open Grid Scheduler is responsible for scheduling the jobs. All nodes have access to shared memory called Eddie distribute

file system. The user either submits batch jobs from login nodes to the cluster or starts an interactive session by specifying the number of cores required and memory per core.

After setting up the environment for PySpark and Python, PySpark was launched with a configuration of having one master and three workers node. Each node has the same configuration as shown in Table 2.

Table 2. Experimental setup (each node).

Item	Value
CPU	Intel® Xeon® Processor E5-2630 v3 (2.4 GHz)
Memory	16 G
Operating system	Scientific Linux 7
PySpark version	2.4.1
JDK version	1.8.0
Hadoop version	3.1.1
Python version	3.4.3

In order to run the job on the localhost, Spark magic is used. Spark magic is a set of tools used to connect to remote Spark clusters through Livy, a Spark REST server, in Jupyter notebooks, as shown in Figure 5.

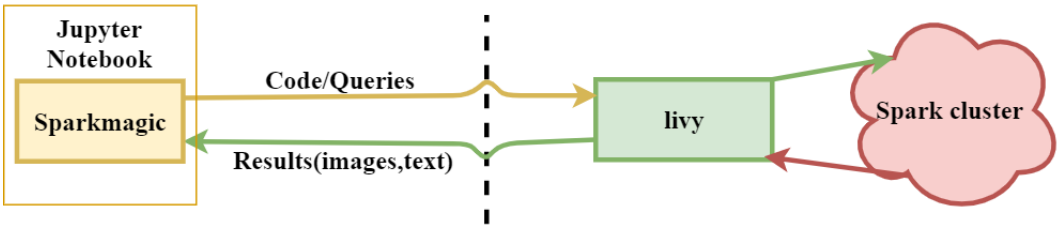

Figure 5. Spark magic provides tools for managing Spark sessions on an external cluster through Livy. Using Spark magic with Jupyter notebook, users can use Spark from their own Jupyter notebook, running on localhost.

The input data contains 06 antenna and 201 points each making 1206 input points in total for a single output image. The efficiency of the MIR and PMIR algorithm is evaluated based on single-image reconstruction in a homogeneous environment.

Performance Analysis of MIR and PMIR

The efficiency is evaluated using a serial approach and a parallelized approach based on the PySpark Map-reduce programming model. Both approaches are tested on Eddie and GCP. For the performance evaluation on Eddie, an interactive session was started with 04 nodes, each having 04 cores and 16 gigabytes of memory. The cluster was configured based on master-slave architecture, where one node acts as a master and others as workers. The master node reads the input data and location of antennas stored on the Eddie distributed file system and creates RDDs. These RDDs are broadcasted to the worker nodes. The worker nodes calculate subsets of pixel values based on the algorithm discussed previously and return the results to the master node using the Map-reduce programming model. The master node converts the matrix values into an image using simple Python functions and saves the results back to Eddie storage. An example image is shown in Figure 6.

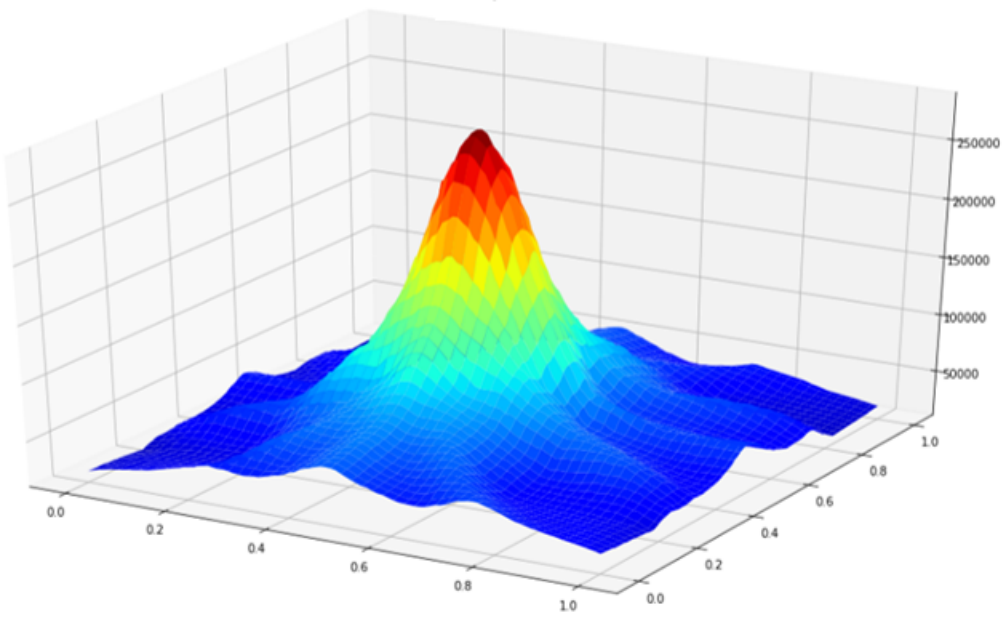

Figure 6. An example of the output image reconstructed from 1206 backscattered points collected using six antennas.

The serial MIR algorithm implemented in Python programming language was first run three times on the cluster and the average wall time was noted. Then the parallelized version of the same algorithm (keeping the input data and antenna location same) was run with the same configurations and the average wall time was noted. The result is shown in Figure 7 and Table 3. It can be noted that the average wall time decreased from 8.14 s to 285 ms, which means that using PySpark is 28.56 times faster than sequential Python implementation.

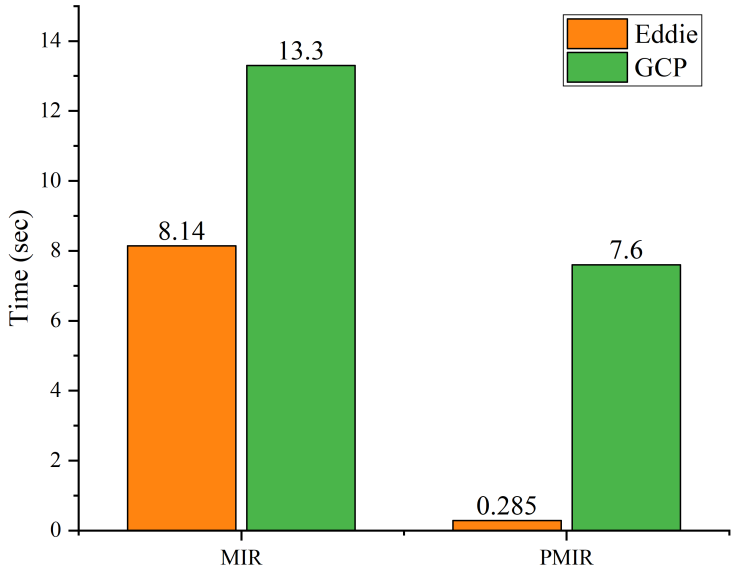

Figure 7. Execution time of MIR and parallel microwave image reconstruction (PMIR) algorithm on Eddie and Google Cloud Platform (GCP).

Table 3. Processing time (in seconds) of MIR and PMIR on stand-alone, Eddie, and GCP.

Algorithm	Stand-Alone	Eddie	GCP
MIR	14.91	8.14	13.3
PMIR	8.04	0.285	7.6

For validation purpose, the obtained results are compared with Reference [30]. We chose this paper because they use PySpark for the implementation of functional magnetic resonance imaging (FMRi) pipeline for brain extraction application (BEA). There are other relevant literature, as described in the Related Work section, however they used Hadoop and Spark instead of PySpark. The method was tested on standalone system using PySpark and standard Python. In terms of processing time, the proposed pipeline is four times faster than the one developed in Python. Although the testing was performed on a single node, it was reported that the performance will increase by a factor of 10 to 20 times using a cluster. As shown in Figure 8, PySpark improves the performance in terms of processing time for both MIR and BEA. The aim of this comparison is to determine the effectiveness of using PySpark for parallel implementation of imaging algorithm, rather than comparing the performance of MIR and BEA.

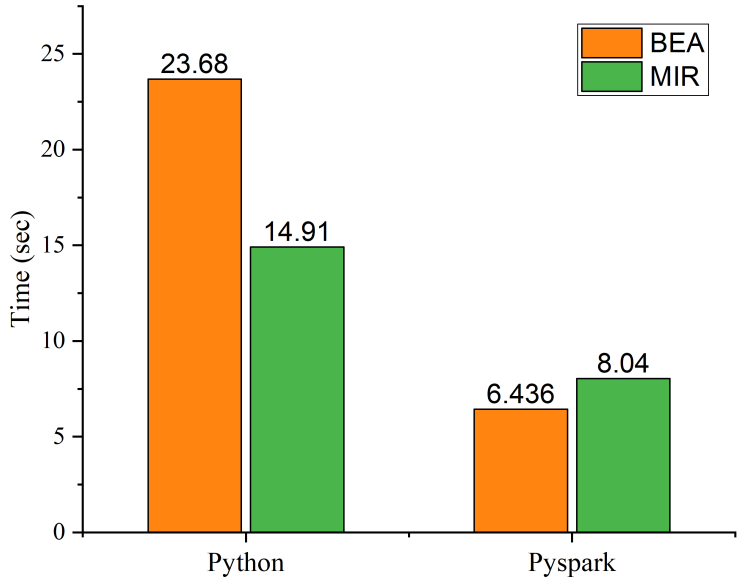

Figure 8. Execution time of MIR and brain extraction application (BEA) [30] on stand-alone system.

In order to check if the algorithm can be generalized, it is deployed on GCP using the Google storage bucket to store data and final output images. Three virtual machine instances were created with a master and two worker nodes, as shown in Figure 9. The master node reads input data and the antennas' location forms the Google storage bucket and runs the algorithm similar to Eddie. The results take slightly more time due to network saturation, the number of cores, and memory. The algorithm faces high latencies due to data transfers between the parallel file system and computation nodes. However, the PMIR algorithm outperforms the MIR algorithm on both Eddie and GCP. The experiment design on GCP does not aim to compare the performance with Eddie, rather evaluate the generality of the PMIR algorithm. The goal is to verify the behavior of the PMIR algorithm that maximizes the localization of data retrieval, processing, and storage for a job.

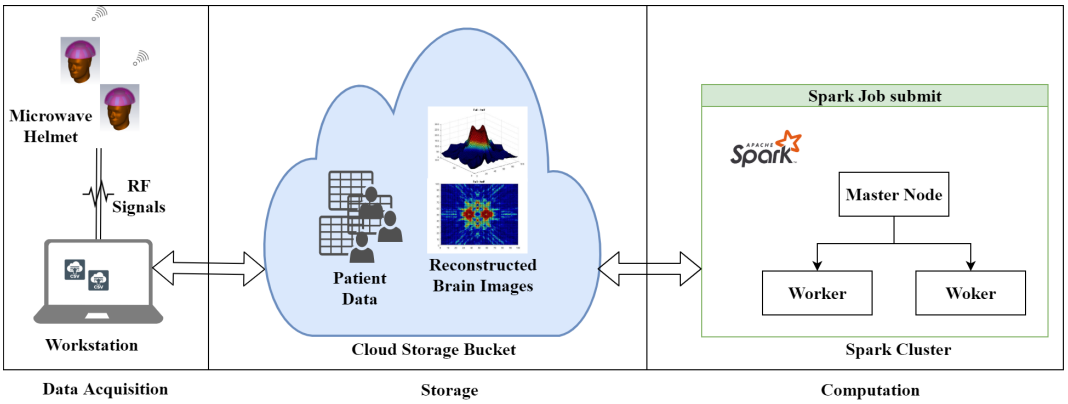

Figure 9. PMIR on Google Cloud Platform (GCP). The data is first send to cloud storage bucket. The master node reads the data, distributes to worker nodes for computation, and saves the output image back to storage bucket.

From the analysis on Eddie and GCP, it can be noticed that serializations overhead and network congestion are negligible for large medical image data. However, Spark provides very flexible solutions for large, fast image processing tasks. Although the use of big data technologies for large-scale image processing has become crucial for both research and clinical practice, very few modern big data approaches are considered in clinical practices. Hence, creating robust algorithms for large-scale image data processing becomes an enormous effort, particularly in the context of HPC and cloud platforms.

6. Conclusions and Future Work

This paper presented a parallel image reconstruction algorithm using PySpark on HPC and GCP platforms. The proposed system achieves high-throughput and effective image reconstruction. With master-worker parallelism, the computation required for image reconstruction is distributed onto multiple worker nodes using the Spark framework. On each worker node, delay and antenna location values are broadcasted first to assist faster computation. The subsets of pixel values are calculated on different worker nodes in parallel and retrieved to a master node where it is used to generate 2D and 3D images of the brain. We tested the performance of the proposed image reconstruction system on Eddie and Eddie distributed file system for storing the images. The aim of this work was to examine the use of big data technologies for large-scale image data processing. The study indicated that the proposed parallel imaging algorithm achieves more than 128% times speed improvement on the construction of multiple images in parallel and distributed manners. Although the algorithm is tested on Eddie and GCP, it can be generalized to any cluster having master-slave architecture. This study adds to the growing body of research that considers the use of different accelerators for medical image processing.

Although the Map-reduce programming model is commonly used to compute intensive jobs, it is not considered an all-purpose big data tool due to data constraints. One of the downside of the proposed algorithm is retrieving the data to memory on every cycle for image reconstruction, resulting in substantial disk input/output. The cloud implementation will require data retrieving on every iteration and can cause network congestion. A potential solution is to place data and algorithm on the computational nodes to avoid network congestion. The proposed algorithm is tested on homogeneous nodes. Further experimental investigations are needed to consider more data inputs and deploy the algorithm on both homogeneous and heterogeneous architecture.

Author Contributions: Conceptualization, R.U.; Methodology, R.U.; Software, R.U.; Validation, R.U.; Formal Analysis, R.U. & T.A.; Writing—Original Draft Preparation, R.U.; Writing—Review & Editing, R.U.; Supervision, T.A.; Project Administration, T.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This work has made use of the resources provided by the Edinburgh Compute and Data Facility ECDF http://www.ecdf.ed.ac.uk/.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this article.

Abbreviations

The following abbreviations are used in this manuscript:

HPC	High Performance Computing
GCP	Google Cloud Platform
RF	Radio Frequency
A_n	Signal of Antenna A at Location n
N_p	Number of Input Points
τ_n	Propagation time of signal n
S_s	Signal Samples
E_{ni}	Energy for Pixel i
RDD	Resilient Distributed Dataset

References

1. Manogaran, G.; Varatharajan, R.; Lopez, D.; Kumar, P.M.; Sundarasekar, R.; Thota, C. A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system. *Future Gener. Comput. Syst.* **2018**, *82*, 375–387. [CrossRef]
2. Makkie, M.; Li, X.; Quinn, S.; Lin, B.; Ye, J.; Mon, G.; Liu, T. A Distributed Computing Platform for fMRI Big Data Analytics. *IEEE Trans. Big Data* **2018**, *5*, 109–119. [CrossRef] [PubMed]
3. Dhayne, H.; Haque, R.; Kilany, R.; Taher, Y. In Search of Big Medical Data Integration Solutions—A Comprehensive Survey. *IEEE Access* **2019**, *7*, 91265–91290. [CrossRef]
4. Karadima, O.; Rahman, M.; Sotiriou, I.; Ghavami, N.; Lu, P.; Ahsan, S.; Kosmas, P. Experimental Validation of Microwave Tomography with the DBIM-TwIST Algorithm for Brain Stroke Detection and Classification. *Sensors* **2020**, *20*, 840. [CrossRef]
5. Makarov, S.N.; Noetscher, G.M.; Arum, S.; Rabiner, R.; Nazarian, A. concept of a Radiofrequency Device for osteopenia/osteoporosis Screening. *Sci. Rep.* **2020**, *10*, 3540. [CrossRef]
6. Guo, B.; Li, J.; Zmuda, H.; Sheplak, M. Multifrequency microwave-induced thermal acoustic imaging for breast cancer detection. *IEEE Trans. Biomed. Eng.* **2007**, *54*, 2000–2010. [CrossRef]
7. Nasiriavanaki, M.; Xia, J.; Wan, H.; Bauer, A.Q.; Culver, J.P.; Wang, L.V. High-resolution photoacoustic tomography of resting-state functional connectivity in the mouse brain. *Proc. Natl. Acad. Sci. USA* **2014**, *111*, 21–26. [CrossRef] [PubMed]
8. Mozaffarzadeh, M.; Mahloojifar, A.; Orooji, M.; Adabi, S.; Nasiriavanaki, M. Double-stage delay multiply and sum beamforming algorithm: Application to linear-array photoacoustic imaging. *IEEE Trans. Biomed. Eng.* **2017**, *65*, 31–42. [CrossRef]
9. Islam, M.; Mahmud, M.; Islam, M.T.; Kibria, S.; Samsuzzaman, M. A Low Cost and Portable Microwave Imaging System for Breast Tumor Detection Using UWB Directional Antenna array. *Sci. Rep.* **2019**, *9*, 15491. [CrossRef]
10. Chandra, R.; Zhou, H.; Balasingham, I.; Narayanan, R.M. On the opportunities and challenges in microwave medical sensing and imaging. *IEEE Trans. Biomed. Eng.* **2015**, *62*, 1667–1682. [CrossRef]
11. Stancombe, A.E.; Bialkowski, K.S.; Abbosh, A.M. Portable microwave head imaging system using software-defined radio and switching network. *IEEE J. Electromagn. RF Microwaves Med. Biol.* **2019**, *3*, 284–291. [CrossRef]
12. Bolomey, J.C. Crossed viewpoints on microwave-based imaging for medical diagnosis: From genesis to earliest clinical outcomes. In *The World of Applied Electromagnetics*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 369–414.

13. O'Loughlin, D.; O'Halloran, M.; Moloney, B.M.; Glavin, M.; Jones, E.; Elahi, M.A. Microwave breast imaging: Clinical advances and remaining challenges. *IEEE Trans. Biomed. Eng.* **2018**, *65*, 2580–2590. [CrossRef] [PubMed]
14. Wang, F.; Arslan, T.; Wang, G. Breast cancer detection with microwave imaging system using wearable conformal antenna arrays. In Proceedings of the 2017 IEEE International Conference on Imaging Systems and Techniques (IST), Beijing, China, 20 October 2017; pp. 1–6.
15. Vasquez, J.T.; Turvani, G.; Dassano, G.; Casu, M.; Vipiana, F.; Joachimowicz, N.; Scapaticci, R.; Crocco, L.; Duchêne, B. Ongoing developments towards the realization of a microwave device for brain stroke monitoring. In Proceedings of the 2018 IEEE International Symposium on Antennas and Propagation & USNC/URSI National Radio Science Meeting, Boston, MA, USA, 13 July 2018; pp. 1139–1140.
16. Mobashsher, A.T.; Abbosh, A. On-site rapid diagnosis of intracranial hematoma using portable multi-slice microwave imaging system. *Sci. Rep.* **2016**, *6*, 37620. [CrossRef] [PubMed]
17. Saied, I.; Chandran, S.; Arslan, T. Integrated Flexible Hybrid Silicone-Textile Dual-Resonant Sensors and Switching Circuit for Wearable Neurodegeneration Monitoring Systems. *IEEE Trans. Biomed. Circuits Syst.* **2019**, *13*, 1304–1312. [CrossRef] [PubMed]
18. Bao, S.; Parvarthaneni, P.; Huo, Y.; Barve, Y.; Plassard, A.J.; Yao, Y.; Sun, H.; Lyu, I.; Zald, D.H.; Landman, B.A.; et al. Technology Enablers for Big Data, Multi-Stage Analysis in Medical Image Processing. In Proceedings of the 2018 IEEE International Conference on Big Data, Seattle, WA, USA, 10 Decembr 2019; pp. 1337–1346. [CrossRef]
19. Etminan, A.; Moghaddam, M. A Novel Global Optimization Technique for Microwave Imaging Based on the Simulated Annealing and Multi -Directional Search. In Proceedings of the 2018 IEEE International Symposium on Antennas and Propagation & USNC/URSI National Radio Science Meeting, Boston, MA, USA, 13 July 2018; pp. 1791–1792. [CrossRef]
20. Chen, D.; Hu, Y.; Cai, C.; Zeng, K.; Li, X. Brain big data processing with massively parallel computing technology: Challenges and opportunities. *Softw. Pract. Exp.* **2017**, *47*, 405–420. [CrossRef]
21. Siddiqui, F.; Amiri, S.; Minhas, U.I.; Deng, T.; Woods, R.; Rafferty, K.; Crookes, D. FPGA-based processor acceleration for image processing applications. *J. Imaging* **2019**, *5*, 16. [CrossRef]
22. Wong, K.K.L.; Fong, S.; Wang, D.; Kian, K.; Wong, L. Impact of advanced parallel or cloud computing technologies for image guided diagnosis and therapy. *J. X-ray Sci. Technol.* **2017**, *25*, 187–192. [CrossRef]
23. Ianni, M.; Masciari, E.; Mazzeo, G.M.; Mezzanzanica, M.; Zaniolo, C. Fast and effective Big Data exploration by clustering. *Future Gener. Comput. Syst.* **2019**, *102*, 84–94. [CrossRef]
24. Basha, S.A.K.; Basha, S.M.; Vincent, D.R.; Rajput, D.S. Challenges in Storing and Processing Big Data Using Hadoop and Spark. In *Deep Learning and Parallel Computing Environment for Bioengineering Systems*; Elsevier: Amsterdam, The Netherlands, 2019; pp. 179–187. [CrossRef]
25. Fu, G.S.; Levin-Schwartz, Y.; Lin, Q.H.; Zhang, D. Machine Learning for Medical Imaging. *J. Healthc. Eng.* **2019**. [CrossRef]
26. Essa, Y.M.; Hemdan, E.E.D.; El-Mahalawy, A.; Attiya, G.; El-Sayed, A. IFHDS: Intelligent Framework for Securing Healthcare BigData. *J. Med. Syst.* **2019**, *43*, 124. [CrossRef]
27. Edinburgh Compute and Data Facility Website. University of Edinburgh. Available online: http://www.ecdf.ed.ac.uk/ (accessed on 23 January 2020).
28. Li, Y.; Liu, P.; Li, Y.; Fan, H.; Su, P.; Peng, S.L.; Park, D.C.; Rodrigue, K.M.; Jiang, H.; Faria, A.V.; et al. ASL-MRICloud: An online tool for the processing of ASL MRI data. *NMR Biomed.* **2019**, *32*, e4051. [CrossRef] [PubMed]
29. Fahmi, F.; Nasution, T.H.; Anggreiny, A. Smart cloud system with image processing server in diagnosing brain diseases dedicated for hospitals with limited resources. *Technol. Health Care* **2017**, *25*, 607–610. [CrossRef] [PubMed]
30. Sarraf, S.; Ostadhashem, M. Big data application in functional magnetic resonance imaging using apache Spark. In Proceedings of the 2016 Future Technologies Conference (FTC), San Francisco, CA, USA, 7 December 2016; pp. 281–284.
31. Liu, B.; He, S.; He, D.; Zhang, Y.; Guizani, M. A Spark-Based Parallel Fuzzy c -Means Segmentation Algorithm for Agricultural Image Big Data. *IEEE Access* **2019**, *7*, 42169–42180. [CrossRef]
32. Cui, L.; Feng, J.; Zhang, Z.; Yang, L. High throughput automatic muscle image segmentation using parallel framework. *BMC Bioinform.* **2019**, *20*, 158. [CrossRef] [PubMed]

33. Munro, I.; GarcÍA, E.; Yan, M.; Guldbrand, S.; Kumar, S.; Kwakwa, K.; Dunsby, C.; Neil, M.; French, P. Accelerating single molecule localization microscopy through parallel processing on a high-performance computing cluster. *J. Microsc.* **2019**, *273*, 148–160. [CrossRef] [PubMed]
34. Qin, Y.; Wu, J.; Hu, Q.; Ghista, D.N.; Wong, K.K. Computational evaluation of smoothed particle hydrodynamics for implementing blood flow modelling through CT reconstructed arteries. *J. X-ray Sci. Technol.* **2017**, *25*, 213–232. [CrossRef]
35. Cai, G.; Wang, J.; Mei, X.; Zhang, W.; Luan, G.; Liu, X. Electroclinical semiology of the bilateral asymmetric tonic seizures observed in patients with supplementary sensorimotor area epilepsy confirmed by pre- and post-operative MRI. *J. X-ray Sci. Technol.* **2017**, *25*, 247–259. [CrossRef]
36. Liu, L.; Chen, W.; Nie, M.; Zhang, F.; Wang, Y.; He, A.; Wang, X.; Yan, G. iMAGE cloud: Medical image processing as a service for regional healthcare in a hybrid cloud environment. *Environ. Health Prev. Med.* **2016**, *21*, 563–571. [CrossRef]
37. Chard, R.; Madduri, R.; Karonis, N.T.; Chard, K.; Duffin, K.L.; Ordonez, C.E.; Uram, T.D.; Fleischauer, J.; Foster, I.T.; Papka, M.E.; et al. Scalable pCT Image Reconstruction Delivered as a Cloud Service. *IEEE Trans. Cloud Comput.* **2018**, *6*, 182–195. [CrossRef]
38. Roychowdhury, S.; Hage, P.; Vasquez, J. Azure-Based Smart Monitoring System for Anemia-Like Pallor. *Future Internet* **2017**, *9*, 39. [CrossRef]
39. Serrano, E.; Garcia-Blas, J.; Carretero, J. A Cloud Environment for Ubiquitous Medical Image Reconstruction. In Proceedings of the 2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom), Melbourne, Australia, 13 December 2018; pp. 1048–1055. [CrossRef]
40. Meng, B.; Pratx, G.; Xing, L. Ultrafast and scalable cone-beam CT reconstruction using MapReduce in a cloud computing environment. *Med. Phys.* **2011**, *38*, 6603–6609. [CrossRef] [PubMed]
41. Bao, S.; Damon, S.M.; Landman, B.A.; Gokhale, A. *Performance Management of High Performance Computing for Medical Image Processing in Amazon Web Services*; Zhang, J., Cook, T.S., Eds.; SPIE: Bellingham, WA, USA, 2016; Volume 9789, p. 97890Q. [CrossRef]
42. Tchagna Kouanou, A.; Tchiotsop, D.; Kengne, R.; Zephirin, D.T.; Adele Armele, N.M.; Tchinda, R. An optimal big data workflow for biomedical image analysis. *Inf. Med. Unlocked* **2018**, *11*, 68–74. [CrossRef]
43. Bond, E.J.; Li, X.; Hagness, S.C.; Van Veen, B.D. Microwave imaging via space-time beamforming for early detection of breast cancer. *IEEE Trans. Antennas Propag.* **2003**, *51*, 1690–1705. [CrossRef]
44. Kibria, S.; Samsuzzaman, M.; Islam, M.T.; Mahmud, M.Z.; Misran, N.; Islam, M.T. Breast phantom imaging using iteratively corrected coherence factor delay and sum. *IEEE Access* **2019**, *7*, 40822–40832. [CrossRef]
45. Been Lim, H.; Thi Tuyet Nhung, N.; Li, E.P.; Duc Thang, N. Confocal microwave imaging for breast cancer detection: Delay-multiply-and-sum image reconstruction algorithm. *IEEE Trans. Biomed. Eng.* **2008**, *55*, 1697–1704. [CrossRef]
46. Elahi, M.A.; O'Loughlin, D.; Lavoie, B.R.; Glavin, M.; Jones, E.; Fear, E.C.; O'Halloran, M. Evaluation of image reconstruction algorithms for confocal microwave imaging: Application to patient data. *Sensors* **2018**, *18*, 1678. [CrossRef]
47. Saied, I.; Arslan, T. Microwave Imaging Algorithm for Detecting Brain Disorders. In Proceedings of the 29th International Conference Radioelektronika (RADIOELEKTRONIKA), Kosice, Slovakia, Czech Republic, 16 April 2019; pp. 1–5.
48. Apache Hadoop YARN. Available online: https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html (accessed on 4 March 2020).
49. Apache Mesos. Available online: https://http://mesos.apache.org/ (accessed on 4 March 2020).
50. Zaharia, M.; Chowdhury, M.; Franklin, M.J.; Shenker, S. Spark: Cluster Computing with Working Sets. Technical Report. 2010. Available online: https://www.usenix.org/legacy/event/hotcloud10/tech/full_papers/Zaharia.pdf (accessed on 2 February 2020).
51. Apache Spark. Available online: https://Spark.apache.org/docs/latest/index.html (accessed on 18 September 2019).
52. Saied, I.; Arslan, T. Non-Invasive Wearable RF Device towards Monitoring Brain Atrophy and Lateral Ventricle Enlargement. *IEEE J. Electromagn. RF Microw. Med. Biol.* **2019**. [CrossRef]

53. Chew, K.M.; Yong, C.Y.; Sudirman, R.; Wei, S.T.C. Human brain modeling tumor detection in 2D and 3D representation using microwave signal analysis. In Proceedings of the ISCAIE 2018—IEEE Symposium on Computer Applications and Industrial Electronics, Penang, Malaysia, 29 April 2018; pp. 310–316. [CrossRef]
54. Chew, K.M.; Sudirman, R.; Mahmood, N.H.; Seman, N.; Yong, C.Y. Human Brain Microwave Imaging Signal Processing: Frequency Domain (S-parameters) to Time Domain Conversion. *Engineering* **2013**, *5*, 31–36. [CrossRef]