Article

Parallel Delay Multiply and Sum Algorithm for Microwave Medical Imaging Using Spark Big Data Framework

Rahmat Ullah *[ID] and Tughrul Arslan [ID]

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

Abstract: Microwave imaging systems are currently being investigated for breast cancer, brain stroke and neurodegenerative disease detection due to their low cost, portable and wearable nature. At present, commonly used radar-based algorithms for microwave imaging are based on the delay and sum algorithm. These algorithms use ultra-wideband signals to reconstruct a 2D image of the targeted object or region. Delay multiply and sum is an extended version of the delay and sum algorithm. However, it is computationally expensive and time-consuming. In this paper, the delay multiply and sum algorithm is parallelised using a big data framework. The algorithm uses the Spark MapReduce programming model to improve its efficiency. The most computational part of the algorithm is pixel value calculation, where signals need to be multiplied in pairs and summed. The proposed algorithm broadcasts the input data and executes it in parallel in a distributed manner. The Spark-based parallel algorithm is compared with sequential and Python multiprocessing library implementation. The experimental results on both a standalone machine and a high-performance cluster show that Spark significantly accelerates the image reconstruction process without affecting its accuracy.

Keywords: microwave imaging; mapreduce; medical imaging; Spark framework; parallel algorithm; high-performance computing

1. Introduction

Microwave imaging technology is a potential imaging method that aims to provide inexpensive, portable and wearable imaging devices. It was originally proposed for breast cancer detection [1] and was later extended for brain stroke or brain injuries [2]. The basic principle is the difference in the dielectric values (relative permittivity and conductivity) in healthy and malignant tissues [3,4]. Several wearable antennas are used as sensors that transmit ultra-wideband (UWB) signals to the imaging domain, and the reflected and transmitted data are stored. An imaging algorithm then uses these data to reconstruct images of the targeted object/tissue.

Microwave head imaging is a biomedical imaging application, and a number of microwave head imaging systems have been proposed targeting brain tumours [5], stroke [6] and recently neurodegenerative disease detection [7]. Microwave imaging systems are composed of two important components: the hardware and software. The hardware components include the sensors, the patient interface, the transceiver and switching system, and a PC. The software components include data preprocessing and an imaging algorithm. The reliability of these systems is substantially dependent on imaging algorithms [8].

In terms of imaging algorithms, there are typically two types, namely tomography and radar-based. Microwave tomography finds the location, shape or size of the malignant tissue by reconstructing the dielectric profile using electromagnetic properties such as permittivity and conductivity [9]. It is a well-known quantitative method and provides accurate reconstruction of the dielectric properties. The scattered field data are processed using a suitable inversion method to obtain the dielectric profile. The accuracy and rate

of convergence is improved by using nonlinear inversion techniques [10,11]. However, it is computationally complex and therefore cannot be used for real-time applications. On the other hand, radar-based imaging aims to localise strong reflected scatters in the area of interest. There are two types of radar-based beamformers: data-independent [12] and data-adaptive beamformers [13]. In data-independent beamforming algorithms, the coherent addition of signals is performed on the basis of an assumed propagation model. Data-adaptive algorithms, on the other hand, use scattered signals to estimate the propagation model and to then apply compensation factors based on the estimated channel model. Data-independent beamformers take less time in image reconstruction, while data-adaptive beamformers have excessive computational complexity but produce accurate results [14].

Radar-based beamforming techniques in microwave imaging aim to reconstruct a reflective map of the imaged tissues, where high contrast regions (malignant tissue) are prominently detected. The basic image reconstruction method in radar-based microwave imaging is the delay and sum (DAS) beamformer [15]. This technique thoroughly delays the collected time domain signals and sums it up to reconstruct an image. Several improvements have been made to the DAS beamformer, including delay multiply and sum (DMAS) [16], improved delay and sum (IDAS) [15], and channel-ranked delay and sum (CR-DAS) [17]. These improvements to the DAS have primarily focused on coherence between reflected or transmitted radar signals or on tackling path dependency. A recent study compared these radar-based algorithms using clinical data for breast cancer detection. The results demonstrated that DMAS produces more accurate and high-resolution results [18]. It reduces clutter and background noise that leads to more accurate localisation of the tumour in resulting images [19].

The computational complexity of DAS is $O(M)$. The DMAS gains better resolution and contrast at the cost of increasing the computational complexity to $O(M^2)$ [16]. The advantages of using DMAS over DAS has been proven in [20,21]. The iterative variant of DMAS was proposed in [22], where the background clutter was removed in the resulting images. Due to its advantages, DMAS is used in many research works instead of DAS [22]. The downside of using DMAS in comparison to DAS is the significant increase in computational load [23].

Hadoop and MapReduce are commonly used in commerce, social media and video streaming. However, they are not widely integrated into the healthcare sector. For example, a pipeline for functional magnetic resonance imaging (fMRI) was proposed using the PySpark [24]. In the proposed pipeline, the template method is used to extricate brain networks from fMRI data. The pipeline is four times faster due to in-memory data processing in parallel compared to sequential execution. Similarly, Spark was used for genomics sequencing to reduce the processing time [25]. Spark cloud computing platform was used for parallel implementation of the MAX-MIN Ant System algorithm [26]. Experimental results showed significant improvement in speed for a large number of ants. Similarly, the most commonly used K-means clustering algorithm was improved in terms of scalability, accuracy and effectiveness by using a Tabu Search Strategy and Spark framework [27]. Computational experiments indicated that the proposed parallel algorithm yields better results in terms of quality, stability and efficiency than the K-means algorithm of SparkMLLib.

Field programmable gates arrays [28] and graphics processing unit-based parallel approaches are mostly used to decrease computation time [29]. A GPU-based implementation of CMI algorithms was presented for breast cancer detection [30]. Although these approaches significantly increase processing speed, they are non-scalable and costly. An alternative approach is to use high-performance computing (HPC) or a Cloud platform such as Google Cloud or Amazon Web Services (AWS) to improve the efficiency of imaging modalities. HPC has been used for quantitative imaging such as tomography [31,32], where the data obtained for human head imaging was wirelessly transferred to a remote HPC for postprocessing. Big data frameworks such as Hadoop and Spark have recently gained attention to make the most of these platforms. Therefore, these frameworks need to be considered as accelerators for medical imaging algorithms [33].

In our previous work [34], a parallel microwave image reconstruction algorithm (PMIR) was proposed based on the Spark framework. The proposed algorithm was tested on a high-performance computing (HPC) cluster and Google Cloud platform. The results demonstrated that the parallel is about 128 % faster than its sequential version. The objective of this work is to investigate the use of big data frameworks, especially Spark, to improve the efficiency of the DMAS algorithm. The parallel version of the DMAS algorithm is compared with the sequential version as well as Python built-in multiprocessing library. The increase in processing speed is evaluated on both a standalone PC and on HPC.

The main contributions of this paper are as follows.

- We examine the potential gains in performance and scalability of DMAS using big data framework.
- We use parallel implementation of the DMAS algorithm using Pyspark and Python multiprocessing library.
- We evaluate the sequential and parallel versions of DMAS on a standalone PC and on an HPC cluster for efficiency.

The proposed parallel version of the DMAS algorithm is implemented in Python programming language and deployed using PySpark API for Apache Spark. The algorithm is deployed on both the standalone and cluster modes. The computational results demonstrate that the parallel DMAS fully inherits the parallelism of the DMAS algorithm, resulting in an efficient reconstruction of images.

2. Principle of Radar-Based Algorithms

The mechanism of the microwave imaging is based on the difference in the dielectric property (relative permittivity and conductivity) for healthy and malignant tissues [35]. The dielectric property is an inherent characteristic of the tissue that characterises the transmission, absorption and reflection of electromagnetic energy. An array of antennas is used to penetrate the targeted tissue, and the reflected and transmitted signals are stored. These signals are used as input to imaging algorithms. The most common microwave imaging algorithm used in medicine is CMI. These methods are usually referred to as radar-based methods. It uses the reflected or transmitted signals from the imaged object to reconstruct the resulting image using different beamformers [36]. These beamformers calculate the energy from a focal point and integrate the information to reconstruct an energy map.

The basic principle of a radar-based algorithm is to thoroughly delay and sum the signals to reconstruct the image. Each antenna in the microwave imaging system transmits a signal to each focal point in the imaging domain, and the reflected and transmitted signals are stored. The time delay is calculated for each measured signal from the antenna to the focal point and back to the transmitting antenna (monostatic mode) or and receiving antenna (multistatic mode). In time delay calculation, the propagation speed of the signals depends on the relative permittivity of the medium. The signals are thoroughly delayed and summed to calculate energy. When the energy distribution of the whole imaging area for each antenna is obtained, a 2D image is formed.

DMAS is an extended version of the DAS algorithm. It produces more accurate and high-resolution results. Similar to DAS, the reflected or transmitted signals are first aligned and then summed to reconstruct the image. The signal energy corresponding to each pixel in the resulting images is equal to the square of the sum of the product of the signal. DMAS increases the sample size by multiplying the signals in pairs before summation, as shown in Figure 1, significantly increasing its computation time.
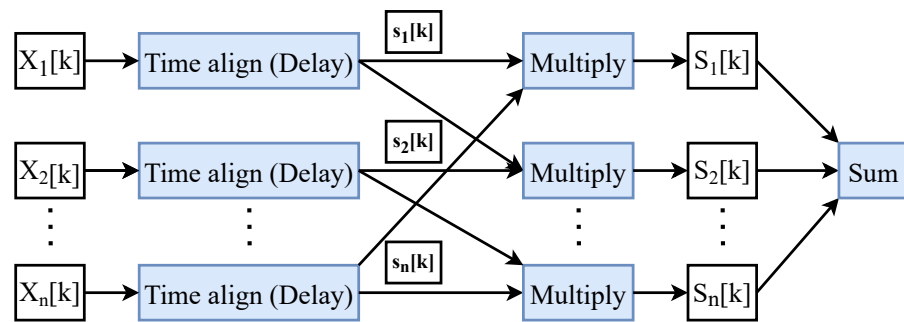
Figure 1. Block diagram for DMAS. The reflected signal is thoroughly delayed, multiplied in pairs and summed to calculate energy. The process is repeated for all input signals and integrated as a matrix of pixel values for image reconstruction.

The DMAS algorithm performs a correlation process for each focal point to obtain data with a high degree of similarity. The improvement in the accuracy of the resulting images is due to the substantial increase in the sample size. The observations are shifted according to focal points in the imaging domain using delay values calculated as follows:

$$\tau(r_0) = \frac{|t_x - r_0| + |r_x - r_0|}{C_0 / \sqrt{\varepsilon_r}} \tag{1}$$

where $\tau(r_0)$ is the delay factor for a focal point r_0 for the signal transmitted by an antenna located at t_x and receiving antenna located at r_x, C_0 denotes the speed of light in space and ε_r is the relative permittivity of the human head.

Assume that there are M observations ($S_i[k]$; $i = 1..M$, $k = 1..L$), each with L samples, shifted using (1). Then, the output of DAS is as follows:

$$I_{DAS} = \sum_{k=1}^{L} \left(\sum_{i=1}^{M} S_i[k] \right)^2 \tag{2}$$

where $S_i[K]$ is the *i*th delayed signal

On the other hand, the output of DMAS is as follows:

$$I_{DMAS} = \sum_{k=1}^{L} \left(\sum_{i=1}^{M} \sum_{\substack{j=1 \\ j \neq 1}}^{M} S_i[k] S_j[k] \right) \tag{3}$$

where M is the total number of received signals and $S_i[K]$ and $S_j[K]$ are delayed signals according to the positions of the transmitter and receiver antennas.

DAS requires M summations and one multiplication for M observations to estimate the desired value. On the other hand, the DMAS method requires M^2 multiplications and M^2 summations, increasing the computations and requiring M times longer to reconstruct an image.

A faster version of DMAS (FDMAS) was proposed in [20], where energy is calculated as follows:

$$I_{FDMAS} = \sum_{k=1}^{L} \left(\left(\sum_{i=1}^{M} S_i[k] \right)^2 - \sum_{i=1}^{M} S_i[k]^2 \right) \tag{4}$$

Although the outputs of DMAS and FDMAS are mathematically equal, the FDMAS is best suited for parallel implementation due to its ability to update the results without redundant calculation.

3. Overview of Spark MapReduce

MapReduce was introduced by Google in 2004. MapReduce aimed to process big data by leveraging the capacity of commodity hardware in a distributed manner [37]. The key benefit of the MapReduce programming model is that it does not require users or developers to be experts in parallel or distributed systems in order to exploit its capabilities. Furthermore, MapReduce is highly scalable and can process data up to terabytes.

Apache Spark is a big data framework with built-in modules for streaming, structured query language, machine learning and graph processing. It is 100 times faster than Hadoop, another big data framework that uses the MapReduce Programming paradigm [38]. Apache Spark, compared to Hadoop, achieved higher performance due to in-memory computations by introducing the resilient distributed datasets (RDDs). The RDD are read-only sets of data, split into logical partitions and spread over different machines in a cluster, being operated in parallel. These logical partitions of data are the main drivers for parallelism. Spark allows for two main operations, as shown in Figure 2, to operate on these partitions: transformations (map), which create new RDDs and create dependency among them, and actions (reduce), which return intermediate results after any operation performed on the data.
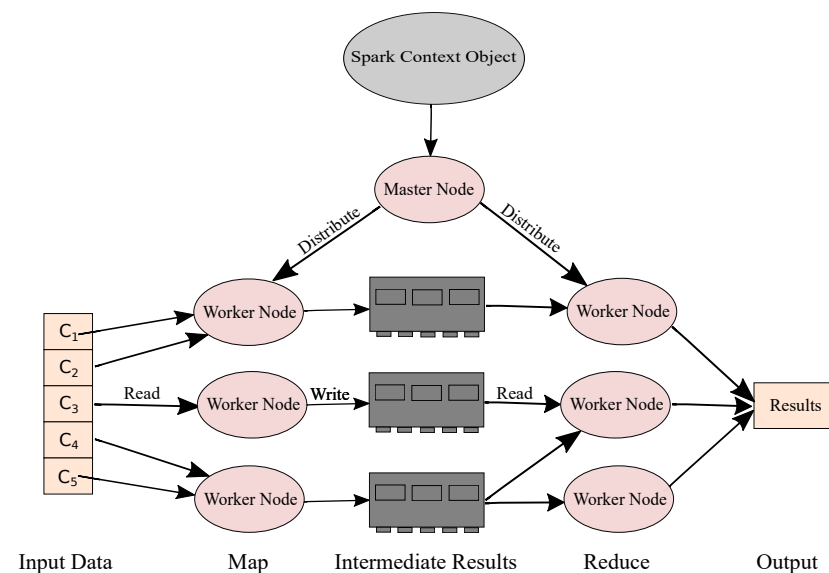
Figure 2. Working principle of the Spark framework. The input data are distributed on multiple machines (workers) and operated in parallel. The reduce operation retrieve the intermediate results and integrate it for the final results (output).

Each RDD contains information about dependency with other RDDs, computation needed, accessible node and metadata regarding the RDD scheme [39]. Apache Spark is based on a master–slave architecture. It divides a cluster into a master and several worker nodes. The master node deals with resource allocation, task scheduling and error management. The worker nodes are responsible for map and reduce operations in parallel.

4. Parallel Design and Implementation of DMAS Algorithm

In this paper, PySpark is used for the parallel implementation of DMAS. PySpark is an interface that gives access to the Spark framework using the Python programming language. It employs the RDDs to increase the execution speed by performing in-memory computations.

The iterative procedure of the FDMAS makes it suitable to be implemented using a big data framework such as Spark. The parallelism and dependency among different processes in the DMAS algorithm was first identified. There are multiple MapReduce operations for the parallel scheme of the DMAS algorithm. The entire input data are uploaded to the

shared memory accessible to the master and all worker nodes. The master node converts these data into RDDs, and each map and reduce operation executes the required operation on the data block. These MapReduce operations are associated with one computing unit. When the entire data set is split and executed simultaneously, the execution time is expected to reduce.

The two most computationally intensive parts of the DMAS algorithm can be parallelised: delay and pixel value calculation. Since the delay values in the DMAS algorithm need to be calculated once, its parallel executions do not affect the overall performance [34]. The most computationally and memory-intensive part is the multiplication of signals in pairs and the summation of all antennas for each focal point in the resulting images. In this part, each process is independent and can be executed in parallel by caching intermediate results in memory across iterative computations.

4.1. Parallel Design of DMAS Using Spark

The multiplication and summation steps of the DMAS are well suited for map and reduce operations. The steps of DMAS based on the MapReduce paradigm and Spark framework are as follows:

1. Data Parallelism: Retrieve and store the sensor data. Let the number of antennas used for data collection be M; an array of the same size in monostatic mode or $M(M-1)/2$ size in multistatic mode needs to be created. Use the Spark context object to parallelise the data and to convert it to RDDs. Distribute these RDDs to the worker nodes. This also includes broadcasting an empty matrix to store pixel values to each worker node. These RDDs are input to the next step.
2. Map Operations: Since pixel values are calculated using the same equation (Equation (4) in this work), they can be calculated in parallel. The Map operation is used to multiply the signals in pairs and to sum it up to calculate pixel values.
3. Reduce Operations: Integrate the subsets of pixel values using the "reduce" operation.
4. Iterate: Update the empty matrix of pixel values according to the values obtained from the "reduce" operation. Iterate the map and reduce tasks until all signals are processed.

4.2. PDMAS Implementation Using Spark

The proposed PDMAS is implemented using PySpark API for Spark. Algorithm 1 provides the pseudo-code for parallel implementation of the DMAS algorithm using Spark. The master node reads the input data and antennas location in Cartesian coordinates. These antenna location are used to estimate the signal propagation time using Equation (1). The estimated delay values along with the input data are converted into RDDs and distributed to all workers nodes to accelerate each calculation. Similarly, an empty matrix is created to store the subsets of pixel values and is broadcasted. This matrix is shared by different worker nodes. The worker nodes calculate the subsets of pixel values using the map operation from the MapReduce programming model. These subsets of pixel values being computed in parallel are integrated using the reduce operation and returned to the master node. Finally, the master node transforms the matrix of pixel values into an image using a simple Python function and and the results are stored.

Algorithm 1 PDMAS algorithm using Spark.

Input: RF Sensors Data
Output: Image Reconstruction
   Master Node:

 1: Read input data along with antenna locations
 2: Estimate $\tau_r(x, y)$: signal propagation time for all focal points in the imaging domain using Equation (1)
 3: Create a matrix I_{MN} for storing pixel values
 4: Convert the input data and delay values into RDDs and broadcast it to each worker node
   Worker Node:

 5: **for** subset of t_a < range **do**
 6: **for** subset of r_a < range **do**
 7:   $I_{mn} \leftarrow I_{mn} + S_{ta,ra}(\tau_r) \times S_{ta,ra+1}(\tau_r)$: Calculate the subset of pixel values in parallel using Map operation
 8: **end for**
 9: **end for**
10: Integrate I_{mn} to I_{MN} using Reduce operation
11: Return I_{MN} to master node
   Master Node
12: Convert I_{MN} into image and save.

4.3. DMAS Using Python Multiprocessing

The Pyspark version is compared with the Python built-in multiprocessing library [40]. For this purpose, the SharedArray module [41] was used to create a Numpy array to make it accessible to multiple processes at the same time. A SharedArray for input data was created on the master node. The process class of the multiprocessing library was used. The process was instantiated with two arguments: the matrix of input signal and a function for pixel value calculation using Equation (4). Finally the processes were joined using the join function, and the resulting matrix was stored.

5. Experiments

The experiments were performed on data obtained for the detection of brain atrophy due to Alzheimer's disease.

5.1. Imaging System Setup

A wearable and portable device in a hat-like shape proposed in our previous work [42] was used. The device contains an array of six flexible antennas created using thin polyethylene terephthalate. The operating frequency of these antennas are 800 MHz to 2.5 GHz. The antennas were first created in CST studio suite [43]. A vector network analyzer was used to transmit and receive the signals. The generated signals were sent to a PC for further processing. The proposed wearable device is radar-based and used planer monopole antennas due to its low cost.The use of only six antennas makes the device lightweight; however, it affects the spatial resolution of the reconstructed brain images.

The sensors were made using A8 silicon rubber with a relative permittivity of 2.99. An SMA was attached to the sensor using epoxy. The design of the sensor is based on a rectangular planar monopole antenna structure. The substrate of the antenna is made using a flexible textile, and the conductive part of the antenna is made using a 6 mm thick material. It operates from 1 to 4.2 GHz and exhibits a radiation pattern of 6 dB front-to-back ratio at 2.5 GHz. Further information about antenna can be found in [44].

Furthermore, a flexible, low-cost switching system was developed that consists of SMA connectors and one-pole and one-throw (1P1T) switches. The sensors and switching systems were then combined into a wearable device. The total weight of the device (components include a hat, two switching circuits and six sensors) is 291.7 g. The insertion loss of the proposed system is -3.9 dB, which is slightly larger due to the flexibility of the device. Further information about the switching systems, VNA and a PC can be found in [42].

5.2. Phantom Used

Phantoms are often used to validate the antennas and imaging algorithms as it avoids exposure towards a living human. To validate the results of the proposed parallel algorithm, Lamb brain phantom was used. These phantoms were used to mimic different stages of brain atrophy caused by Alzheimer's disease.

5.3. Data Acquisition

According to the configuration of the imaging setup, an imaging algorithm needs to be used to reconstruct an image from measurements of the reflected or transmitted fields. The configuration of the imaging system can either be monostatic or multistatic, where one or more antennas are used as transmitters and receivers. In monostatic configuration, a single antenna serves as both transmitter and receiver. In multistatic configuration, each antenna transmits the microwave signal, and the scattered signals are captured by all other antennas. This procedure is repeated for all antennas until all signal samples are obtained. Although, the multistatic approach is more complex compared to the monostatic approach, it provides detailed information about the imaged area/object and hence improves the accuracy.

In this work, an array of six monopole directional antennas were placed around the phantom at equal distances. To achieve good penetration, the antennas were separated from the absorber using a 5 mm thick foam. The dielectric value of the foam is similar to that for air to ensure the integrity of antennas. The phantoms were penetrated in a way where one antenna transmitted the signal and where reflected and transmitted signals were captured by all other antennas. The experiments were repeated for each antenna to capture both the reflected signals (S_{11}) and transmitted signals (S_{21}). However, only the reflected signals are used for image reconstruction. The total number of 1001 frequency points were stored for each antenna. These points were organised as a matrix for each case.

Similarly, an empty skull was penetrated with the same configuration and the reflected data were stored. These data were used as a reference signal to remove strong reflections from the skin and skull.The matrix of signals was then converted to a time domain using Fourier transform. The resulting signals were used as input to the imaging algorithm for image reconstruction.

5.4. Experimental Setup

The experiments were performed on both a standalone PC and a high-performance computing cluster. For the cluster mode, Eddie (see http://www.ecdf.ed.ac.uk, accessed on 22 February 2021), the University of Edinburgh high-performance computing cluster was used. It is the third iteration of the University's compute cluster and consists of 7000 Intel Xeon cores. Each compute node has up to 3 TB of memory. It is suitable to increase the processing speed by running the task (algorithm) in parallel. It allows for breaking the task into several more easily addressable subtasks, each of which can be run on a separate CPU in parallel. It runs Open Grid scheduler and Scientific Linux 7. An overview of the Eddie architecture is shown in Figure 3.

The login node allows users to either submit batch jobs or to start an interactive session, where the number of cores and memory per core can be specified. The compute node has local memory and is connected to shared memory called an Eddie distribute file system. Users can store the input data in the shared memory. All compute nodes are

also interconnected with a 10 Gbps Ethernet network. It uses an Open Grid Scheduler for memory allocations and job scheduling.
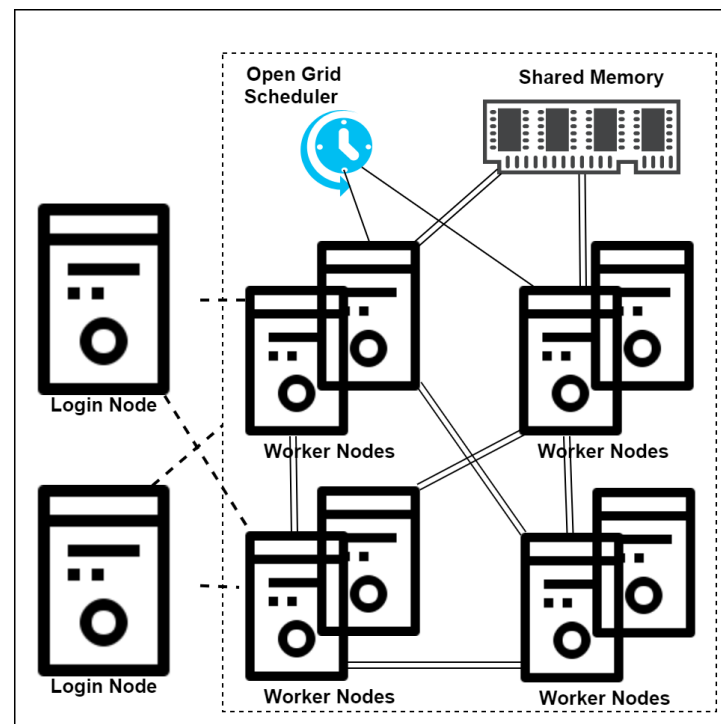
Figure 3. Overview of the Eddie architecture [34]. The login node has limited memory and allows users to submit batch jobs or to start an interactive session. The submitted jobs run on the assigned worker nodes connected through a 10 Gbps Ethernet network shown by double lines.

Using the MapReduce programming model, the DMAS algorithm was designed and implemented using Pyspark API. A sequential version of DMAS was implemented in Python programming language. Similarly, a parallel version was implemented using Python multiprocessing library. The efficiency of all these approaches were tested on a standalone PC and Eddie. The configuration for both the standalone PC and the Eddie cluster can be found in Table 1. After setting up the environment for PySpark and Python, PySpark was launched with a configuration with one master and multiple workers node. Each node has the same configuration, as shown in Table 1.

Table 1. Experimental setup for both stand-alone and Eddie cluster.

Item	Standalone	Eddie
CPU	Intel(R) Core(TM) i7-3770 CPU @ 3.40 GHz (8 CPUs), 3.4 GHz	Intel® Xeon® Processor E5-2630 v3 (2.4 GHz)
Memory	16 GB	16 GB
Operating System	Windows 10 Education 64-bit	Scientific Linux 7
JDK Version	1.8.0	1.8.0
Spark Version	2.4.1	2.4.1
Python Version	3.7.5	3.7.5

For the performance evaluation on a standalone PC, all versions were run three times, and wall time was noted. Similarly, an interactive session on Eddie was started with four nodes, where one node acts as a master and the other three nodes act as workers. The master node reads and generates RDDs from the input data and antenna positions stored on the Eddie distributed file system. These RDDs are broadcasted to all worker nodes, along with an empty matrix to store pixel values. The worker nodes calculate subsets of

pixel values based on the algorithm discussed in Section 4.1 and return the results to the master node using the MapReduce programming model. The master node transforms the resulting matrix into an image using Python built-in functions and saves the results to the Eddie storage. An example image generated using DMAS and DAS for a lamb brain phantom can be found in Figure 4.

(**a**) (**b**) (**c**)
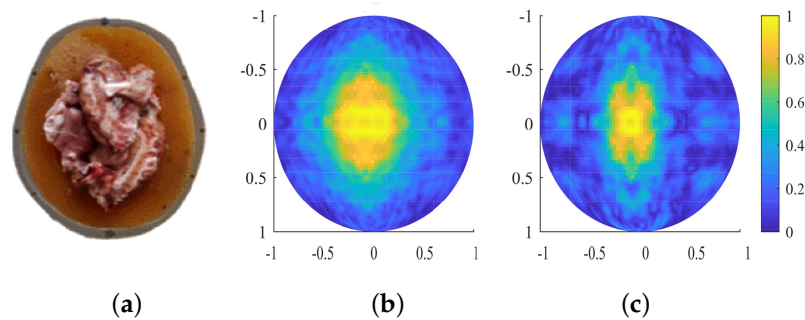
Figure 4. Experimental results for (**a**) lamb brain phantom using (**b**) DMAS and (**c**) DAS algorithm. It can be noted that DMAS produce accurate results compared to DAS.

It can be noted that the DMAS reconstructs the brain volume more accurately compared to DAS. It is important to mention that the data were preprocessed before image reconstruction. The signal obtained for lamb brain phantom were subtracted from reference signals obtained from an empty skull model to remove strong reflections from skull.

5.5. Results Analysis and Validation

To validate the results of the proposed Spark-based parallel DMAS algorithm, the results are compared in terms of processing speed with both sequential and Python multiprocessing library. These three versions of algorithm were deployed on both a standalone PC and on Eddie and were run for three times, and the average wall time was noted. The computational results for sequential, Pyspark-based, and Python multiprocessing versions on both the standalone PC and on Eddie can be found in Figure 5. It can be observed that the Pyspark version is 34.12 and 43.2 times faster than sequential implementation of DMAS on a standalone PC and on Eddie, respectively. The Pyspark version also outperforms the Python multiprocessing library and is 4.62 times faster on the standalone PC and 6.72 times faster on Eddie. It can be observed that the Python multiprocessing library also improves the speed significantly compared to the sequential version; however, Pyspark achieves better speed due to in-memory processing.

On the other hand, the cluster mode significantly improves the speed; however, there is a slight difference in performance due to data retrieval from the shared memory.

A comparison with the MIR algorithm, based on DAS, proposed in our previous work [34], is also performed to further validate these results. The results can be found in Figure 6. It can be observed that the parallel DMAS takes sightly longer due to M^2 multiplications. This comparison aims to determine the effectiveness of using the Spark framework for health big data processing such as medical image reconstruction.

It can also be observed that the Spark MapReduce framework significantly improves the efficiency of these imaging algorithms. One of the downsides of the proposed algorithm is that it retrieves the data into a memory disk on every cycle to calculate pixel value, which results in considerable disk input/output. A potential solution is to copy the input data and algorithm to the computational nodes.
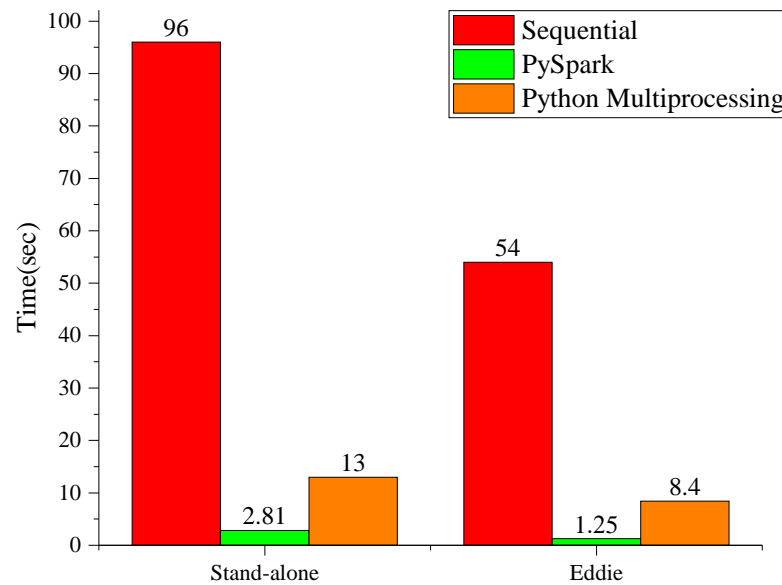
Figure 5. Experimental results of DMAS and its parallel versions using Pyspark and Python multiprocessing on a standalone PC and on Eddie. The results shows that both Spark and Python multiprocessing reduces processing time significantly; however, Spark outperforms Python.
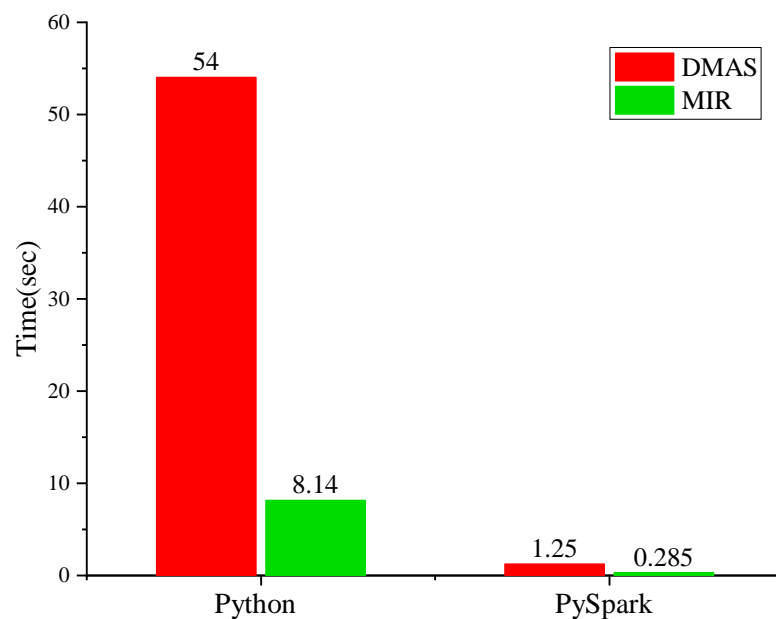
Figure 6. Comparison of the DMAS and MIR algorithms based on DAS proposed in our previous work [34]. The Pyspark improves the performance for both algorithms; however, DMAS takes slightly longer due to multiplication of signal in pairs.

6. Discussion

Microwave imaging for biomedical imaging applications has gained much attention over the past few decades. It has many advantages such as fast diagnosis and non-ionizing radiation and is emerging as a potential imaging modality compared to the existing imaging modalities such as magnetic resonance imaging (MRI), computed tomography (CT) and ultrasound. The reliability of microwave medical imaging system is largely dependent on the efficiency and accuracy of preprocessing and imaging algorithms. The imaging algorithm can be divided into two types: tomography-based algorithm and radar-based imaging algorithm. The tomography-based methods reconstruct the electrical properties

of the imaging area by solving an ill-posed inverse problem and is therefore complex and time-consuming. Conversely, radar-based methods use the reflected or transmitted signals to find the position/size of the targeted object, such as breast tumour, brain stroke or brain atrophy. Among the two types of radar-based beamformers, data-independent beamformers are faster at image restoration than data-adaptive beamformers; however, the latter yields better results.

Several data-independent and data adaptive algorithms are utilised for image reconstruction. However, DAS and its variations are mostly used due to their robust performance and simplicity. The most commonly used extension of DAS beamformer is DMAS, which provides improved contrast and resolution in the reconstructed images by trading the complexity of computation [45]. The computation is further increased for the multistatic approach; however, it improves the accuracy. This work uses Spark as an accelerator to improve the efficiency of the DMAS algorithm. The most computationally intensive part of the algorithm is first identified. The algorithm is implemented using a MapReduce operation, where subsets of pixel values are calculated on different worker nodes in parallel and retrieved to a master node. The Spark-based parallel implementation is compared with a sequential and Python built-in multiprocessing library implementation. The comparison are performed on a standalone PC and in cluster mode.

The experimental results shows that the Spark-based parallel algorithm performs an average of 34 and 43.2 times faster than the sequential one on a standalone PC and on HPC, respectively. Similarly, it outperforms Python multiprocessing due to in-memory processing. The comparison in terms of processing speed for all three versions of DMAS can be found in Figure 7. The comparison indicates that the Pyspark version improves the speed significantly compared to both sequential and multiprocessing version of DMAS.
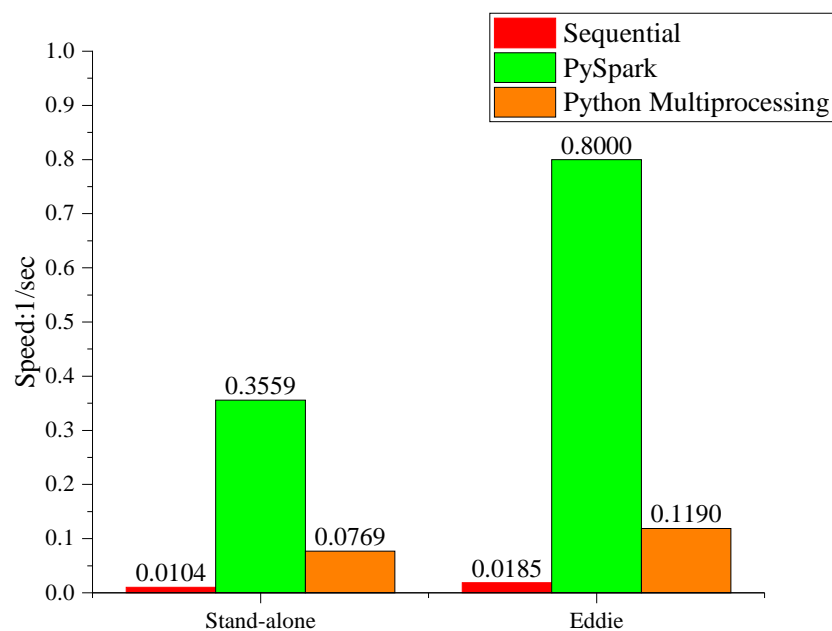
Figure 7. Processing speed (1/second) for sequential DMAS and its parallel versions using Pyspark and Python multiprocessing on a standalone PC and on Eddie. The results show that the Pyspark-based algorithm performs faster.

The downside of the proposed Spark-based algorithm is its retrieval of data to a memory disk on every cycle for image reconstruction, resulting in substantial disk input/output. The results revealed that the PDMAS achieves significant speed improvements on image reconstruction in parallel. The Spark-based results and comparison indicated that big data frameworks have the potential to improve the efficiency of microwave imaging algorithms. The master–slave architecture along with the Spark framework is generalisable and can be used for tomography-based algorithms as well as data-adaptive algorithms.

7. Conclusions

This paper presents the design and implementation of a parallel DMAS algorithm using the Spark framework on a standalone PC and on an Eddie cluster. The Spark framework distributes the data needed for image reconstruction across several worker nodes. The most computationally intensive part of the DMAS is computed in parallel on separate worker nodes and then retrieved to a master node to aid faster computation. The parallel version of DMAS is compared with sequential and multiprocessing-based DMAS. Computational experiments demonstrate the superiority of the proposed algorithm over the widely used sequential version of DMAS as well as on Python multiprocessing. The proposed algorithm is generalisable and can be implemented on any cluster with a master–slave architecture. The algorithm is best suited to being deployed on the cloud. This paper shows the potential use of big data frameworks such as Spark as an accelerator for medical image processing.

The proposed algorithm is tested on homogeneous nodes. In the future, the proposed parallel algorithm will be tested on heterogeneous computing nodes. These big data frameworks can also be utilised for tomography-based and data-adaptive algorithms. The computationally intensive and independent process in these algorithms can be parallelised using Spark to improve their performance.

Author Contributions: Conceptualisation, R.U.; methodology, R.U.; software, R.U.; validation, R.U.; formal analysis, R.U. and T.A.; writing—original draft preparation, R.U.; writing—review and editing, R.U. and T.A.; supervision, T.A.; project administration, T.A. Both authors have read and agreed to the published version of the manuscript.

Abbreviations

The following abbreviations are used in this manuscript:

HPC	High-Performance Computing
GCP	Google Cloud Platform
RF	Radio Frequency
AWS	Amazon Web Services
DAS	Delay and Sum
DMAS	Delay Multiply and Sum
PDMAS	Parallel Delay Multiply and Sum
RDD	Resilient Distributed Datasets

References

1. Nikolova, N.K. Microwave imaging for breast cancer. *IEEE Microw. Mag.* **2011**, *12*, 78–94. [CrossRef]
2. Scapaticci, R.; Di Donato, L.; Catapano, I.; Crocco, L. A feasibility study on microwave imaging for brain stroke monitoring. *Prog. Electromagn. Res.* **2012**, *40*, 305–324. [CrossRef]
3. Saied, I.; Bashri, M.; Arslan, T.; Smith, C.; Chandran, S. Dielectric Measurements of Brain Tissues with Alzheimer's Disease Pathology in the Microwave Region. In Proceedings of the 2019 IEEE International Symposium on Medical Measurements and Applications (MeMeA), Istanbul, Turkey, 26–28 June 2019; pp. 1–6.
4. Cheng, Y.; Fu, M. Dielectric properties for non-invasive detection of normal, benign, and malignant breast tissues using microwave theories. *Thorac. Cancer* **2018**, *9*, 459–465. [CrossRef]
5. Rezaeieh, S.A.; Zamani, A.; Abbosh, A. 3-D wideband antenna for head-imaging system with performance verification in brain tumor detection. *IEEE Antennas Wirel. Propag. Lett.* **2014**, *14*, 910–914. [CrossRef]
6. Alqadami, A.S.; Bialkowski, K.S.; Mobashsher, A.T.; Abbosh, A.M. Wearable electromagnetic head imaging system using flexible wideband antenna array based on polymer technology for brain stroke diagnosis. *IEEE Trans. Biomed. Circuits Syst.* **2018**, *13*, 124–134. [CrossRef]

7.   Saied, I.; Arslan, T.; Chandran, S.; Smith, C.; Spires-Jones, T.; Pal, S. Non-Invasive RF Technique for Detecting Different Stages of Alzheimer's Disease and Imaging Beta-Amyloid Plaques and Tau Tangles in the Brain. *IEEE Trans. Med Imaging* **2020**, *39*, 4060–4070. [CrossRef] [PubMed]

8.   Islam, M.T.; Islam, M.T.; Samsuzzaman, M.; Kibria, S.; Chowdhury, M.E. Microwave Breast Imaging Using Compressed Sensing Approach of Iteratively Corrected Delay Multiply and Sum Beamforming. *Diagnostics* **2021**, *11*, 470. [CrossRef]

9.   Semenov, S.Y.; Corfield, D.R. Microwave tomography for brain imaging: Feasibility assessment for stroke detection. *Int. J. Antennas Propag.* **2008**, *2008*. [CrossRef]

10.   Fedeli, A.; Schenone, V.; Randazzo, A.; Pastorino, M.; Henriksson, T.; Semenov, S. Nonlinear S-Parameters Inversion for Stroke Imaging. *IEEE Trans. Microw. Theory Tech.* **2020**. [CrossRef]

11.   Bisio, I.; Estatico, C.; Fedeli, A.; Lavagetto, F.; Pastorino, M.; Randazzo, A.; Sciarrone, A. Variable-exponent Lebesgue-space inversion for brain stroke microwave imaging. *IEEE Trans. Microw. Theory Tech.* **2020**, *68*, 1882–1895. [CrossRef]

12.   Byrne, D.; O'Halloran, M.; Glavin, M.; Jones, E. Data independent radar beamforming algorithms for breast cancer detection. *Prog. Electromagn. Res.* **2010**, *107*, 331–348. [CrossRef]

13.   Xie, Y.; Guo, B.; Xu, L.; Li, J.; Stoica, P. Multistatic adaptive microwave imaging for early breast cancer detection. *IEEE Trans. Biomed. Eng.* **2006**, *53*, 1647–1657. [CrossRef] [PubMed]

14.   Byrne, D.; Craddock, I.J. Time-domain wideband adaptive beamforming for radar breast imaging. *IEEE Trans. Antennas Propag.* **2015**, *63*, 1725–1735. [CrossRef]

15.   Klemm, M.; Craddock, I.; Leendertz, J.; Preece, A.; Benjamin, R. Improved delay-and-sum beamforming algorithm for breast cancer detection. *Int. J. Antennas Propag.* **2008**, *2008*. [CrossRef]

16.   Lim, H.B.; Nhung, N.T.T.; Li, E.P.; Thang, N.D. Confocal microwave imaging for breast cancer detection: Delay-multiply-and-sum image reconstruction algorithm. *IEEE Trans. Biomed. Eng.* **2008**, *55*, 1697–1704. [PubMed]

17.   O'Halloran, M.; Glavin, M.; Jones, E. Improved Confocal Microwave Imaging of the breast using path-dependent signal weighting. In Proceedings of the 2011 XXXth URSI General Assembly and Scientific Symposium, Istanbul, Turkey, 13–20 August 2011; pp. 1–4.

18.   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]

19.   Elahi, M.A.; Lavoie, B.; Porter, E.; Olavini, M.; Jones, E.; Fear, E.; O'Halloran, M. Comparison of radar-based microwave imaging algorithms applied to experimental breast phantoms. In Proceedings of the 2017 XXXIInd General Assembly and Scientific Symposium of the International Union of Radio Science (URSI GASS), Montreal, QC, Canada, 19–26 August 2017; pp. 1–4.

20.   KaramFard, S.S.; Asl, B.M. Fast delay-multiply-and-sum beamformer: Application to confocal microwave imaging. *IEEE Antennas Wirel. Propag. Lett.* **2019**, *19*, 14–18. [CrossRef]

21.   Islam, M.T.; Samsuzzaman, M.; Kibria, S.; Misran, N.; Islam, M.T. Metasurface loaded high gain antenna based microwave imaging using iteratively corrected delay multiply and sum algorithm. *Sci. Rep.* **2019**, *9*, 17317. [CrossRef] [PubMed]

22.   Reimer, T.; Solis-Nepote, M.; Pistorius, S. The Application of an Iterative Structure to the Delay-and-Sum and the Delay-Multiply-and-Sum Beamformers in Breast Microwave Imaging. *Diagnostics* **2020**, *10*, 411. [CrossRef]

23.   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]

24.   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, 6–7 December 2016; pp. 281–284. [CrossRef]

25.   Guo, R.; Zhao, Y.; Zou, Q.; Fang, X.; Peng, S. Bioinformatics applications on apache spark. *GigaScience* **2018**, *7*, giy098. [CrossRef]

26.   Wang, L.; Wang, Y.; Xie, Y. Implementation of a parallel algorithm based on a spark cloud computing platform. *Algorithms* **2015**, *8*, 407–414. [CrossRef]

27.   Lu, Y.; Cao, B.; Rego, C.; Glover, F. A Tabu Search based clustering algorithm and its parallel implementation on Spark. *Appl. Soft Comput.* **2018**, *63*, 97–109. [CrossRef]

28.   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]

29.   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]

30.   Elahi, M.; Shahzad, A.; Glavin, M.; Jones, E.; O'Halloran, M. GPU accelerated confocal microwave imaging algorithms for breast cancer detection. In Proceedings of the 2015 9th European Conference on Antennas and Propagation (EuCAP), Lisbon, Portugal, 12–17 April 2015; pp. 1–2.

31.   Tournier, P.H.; Bonazzoli, M.; Dolean, V.; Rapetti, F.; Hecht, F.; Nataf, F.; Aliferis, I.; El Kanfoud, I.; Migliaccio, C.; De Buhan, M.; et al. Numerical Modeling and High-Speed Parallel Computing: New Perspectives on Tomographic Microwave Imaging for Brain Stroke Detection and Monitoring. *IEEE Antennas Propag. Mag.* **2017**, *59*, 98–110. [CrossRef]

32.   Tournier, P.H.; Hecht, F.; Nataf, F.; Bonazzoli, M.; Rapetti, F.; Dolean, V.; Semenov, S.; El Kanfoud, I.; Aliferis, I.; Migliaccio, C.; et al. Microwave tomography for brain stroke imaging. In Proceedings of the 2017 IEEE International Symposium on Antennas and Propagation & USNC/URSI National Radio Science Meeting, San Diego, CA, USA, 9–15 July 2017; pp. 29–30.

33.   Ianni, M.; Masciari, E.; Mazzeo, G.M.; Mezzanzanica, M.; Zaniolo, C. Fast and effective Big Data exploration by clustering. *Future Gener. Comput. Syst.* **2020**, *102*, 84–94. [CrossRef]

34. Ullah, R.; Arslan, T. PySpark-Based Optimization of Microwave Image Reconstruction Algorithm for Head Imaging Big Data on High-Performance Computing and Google Cloud Platform. *Appl. Sci.* **2020**, *10*, 3382. [CrossRef]

35. Ullah, R.; Arslan, T. Detecting Pathological Changes in the Brain Due to Alzheimer Disease Using Numerical Microwave Signal Analysis. In Proceedings of the 2020 IEEE International RF and Microwave Conference (RFM), Penang, Malaysia, 17–19 December 2020; pp. 1–4.

36. Karam, S.A.S.; O'Loughlin, D.; Oliveira, B.L.; O'Halloran, M.; Asl, B.M. Weighted delay-and-sum beamformer for breast cancer detection using microwave imaging. *Measurement* **2021**, *177*, 109283. [CrossRef]

37. Tekiner, F.; Keane, J.A. Big data framework. In Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics, Manchester, UK, 13–16 October 2013; pp. 1494–1499.

38. Apache Spark™—Unified Analytics Engine for Big Data. Available online: https://spark.apache.org/ (accessed on 16 January 2021).

39. Zaharia, M.; Chowdhury, M.; Das, T.; Dave, A.; Ma, J.; McCauly, M.; Franklin, M.J.; Shenker, S.; Stoica, I. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12), San Jose, CA, USA, 25–27 April 2012; pp. 15–28.

40. Multiprocessing—Process-Based Parallelism—Python 3.7.10 Documentation. Available online: https://docs.python.org/3.7/library/multiprocessing.html (accessed on 14 April 2021).

41. SharedArray. PyPI. Available online: https://pypi.org/project/SharedArray/ (accessed on 14 April 2021).

42. Saied, I.M.; Arslan, T. Noninvasive wearable RF device towards monitoring brain atrophy and lateral ventricle enlargement. *IEEE J. Electromagn. RF Microw. Med. Biol.* **2019**, *4*, 61–68. [CrossRef]

43. CST Studio Suite 3D EM Simulation and Analysis Software. Available online: https://www.3ds.com/products-services/simulia/products/cst-studio-suite/ (accessed on 21 April 2021).

44. Saied, I.; Arslan, T. Wideband textile antenna for monitoring neurodegenerative diseases. In Proceedings of the 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Bologna, Italy, 9–12 September 2018; pp. 356–360.

45. Byrne, D.; O'Halloran, M.; Jones, E.; Glavin, M. A comparison of data-independent microwave beamforming algorithms for the early detection of breast cancer. In Proceedings of the 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Minneapolis, MN, USA, 3–6 September 2009; pp. 2731–2734.