Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Software Department

Performance Improvement for the Best Replica Selection Using Association Rule and Job Scheduling in Data Grid

A Thesis

Submitted to the Council of the College of Information Technology at University of Babylon in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology/ Software Department

By:

Ruaa Sattar Jabbar Mohamed

Supervised by:

Asst. Prof. Dr. Mahdi Salih Neama Mussa

2023 AD ***1444 AH***

بسم الله الرحمن الرحيم

فَبَدَأَ بِأَوْعِيَتِهِمْ قَبْلَ وِعَاءِ أَخِيهِ ثُمَّ
اسْتَخْرَجَهَا مِنْ وِعَاءِ أَخِيهِ كذلك
كِدْنَا لِيُوسُفَ مَا كَانَ لِيَأْخُذَ أَخَاهُ
فِي دِينِ الْمَلِكِ إِلَّا أَنْ يَشَاءَ اللَّهُ
نَرْفَعُ دَرَجَاتٍ مَنْ نَشَاءُ وَفَوْقَ كُلِّ
ذِي عِلْمٍ عَلِيمٌ

سورة يوسف الآية 76

Supervisor Certification

I certify that the thesis entitled (**Performance Improvement for the Best Replica Selection Using Association Rule and Job Scheduling**) is prepared under my supervision at the department of Software/ College of Information Technology/ University of Babylon as partial fulfillment of the requirements of the degree of Master in Information Technology-Software.

Signature:

Supervisor Name: Assist. Prof. Dr. Mahdi Salih Al-Mahana

Date: / / 2023

The Head of the Department Certification

Given the available recommendations, I forward the thesis entitled "**Performance Improvement for the Best Replica Selection Using Association Rule and Job Scheduling**" for debate by the examination committee.

Signature:

Prof. Dr. Ahmed Saleem Abbas

Head of Software Department

Date: / / 2023

Declaration

I hereby declare that this thesis, submitted to the University of Babylon in partial fulfillment of requirement for the degree of Master in Information Technology \ Software, has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for experts and summaries whose sources are appropriately cited in the references.

Dedication

In my opinion, my family deserves thanks for being the main pillar and my source of strength in life. My dear father spent his life in order to reach our ambitions, and his words were like the light that illuminates my path and overcome obstacles. My mother did not miss a corner of prayer without praying for me. My siblings, friends, husband and children I am grateful to all of you.

I am grateful for the harsh conditions I went through, which would have ended my undergraduate studies, had it not been for the kindness of God and everyone who supported me at that time.

My thanks to everyone who encouraged me to continue and achieve my dream, even with a word! In the end, I owe it to everyone who believed in my ability to realize my ambitions. I am indebted to my family, my husband and some friends for their constant giving, love and faith in me.

Sincerely,

Ruaa Sattar AL-Atabee

Acknowledgment

First and foremost, praises and thanks to God, the Almighty, for His showers of blessings throughout my work to complete the research successfully.

I would like to express my deep and sincere gratitude to my thesis supervisor, Dr. Mahdi Salih Al-Mahana for allowing me to do research and providing invaluable guidance throughout this research. His dynamism, vision, sincerity, and motivation have deeply inspired me. He has taught me the methodology to carry out the research and to present the research works as clearly as possible. Working and studying under his guidance was a great privilege and honour. I am extremely grateful for what he has offered me.

I am extremely grateful to my parents for their love, prayers, caring, and sacrifices for educating and preparing me for my future. I am very thankful to my husband and my children for their love, understanding, prayers, and continuing support in completing this research work.

I express my special thanks to the Head of the Software Department, Faculty of Information Technology, University of Babylon, Prof. Ahmed Saleem Al-Saffar, and all professors of the Software Department for all their advice that helped me and their genuine support throughout this research work. I also thank all the staff of the Software Department, Information Technology Collage, Babylon University, for their kindness.

Ruaa Sattar Al-atabee

Thesis Related Publication (1)

Name of Conference: 3rd International Conference of Information Technology to enhance E-learning and other Applications-2022 [3rd IT-ELA 2022] organized by Computer Sciences Department / Baghdad College of Economic Sciences University

- Paper Title 1: Determining the priority of implementing functions in the grid Computing to reduce the waiting and idle time.
- Paper link: **https://ieeexplore.ieee.org/document/10107959**
- Authors: 1) Ruaa Sattar 2) Mahdi Salih. Software Department, College of Information Technology, Babylon University, Iraq.

Email: ruaasattar.sw.msc@student.uobabylon.edu.iq

Email: **mahdi.almhanna@uobabylon.edu.iq**

Thesis Related Publication (2)

Name of Conference: 3rd International Conference of Information Technology to enhance E-learning and other Applications-2022 [3rd IT-ELA 2022] organized by Computer Sciences Department / Baghdad College of Economic Sciences University

- Paper Title 2: Best Replicas Selection using A priori Algorithm in Data Grid

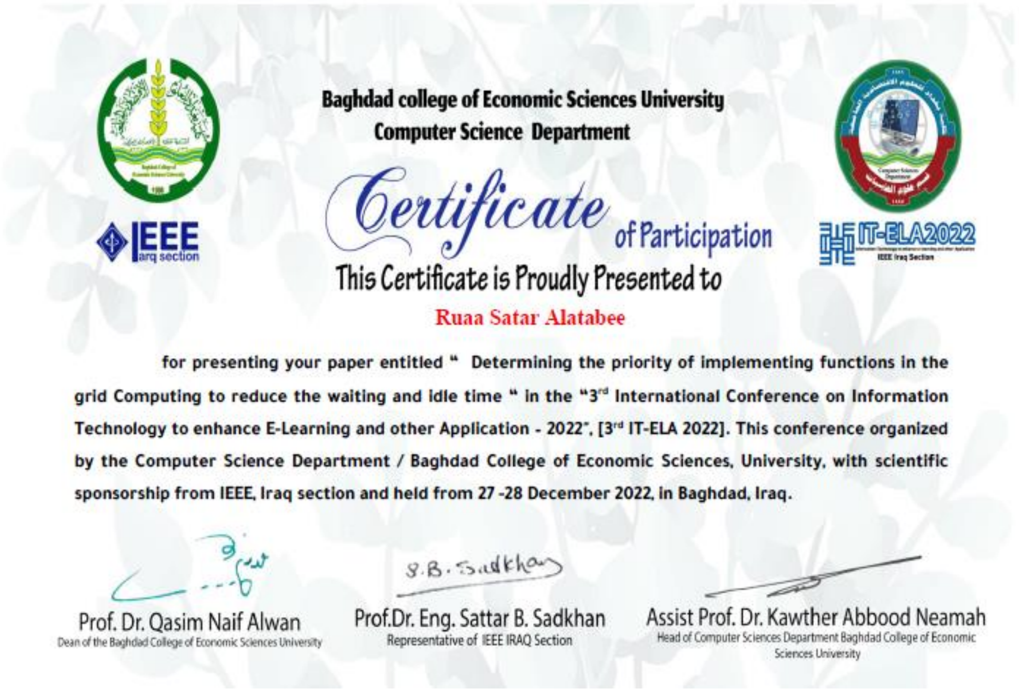
- Paper link: **https://ieeexplore.ieee.org/document/10107926/**

- Authors: 1) Ruaa Sattar 2) Mahdi Salih. Software Department, College of Information Technology, Babylon University, Iraq.

Email: ruaasattar.sw.msc@student.uobabylon.edu.iq

Email: **mahdi.almhanna@uobabylon.edu.iq**

Certificates of participation in the conference



Baghdad college of Economic Sciences University

Computer Science Department

Certificate of Participation

This Certificate is Proudly Presented to

Ruaa Satar Alatabee

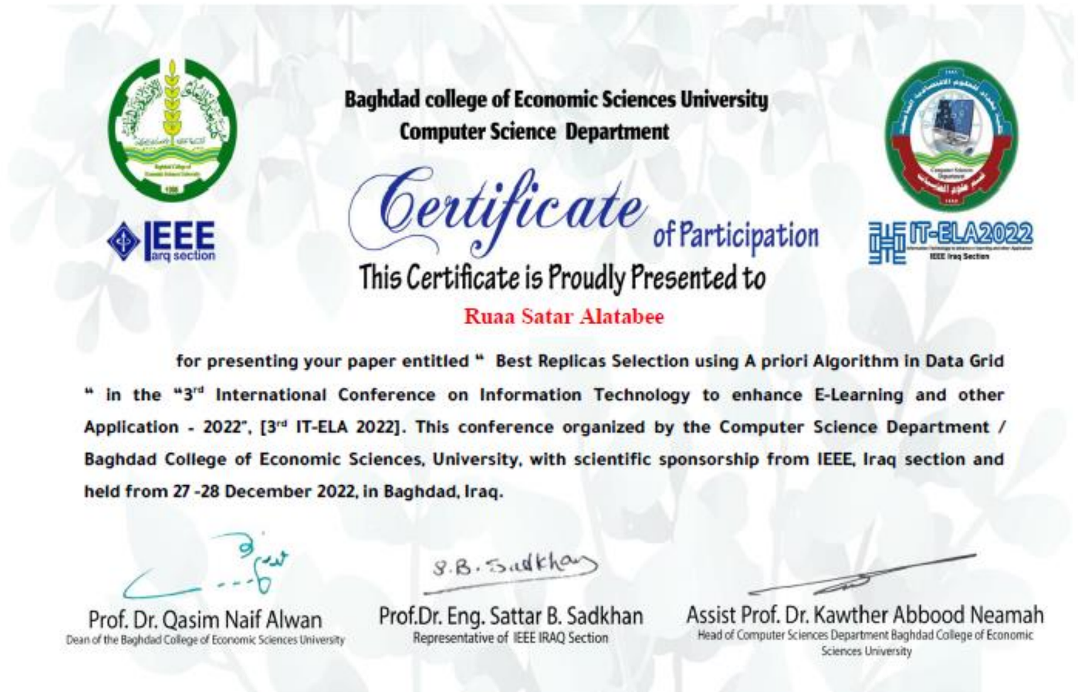
for presenting your paper entitled " Determining the priority of implementing functions in the grid Computing to reduce the waiting and idle time " in the "3rd International Conference on Information Technology to enhance E-Learning and other Application - 2022", [3rd IT-ELA 2022]. This conference organized by the Computer Science Department / Baghdad College of Economic Sciences, University, with scientific sponsorship from IEEE, Iraq section and held from 27 -28 December 2022, in Baghdad, Iraq.

S.B. Sadkhan

Prof. Dr. Qasim Naif Alwan
Dean of the Baghdad College of Economic Sciences University

Prof.Dr. Eng. Sattar B. Sadkhan
Representative of IEEE IRAQ Section

Assist Prof. Dr. Kawther Abbood Neamah
Head of Computer Sciences Department Baghdad College of Economic Sciences University



Baghdad college of Economic Sciences University

Computer Science Department

Certificate of Participation

This Certificate is Proudly Presented to

Ruaa Satar Alatabee

for presenting your paper entitled " Best Replicas Selection using A priori Algorithm in Data Grid " in the "3rd International Conference on Information Technology to enhance E-Learning and other Application - 2022", [3rd IT-ELA 2022]. This conference organized by the Computer Science Department / Baghdad College of Economic Sciences, University, with scientific sponsorship from IEEE, Iraq section and held from 27 -28 December 2022, in Baghdad, Iraq.

S.B. Sadkhan

Prof. Dr. Qasim Naif Alwan
Dean of the Baghdad College of Economic Sciences University

Prof.Dr. Eng. Sattar B. Sadkhan
Representative of IEEE IRAQ Section

Assist Prof. Dr. Kawther Abbood Neamah
Head of Computer Sciences Department Baghdad College of Economic Sciences University

Abstract

Grid computing's organizational structure allows grid facilities to manage data files and their worldwide copies. Data Grid technology is developed to share data across many sites in different geographical locations to improve data access and increase the speed of transmission data. When different sites hold a copy of the files, significant benefits are realized when selecting the best replica sites that cooperate to speed up the file transfer process.

In this thesis, a data mining technique is used, which is the association rules method by using Apriori algorithm to explore the common characteristics of sites to select uncongested replica sites. Some factors such as memory capacity, CPU performance, and available bandwidth affect server performance. Based on these parameters, the best servers are selected to decrease the time required to transmit all critical data and enhance system performance. Scheduling tasks is one of the most challenging obstacles in grid computing. When several tasks are requested, those jobs must wait before they are assigned to servers, which may cause the queue to grow and the waiting time to rise. After selecting the two best servers based on the highest score, job scheduling is optimized using one of the scheduling algorithms which is the Johnson Scheduling Algorithm to reduce total work completion and idle time and enhance system performance.

The Database used in this work is 1500 IPv4(stable and unstable). The work results showed that the method proposed in the first part of the problem was (26 IPv4) out of 41 IPv4 (stable). Scores were calculated for (26 IPv4) and selected the two servers which are the maximum two scores is the best replica servers. The results of the second part of the problem showed that the Johnson algorithm found that the total completion time of all the tasks between two servers is 373 hours and the total idle time is 4.5 hours. As shown in the results of the second part of the problem, the Johnson algorithm is the best scheduling algorithm performance. compared with other scheduling algorithms (FCFS and SJFS) In terms of reducing total and idle time for executing all jobs on the servers.

Table of contents

Table of Abbreviations

Abbreviations	Descriptions
AR	Association Rule
ARS	Association Replica Sites
AT	Arrival Time
BRS	Best Replica service
BT	Brust Time
CT	Completion Time
DC	Data Computing
DG	Data Grid
DM	Data Mining
DHR	Dynamic Hierarchical Replication
DTS	Data Transfer Service
DMDR	Data Mining Data Replication
ERS	Enhance Replica Selection
FTP	File Transfer Protocol
FCFS	First Come First Service
GUI	Graphical user interface
JR	Johnson Rule
LHF	Logical History File
LH	Logical Name
NHF	Network History File
PFRF	File First Replica
PN	Physical Name
QOS	Quality Of Service
RMS RST	Replica Mangment Service Response Time
ROS	Replica Optimaztion Service
RLS	Replica Location Service

RTT	Round Trib Time
SVM	Support Vetctor Machine
SJFS	Shortest Job First Scheduling
TAT	Total Arrival Time
WT	Waiting Time

Table of Figures

List Of Tables

Chapter One
General Introduction

1.1 Introduction

The Data Grid, a collection of data centres spread out throughout the world and linked by a fast network, is an ideal environment for the data- and computation-intensive tasks required by scientific computing. With a data grid, parties in different locations may share and manage resources that need large amounts of data. The best replica selection issue is a crucial part of any data management strategy in a data grid architecture because it determines which copies of data will provide the greatest results to the users [1][2][3].

Several architectural concerns must be taken into account when implement data sharing in a distributed system such as a grid. Replication is an effective method for acquiring access to geographically scattered data. The selection of replicas is a critical problem affecting the Data Grid ecosystem. Data replication may be used to improve data localization and scalability, as well as access and availability [4][5][6] .

Data replication is a crucial approach used to accomplish these objectives by keeping numerous copies of the data intelligently [7]. Data replication ensures the system will continue to operate even if certain data storage locations become unavailable by creating multiple copies of the data in other places. It also hovers over information from sites that offer faster loading times. It is crucial to strategically position duplicates at optimal times [8][9] [10]. Using data mining techniques has resulted in replication algorithms that attempt to better manage replicas in a grid environment by addressing aspects like availability, bandwidth utilization, storage requirements, and reaction time [11][12][14].

The primary objective of the work schedule is to reduce task completion and idle times and increase system output. This difficulty is also known as job scheduling [15]. Some Traditional scheduling strategies, such as “first come, first served” and “schedule shortest task first”, may not be suitable for the proposed system in terms of reducing compilation time and idle time. With an effective scheduling strategy, system performance can be improved and applications can avoid unnecessary delays. Many algorithms have been proposed for job scheduling in network contexts, such as Johnson's algorithm, which provides a straightforward method for determining the ideal order which leads to reduces total time [17] [18].

In a network environment, reducing waiting or standing time is essential. Reducing waiting and idle time is the toughest challenge in network computing. When there are several requests, there may be additional downtime and waiting for processes to complete. The decrease in both average waiting and idle time is a crucial requirement for scheduling issues. The minimization of the average amount of time spent on work in a shop is analogous to the minimization of the total flow time of all jobs, which is the sum of all job completion times in an environment where all tasks are accessible at time zero and processing conditions are predict [13].

1.2 The Problem Statement

The Proposed system focused on some significant problems that are addressed in this thesis:

1. Replication creates and stores identical replicas of the same data in many scattered nodes located in different dispersed places to enhance the speed and availability of data access. The servers all over the world are not equal in processor speed, bandwidth, congestion...etc. When there are many servers to download some files, there will be a problem that the fast server is waiting for a lazy server to complete its task, and this problem causes network consumption and must be avoided. The solution for this problem is using association rule algorithms to find the servers run concurrently

2. When the database is maintained in multiple locations across many locations, it is called a distributed database. It may be stored in several computers located in the exact areas or shared through a network of computers connected. The problem we may encounter is that in some requests we may need data on one server called S1 which contains a part of the database, and other data is kept on another server S2, the data on server S1 is an entry for that data on server S2. Such a problem would require careful scheduling of these requests to process them in a lot of time, reduce the total elapsed time, and reduce idle time on both servers.

1.3 Thesis Objectives

The primary purpose of this thesis is to:

1. To improve the selection mechanism using association rules of the Data Mining approach in the data grid architecture management system.
2. Improving Replica Selecting performance leads to reducing data access latency, reducing bandwidth consumption and distributing the load of the storage site, and improving data access speed. And to do that, the selection set of criteria equation's embedded variables bandwidth, CPU load, and memory availability are used to calculate the Score.
3. To determine the ideal order for processing activities on two servers.
4. Minimizing idle and total time using the Johnson scheduling algorithm.

1.4 Related Works

People have conducted many studies on replica selection and job scheduling in recent years. This section contains a discussion of relevant work. The current study may be separated into two categories: optimal replica selection and workload scheduling over two servers.

1.4.1 Best replica selection

The technique of data replication may greatly enhance system performance and dependability. Consequently, it is commonly used in distributed environments, such as distributed databases, data grids, content distribution networks, grid computing, etc. The issue of data selection resulting from data duplication is one of the hottest concerns in data management.

Numerous scholars have developed ways for selecting the best replica in published works. One strategy used to increase the effectiveness of data exchange in distant systems is data replication. Load balancing, fault tolerance, dependability, and the quality of service may all be improved with the implementation of a data replication plan. If too many requests must be handled at once while data are being stored on a single server, the server may become congestion. The outcome is a slowdown in the entire system. The main characteristic of distributed system replication techniques is requirements [26].

Earlier research mainly concentrated on identifying the best copies by measuring their file sizes and eliminating the others. In this thesis, provide a method that achieves a high-efficiency level by selecting the perfect replicas for various criteria, such as CPU speed, Memory size, and bandwidth, to reduce the overall transmission time.

In 2013, Almuttairi et. al.[5], presented an approach for replica selection called Two Phased Service Oriented Broker (2SOB). Their research sought to learn about, evaluate, and ultimately implement novel approaches to standard selection procedures. The first part of 2SOB is the coarse-grained stage, which is used to identify and separate low-latency (uncrowned network connection) replicas from high-latency (crowned network link) replicas (crowded network links). The outcomes of their trials are exceptional when compared to those of other modern selection approaches for various intermediates

In 2020,Jaradat et. al. [6], hybrid replica-selection technique is presented and implemented in the data grid environment. Best replica selection may be achieved by combining a genetic algorithm with a user choice algorithm. The findings proved that the suggested hybrid strategy greatly outperformed the established approaches in use at the time. The technique they employed to choose replicas takes into account time, site availability, security, cost, and user preferences as the quality of service (QoS) criteria. Their simulation increasing efficiency up to a maximum of 25%.

In 2019, Almuttairi et. al.[8], selection strategy is proposed in to reduce the amount of time spent obtaining essential data. To locate less crowded replica sites, the Pincer-Search algorithm is used to share site characteristics. Through the enhancement of a data-mining methodology, their study enhanced the replica selection method in data grid management systems. By using the provided method, the optimal set of replicas is dynamically chosen, and the selection score in a Data Grids environment is improved. They used only 8 servers. Several Quality-of-Service requirements, including reliability, security, and availability.

In 2020,Mansouri et. al.[11], Data mining-based data replication (DMDR) is a newly suggested dynamic replication approach presented in 2020 that determines the association of accessible data files by examining the file access history. High-dependency files are successfully grouped into the same replica set. They are able to reduce access latency by the centralization factor by storing replicas in optimal places using the DMDR technique. From their CloudSim simulations, they conclude that the DMDR technique outperforms the state-of-the-art approaches in terms of effective network usage, average response time, and access ratio.

In 2021,Song et. al. [14], advocated ERS as an improved strategy for selecting replicators. As a first step, ERS employed a name resolution process that is limited by distance to find nearby duplicates. A local state table stores the state of replica nodes within a limited domain, and the best replica is selected from this table. ERS considered the amount of route congestion between the requester nodes and the replica in addition to network distance and replica node load when making replica selections. Extensive simulations demonstrate that the proposed technique exceeds the most current options in terms of average content download delay.

In 2019,Awang et. al.[15], has focused on the importance of balancing file popularity and affinity as the primary criteria for picking replicas in distributed systems. To choose a replica, they advised prioritizing shared and affinities data. The approach of picking a distributed data copy aims to increase data availability. The effectiveness of a dynamic replica selection technique was measured using PeerSim, a P2P simulator. Evidence from simulations showed that the suggested affinity replica selection added a new facet to the replica selection approach by taking into account the popularity and affinity of file replicas in distributed systems.

In 2021,Jadarat et. al.[16], they propose a matching algorithm that matches the capabilities of grid user with replica providers that are the best fit. This best-fi t approach takes into account both the capabilities of grid users and the capabilities of replica places and creates matches of almost similar capabilities. Simulation results proved that the best-fit algorithm outperforms previous replica selection algorithms.

In 2016, Bsoul et. al.[17], the authors presented a round data replication technique to determine, after each round, which files would be most suited for replication. Their suggested technique was based on the widely used file-first replication (PFRF) strategy, and it addressed the problems with PFRF. The simulation results demonstrated that the suggested technique outperformed the baseline in terms of average file latency per request, average file bandwidth usage per request, and percentage of files discovered.

In 2014, Zou et. al. [18] ,suggested method they called the ant algorithm for selecting the best replica from among the nodes. The team found that by using the ant method, they were able to speed up data access, decrease wait times, spread out the strain on storage nodes, and save money.

Table (1.1) Summary of the related works(best Replica Selection)

Ref. NO	Strategy	Algorithm used	Dataset used	Simulator used	Evaluation metrics	Performance
[5]	A two phased Service Oriented Broker for replica selection	Association rules	Replica Sites	Real data from European data grid	File transfer time, aver- age job execution time	Outperforms on the traditional model, k-NN neural networks. EST genetic Algorithm, replica selection strategies.
[6]	hybrid replica-selection	A genetic algorithm with hybrid	Site's Rate, Request Number ,User's Requested	OptorSim	account time, site availability, security, Total user satisfaction	Increasing efficiency up to a maximum of 25%.
[8]	Replica Management	Association rules (Pincer-Search)	Replica sites	Real dataset from data grid	Reliability, response time, The total transfer time.	Enhanced replica selection strategy to get best replica selection

[11]	Dynamic replication	Data mining-based data replication	File access history for Sites	CloudSim	Average Response Time, Storage Usage, Number of Communications	Reduces data access time, increases data availability, reliability
[14]	Enhanced Replica Selection	ERS	Node State Values, a local log file	OptorSim	average delay, the standard deviation of replica node load, and the standard deviation of link load.	This method is better approach to discover the nearby replicas
[15]	An Affinity Replica Selection Mechanism (ARSM)	(ARSM)	100 sites	A P2P simulator, PeerSim	load balancing, fault tolerance, reliability	Reduce over replication, y improve data availability and accessibility
[18]	Replica selection	Best -Fit	MTTT for Some of replicas	OptorSim	Mean task Turnaround Time (MTTT)	Simulation results proved that the best-fit is better from previous others.

1.4.2 Two-machine job sequencing problem and minimize waiting and idle time

Numerous scheduling strategies have been proposed in academic literature. The majority of them may be utilized in a grid format with little modifications. Job Scheduling is one of the most pressing unanswered challenges in grid computing.

One of the greatest issues in grid computing is developing effective scheduling methods to minimize downtime. An extensive study has been conducted on the impact of two-stage task scheduling on the performance of a parallel-processing system. When there is a large demand for tasks, they must wait to be dispersed across available servers, which might prolong the queue and the amount of time required to perform a job.

It focuses on reducing wait times after the use of scheduling systems. Jobs that are scheduled aid in producing the greatest possible output from a data grid. Experimenting with different scheduling algorithms aims to reduce the total amount of time required to perform a set of activities by discovering the best execution sequence. Because the grid is a distributed environment composed of diverse systems, the commonly used scheduling strategies may not be applicable [19].

The issue of waiting time and idle time is not new, since these measurements are not only significant to theoretical scheduling models but also to many production contexts.

In 2016, Pal et al . [20], they proposed to create a system for implementing Johnson's scheduling algorithm that offers the ideal sequence. This procedure enabled them to get service timings. They explored scheduling methods and the queueing paradigm for servers with limited capacity and many instances. Utilizing a queueing model with several servers and restricted capacity, they were able to decrease waiting time and line length, hence improving the job scheduling model.

In 2020, Okwu et al. [21], The objective of their study in is to effectively schedule tasks in a manufacturing firm utilizing Johnson's algorithm at a renowned plant for the manufacture of clean water. Each function must undergo the same series of procedures. The tasks are allocated so that the first task is assigned to FM1, and after FM1 has completed its processing, the second job is assigned to CM2. They used Johnson's technique to decrease the idle time for both FM1 and CM2 by choosing the best processing order. Using Johnson's approach, they were able to minimize the idle time for both FM1 and CM2 by calculating the ideal sequence of processed functions. Finally, they found the ideal sequence for finishing the project and decreased the overall time between performing the first and final tasks on FM1 and CM2.

In 2017, Xiong et al. [22] ,analyzed concerns with cloud data focused work scheduling by examining existing constraints and constructing a mathematical model for efficient job scheduling. To tackle the challenge, a hybrid algorithm was created by combining Johnson's rule (JR) and genetic algorithm (GA) to build a Johnson-rule genetic algorithm. Coding was completed and sensitivity analysis was conducted through simulation. The hybrid algorithm produced superior and quicker results than the conventional scheduling method. Other similar schedule optimization studies may be found in the works of the following authors [23][24].

Table (1.2) Summary of the related works (Job scheduling)

Ref.NO	**Year**	**Algorithm used**	**Dataset used**	**Evaluation metrics**	**Performance**
[20]	2016	Johnson algorithm	Times for jobs	Waiting time And line length	Decrease waiting time and improving the jobs scheduling model
[21]	2020	Johnson algorithm	Times on machines	Idle time	Reducing idle time and total time
[22]	2017	Johnson algorithm with GA	Times	Data in cloud	Best results for jobs scheduling

1.5 Thesis Outline

After the first chapter presents a general introduction to the topic, the thesis is organized as follows:

- Chapter Two: **"Theoretical Background “, provides** a comprehensive description of the basic concepts of the
- Chapter Three: **"The Proposed System Design ",** describes the design of the proposed system and the algorithm of the system stages.
- Chapter Four: **" Results and Discussion "**, describes the evaluation result of the proposed system.
- Chapter Five: **"Conclusions and Suggestions for Future Works"**.

Chapter Two
Theoretical Background

2.1 Introduction

However, this chapter should not solely summaries the findings of the previous scholar, the aims of this chapter is to determine what is and is not known about the issue through in-depth discussion and analysis of the existing body of information. The questions and hypotheses of the study are derived from this conclusion. There will be times when you decide it would be beneficial to repeat an earlier thesis.

In this chapter, The grid environment and what are the most critical challenges and problems in this environment. Selection of the most essential replicas based on data mining techniques will also be demonstrated. Also, talk us about the jobs scheduling, the performance measures such as waiting and idle time and all related in this thesis.

2.2 Data Grid (Grid Computing)

Data grid environments are now regarded as the world's single and strong computer systems, which is especially relevant in today's ubiquitous world of wanting information anytime and everywhere. It has become clear that to reap the many benefits of Grid Computing [12].

In reality, it is far more difficult to meet modern industrial challenges' complexity and dynamic character using the more conventional, single computing platform methodologies. In the field of Grid Computing, an infinite number of pervasive computers are linked together via a network, or "grid." This cutting-edge computing method is most easily understood as a gigantic "utility" grid, like the one that supplies electricity to our homes and businesses every day. In many parts of the globe, receiving electricity from a central source is as natural as turning on a light switch [25].

The requirement to solve more complicated computational problems has led to an increase in the complexity of almost all computer systems, as well as in their associated costs and operational considerations. However, the capacity to collaborate effectively via modern computers is essential in almost all fields of industrial and economic issue-solving, from scientific research to the use of market forces to academic problems. Reaching the degree of resource cooperation essential for addressing these complex and dynamic challenges within the constraints of the requisite quality standards of the end user is a challenging task for many technical groups[26].

In the same utility pattern, Grid Computing seeks and can an infinite computer into any data grid, improving capabilities and problem-solving duties within the operational grid environment. No one knows yet how great Grid Computing's problem-solving potential is moving on into this new age of tremendously powerful grid-based problem-solving solutions. The needs of businesses all around the globe need highly developed problem-solving skills for complex issues of immense scale. The demand for dynamic cooperation to solve more complex challenges has spread across all sectors of the global economy capacity for a wide variety of pervasive computing resources to cooperate effectively [38].

To further demonstrate the nature of this setting and the complexity of the associated technological issues, it’s been started to evaluate the various benefits of a Grid Computing solution environment by thinking about some frequent use case situations.

2.2.1 Examples to serve as an introduction to the principles of Grid Computing

- Collaboration across departments of a financial institution processing a wealth management application allows for increased computing capacity and software modelling capabilities. It manages complex data transfer activities by pooling many computer resources, which allows for quicker performance via actual execution of the jobs and quick and easy access to complex pools of storage of data. In the end, this means happier customers and a shorter turnaround[27].
- An international team of researchers examining the ozone layer in the atmosphere will amass massive volumes of experimental data daily. To efficiently meet the processing demands of their research, these experts need sophisticated data storage capabilities spread among several, widely distributed storage facilities. Because of this, crucial scientific research may be conducted more efficiently and productively.
- These days, it's not uncommon for there to be hundreds, if not thousands, of players online at once in a single game, necessitating a network of servers rather than a single, centralized one. Now gamers from all around the world may collaborate in-game in real-time. Computer resources, networks, and data storage facilities must all be made available on-demand for this to be possible. This immediate need fluctuates greatly from one instant to the next and is always determined by the current system load. This leads to bigger gaming communities, which in turn necessitates more complicated infrastructures to handle the influx of new players, boosts revenue for gaming companies, and makes customers happier[28].

The aforementioned issues are only a few of the many that may be tackled and solved using the various tools available today thanks to Grid Computing. Many technologies and open standards are used to build Grid Computing systems. Grid Computing, on the other hand, offers techniques for transparently identifying and negotiating access to distributed computing resources that are highly scalable, secure, and high-performance[29].

This paves the way for an unparalleled level of resource sharing across an endless number of dispersed organizations throughout the globe's computer infrastructure. This has far-reaching implications for how computing is implemented, both at the individual and corporate levels, moving us closer to a model in which a computer is provided as a utility much like electricity or water. These electricity and water utilities, like Grid Computing utilities, may be accessed "on demand" and are always accessible to provide a facility that has been agreed for private or business use[30].

2.2.2 Replica Selection Problem

The necessity for Grid Computing arose from the realization that a single computer could not match the performance required of it, necessitating the integration of a variety of distributed computing resources[5]. The replica selection problem can be divided into two sub-problems:

1) Discover the physical location(s) of a file, given a logical file name.
2) Select the best replica from a set based on certain selection criterion.

Since the same file has multiple physical names in different storage locations, the job of Replica Location Service (RLS) is to maintain associations or mappings between logical file names and one or more physical file names of replicas. A data grid job can provide a logical file name to a RLS server and ask for all the registered physical file names of replicas as shown in Figure (2.1).

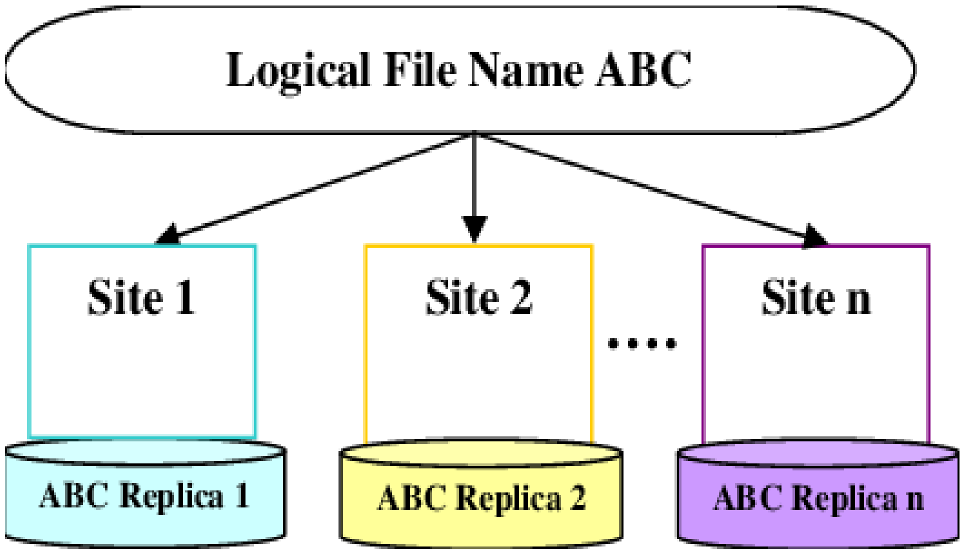


Figure 2.1 Example of the associations between a logical file name and three replicas on different storage sites [9]

2.3 Data Replication

When it comes to performance assurance, or more specifically, meeting a predetermined performance standard, service providers have several options at their disposal. Among them, data replication stands out as a widely used and extensively studied data management strategy. Data replication has several advantages, such as higher performance by careful placement of replicas, greater availability through having multiple copies of data sets, and increased fault tolerance against potential server outages. Tenant queries given to the DMS may need many relations to proceed with the execution, depending on the strategy of execution (e.g., the number of connections)[31].

There is little doubt that not all necessary data will be available on the execution node itself in a large-scale system where relations are fractured and geographically spread across numerous servers running. The probability of some distant data being transported from faraway servers is genuine given that a query is handled on numerous servers according to inter-operator and intra-operator parallelism. The query execution process can hit a bottleneck if the network bandwidth available to the distant servers is limited, for example, if the remote data is stored in a geographically dispersed location[32].

Before a query is ever executed, the bottleneck data should be detected heuristically to be picked for potential replication to guarantee that the response time target will be met.

Another crucial choice that must be taken towards this end is when to initiate the replication process itself. Later on, in the data replication decision process, it will be necessary to decide how many duplicates to produce and what to do with the replicas that are no longer needed. Reducing data access latency and increasing response time satisfaction is greatly aided by strategically placing the newly formed copies[33]. This is particularly true with economically driven, large-scale systems like cloud computing, where every choice must be made with the provider's bottom line in mind.

A good data replication strategy must be able to make meaningful decision making about (i) what to replicate to accurately determine which fragments of relations require replication, (ii) when to replicate to adapt to changes in demand for data on time to exact purpose difficulties, and (iii) how many replicas to produce to reduce waste precious resources like storage to keep costs down [34].

However, depending on the distance between the locations, the effect on application performance might be substantial. The speed issues with synchronous replication may be avoided by asynchronous replication; however, it does require accepting some data loss[35].

2.3.1 The purpose Data Replication

The purpose of the service known as "Data Replication" is to copy and store data in many locations that are conveniently located for end users.

Several grid-based applications are available, including climate data gathering and the physics grid system, benefit from this method because they need quick and easy access to and manipulation of massive information. In addition, if there are numerous replicas, the user will need a Replica Management Service (RMS) to help them find them and choose the one that is the best fit for their needs. RLS is used to catalogue all replicas' logical names and physical locations. The purpose of the replica selection technique is to provide the best possible response to the user's request[14].

Replica Selection is a method for selecting the optimal replica site from among many replica locations based on the quality of service (QoS) criteria. Response time (RsT), security, availability, dependability, and affordability are just a few of the many Quality of Service (QoS) criteria that are vital to the Grid. The replica selection and, in turn, the task turnaround time are both impacted by RsT, or for simplicity time. In the past, replica selection algorithms solely used the replica's movement speed from source to sink while making their decisions[15].

2.4 Data Mining

The term data mining "DM" tends to refer to the automatically generated method of discovering patterns in data. previously undiscovered knowledge and information from big or complicated data sets. This may include discovering patterns, relationships, changes, trends, anomalies, and important structures. The primary objectives of data mining techniques and tools are to obtain specific information that can be applied to huge datasets. The next paragraphs provide an introduction to the four primary data mining tasks supported by data grid strategies: association analysis, prediction, classification, and clustering[36].

Knowledge discovery from databases makes use of many various algorithms and methods, such as classification, clustering, regression, artificial intelligence, neural networks, Association rules, Decision trees, and genetic algorithms. Nearest-neighbor approach, etc.[37].

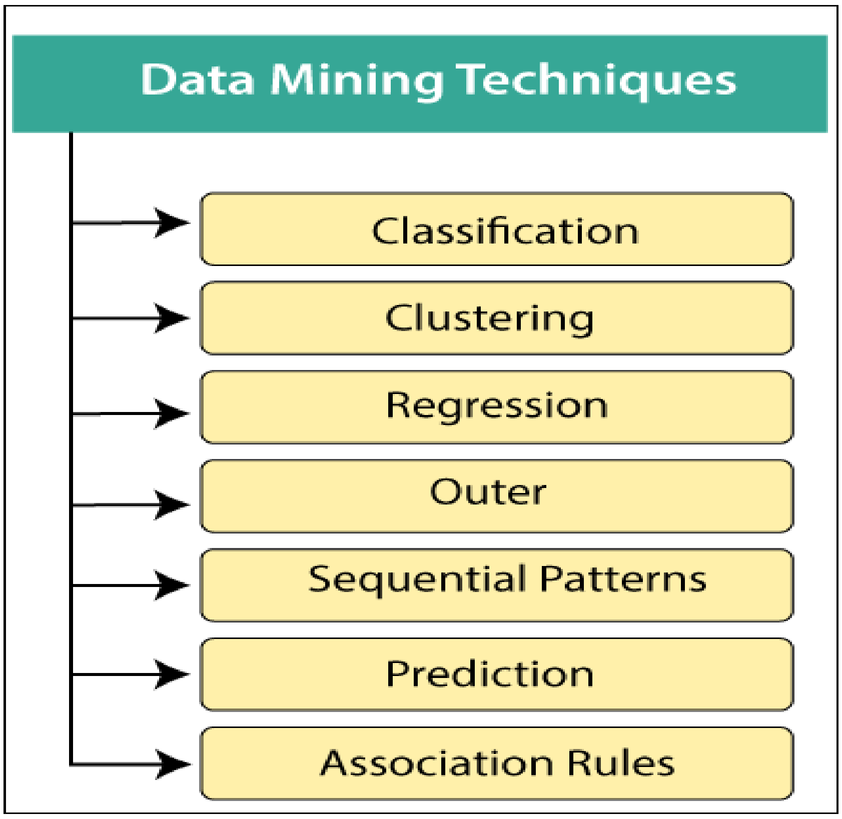


Figure 2. 1 *Data mining Techniques* [38].

2.4.1 Classification

Classification is the most widely used data mining strategic approach, which takes a sample of records that have already been categorized and uses that information to train a model that can generalize to the whole population of records. This kind of analysis shines in contexts like fraud detection and credit risk assessment. Algorithms for classifying data, such as decision trees or neural networks, are widely used in this method. Learning and sorting are integral parts of the data categorization process. For learning, a classification algorithm is used for the data used during training[39].

The reliability of a set of classification rules may be estimated with the use of test data in classification. The new data tuples will be evaluated for correctness before the rules are implemented. Complete records evaluated on a record-by-record basis, are required for a fraud detection application. These labelled samples let the classifier-training system zero in on the precise configuration of discrimination parameters. Next, the algorithm encodes these values into a model known as a classifier. Among the several classifications, models are: The Classification by induction of a decision tree A few examples of classification methods are: Bayesian Classification; The Neural Networks; Support Vector Machines (SVM); Association-Based Classification[40].

2.4.2 Clustering

Clustering one definition of clustering is the discovery of groups of items that have common characteristics. By using clustering methods, learning more about the general distribution pattern of objects and the associations between their various qualities. Clustering may be employed as a technique used prior to attribute subset selection and classification, making the latter a more cost-effective alternative to the classification strategy.

For instance, using this method to generate groups of clients based on their shopping habits. Various Clustering Strategies Methods of Partitioning Agglomerative (divisive) approaches in a hierarchical framework Density-based techniques Strategies using a Grid Applications of Model-Based Techniques[41].

2.4.3 Prediction

Predictions may be made using a modified version of the regression method. Modelling the connection between independent and dependent variables is possible using regression analysis. In DM, the response factors are the ones seeking predictions for, whereas the independent variables are traits that are previously known.

Regrettably, many issues in the actual world cannot be solved by only making predictions[42]. Predicting sales, stock prices, or the frequency with which a product will fail is challenging since each of these factors may be influenced by several other, more suitable factors. Predicting future values may need more advanced methods (such as logistic regression, decision trees, or neural networks). It's common for regression and classification to have the same model types.

Classification trees (used for categorical response variables) and regression trees (used for continuous response variables) may both be constructed with the help of the CART (Classification and Regression Trees) decision tree technique (to forecast continuous response variables). Classification and regression models may be generated using neural networks as well. Various Regression Strategies Nonlinear Regression Multivariate Nonlinear Regression Linear Regression Linear Regression Multivariate Nonlinear Regression[42].

2.4.4 Association Rule

Association is a popular DM technique that identifies patterns based on the association between variables in a single transaction. It is additionally known as the elation technique because it exploits the relationship among items to discover the frequent occurrence of various items with the highest frequencies inside the data set [38].

The common goal of association and correlation analysis on huge data sets is to identify patterns in groups of items. Information like this is useful for a variety of business considerations, including catalogue layout, cross-selling products, and studying consumer purchasing patterns. Rules with confidence levels below one must be generated via Association Rule algorithms. However, there may be a vast number of Association Rules for a particular dataset, and many of these rules will be of little use. A Classification of Association Rule Types "Multilevel Association Rule" Multidimensional Correlational Analysis the Rule of Quantitative Association[43].

2.4.5 Neural networks

A neural network consists of a collection of interconnected input/output nodes, each of which carries some kind of weight. During training, the network adjusts its weights to better anticipate the proper class labels for the input tuples. Neural networks may be used to identify patterns and find trends in data that are too sophisticated to be recognized by people or other computer approaches. For inputs and outputs with continuous values, they are ideal. Examples of where these techniques have found success in practice include reorganizing handwritten characters, teaching a computer to speak English text, and solving a wide range of practical business challenges. For purposes of prediction or forecasting, neural networks excel because of their ability to recognize patterns and trends in data. Back Propagation Neural Networks [44]

2.5 Replica Selection Techniques Based on Data Mining

This next section presents the most important Replica selection techniques based on data mining in the grid environment.

2.5.1 Predictive technique for Replica Selection

By making numerous copies of data in separate places, replication is a method used in Data Grid setups to lessen the impact on access latency and network bandwidth use. The availability of more data, thanks to replication, makes the system more trustworthy. Choosing where replicas should be hosted is one of the problems that must be solved. Furthermore, there is tremendous value in determining the best duplicate of a file when many sites store copies of the same file[45].

Because Grids are always evolving, the candidate sites that presently host copies may not be the optimal places to retrieve a copy in future periods. Therefore, it is recommended to use a replica reallocation technique to move replicas to other locations if the performance measure severely declines. To determine the optimal location for a file's duplicate, an algorithm is presented that takes into account the file's transfer log history and the locations of neighboring files[46].

Additionally, a forecasting method is presented for estimating the time required to move replicas across locations. If many sites presently host a replica, the expected transfer time may be used to determine which site has the best copy and therefore be used to choose the best duplicate. Because data must be cached near consumers, data replication is becoming more important as Data Grid technology is developed to provide data sharing across multiple businesses in various geographically dispersed locations. Data replication ensures that information may be quickly restored from backups even if a primary copy becomes corrupted, lost, or unavailable. In addition, data access performance may be greatly enhanced if data is replicated and stored near its end users[47].

2.5.2 The Replication strategy based on clustering analysis

Plan for a Reproduction by analyzing the dynamic access characteristics of client nodes in a data grid, a clustering analysis is performed based on the relations between access files, and the data files belonging to the same correlative files set are replicated to the data nodes close to the client nodes. All of the duplicated files belong to the same set of correlated files, but the files retrieved by the client nodes only make up a small subset of that set (often one or two files).These definitions and characteristics change the process of obtaining the correlative files set into the process of obtaining the correlative files equivalence class. Using classification techniques to improve replica selection in data grid[48].

2.5.3 Using Associated rules for replica selection based on the APRIORI approach (ARRA)

The introduction of ARRA is split into two sections. A pioneer data mining method called Apriori is used to examine the access patterns of data-intensive activities in the first section. The second section involves the creation and implementation of rules for the replacement of replicas. Data files that are accessed are like objects in a data grid, and the data files needed for a particular data-intensive operation are like transactions. Simulations run on the CMS grid reveal that ARRA outperforms LFU in terms of mean task time, the number of remote file access, and effective network consumption[49]. The Apriori algorithm is an iterative method of mining transactional databases for frequent item sets and producing association rules based on these sets. The idea is grounded on the fact that a set of things is frequent if and only if each of its non-empty subsets is also frequent.

Apriori produces regular patterns of length k at each iteration. All frequent items of length 1 are located via the 1stiteration. In this step, we build a candidate set consisting of all conceivable combinations, each of which has frequent non-empty subsets. On the second try, we find frequent item sets of length 2 [50]. The single-phase Apriori implementation with the generation of a candidate set based on the output of the previous iteration then proceeds to search the dataset for occurrences of item sets in the candidate set and returns those item sets whose occurrence count exceeds the user-specified minimum support. Apriori has been implemented using Map Reduce in Hadoop. Because it provides a parallel storage and computing environment, Hadoop makes Apriori more scalable and trustworthy[9].

2.5.4 Replication strategy based on maximal frequent correlated pattern mining

There is a developed a novel method they dubbed RSBMFCP. i. Retrieving Activity Logs from Files. (ii) Transforming the file-system activity log into an extraction context (i.e., a logical file access history). Discovering the hidden file correlations requires (iii) mining maximum common correlated patterns[51]. For each MFC pattern, the largest possible group of files that occur together often and have a correlation degree higher than a predetermined threshold on a specialized correlation measure is represented. The fourth need is that the replication and replacement process to be carried out. When comparing task execution time and efficient network use, RSBMFCP is superior to the R, Periodic Optimiser, and DPRSKP methods[52].

2.5.5 Support and confidence dynamic replication algorithm

The fundamental notion of the BSCA technique is to pre-fetch frequently visited files and their related files to the area near the access point. Using the data access number and data access serial determines the relationship between the data files. Support and confidence measures for mining association rules form the basis of BSCA in addition to the usage of access numbers in A critical study of data grid replication methodologies[53].

The data-mining algorithm and the replication algorithm are the two subsidiary algorithms powering this tactic. Once the data mining technique is employed to discover frequent files, support and confidence of association rules between these frequent files are determined. Frequently used files and their connected files are duplicated if the support and confidence levels between files surpass respective minimal criteria. On the OptorSim simulator, BSCA performs better than SBU, ABU, and Fast Spread. One way to do this is to have the fastest typical reaction time. To compare BSCA with other strategies, simply the mean reaction time is utilized as an indicator of performance[54].

2.5.6 A pre-fetching-based dynamic data replication algorithm

PDDRA's core premise is similar to PRA's. PDDRA analyses a grid site's file access logs to foresee what files it will require shortly and then pre-fetches those files to the requesting grid site. Because of this, if there is a need for this specific file in the future, it will be accessible on the local server. File access patterns are recorded, a file is requested, and then replication, pre-fetching, and replacement are carried out as part of PDDRA [54].

2.6 Grid architecture

In this section, Data Grid Architecture is explained with functionality of each component [46].

2.6.1 The Replica Management Software (RMS)

Replica Management System (RMS) As we see in Figure (2.3) below, the main component of Data Grid is RMS. RMS acts as a logical single entry point to the system and interacts with the other components of the systems [55].

The following words will need to be defined before a full explanation of RLS can be given[56]: The contents of a file may be identified by its unique name, known as its Logical File Name (LN).

In computer storage systems, a file's PN refers to its exact position. Figure (2.2) is an illustration of these concepts. RLS's responsibility is to store mappings (or relationships) between logical file names and their corresponding replicas' physical file names. An RLS server allows a user to provide a logical file name and get all of the associated physical file names for registered replicas[57].

Before explaining RLS in detail, The definitions a for a:

A• Logical File Name (LFN) is a unique identifier for the contents of a file

B• Physical File Name (PFN) is the location of a copy of the file on a storage system

A user may also make use of RLS servers to discover the logical name of a file that corresponds to a given physical location. Additionally, RLS enables users to link logical or physical file names registered in the catalogue with characteristics or descriptive information (such as size or checksum). Moreover, users may query RLS depending on these characteristics[58].

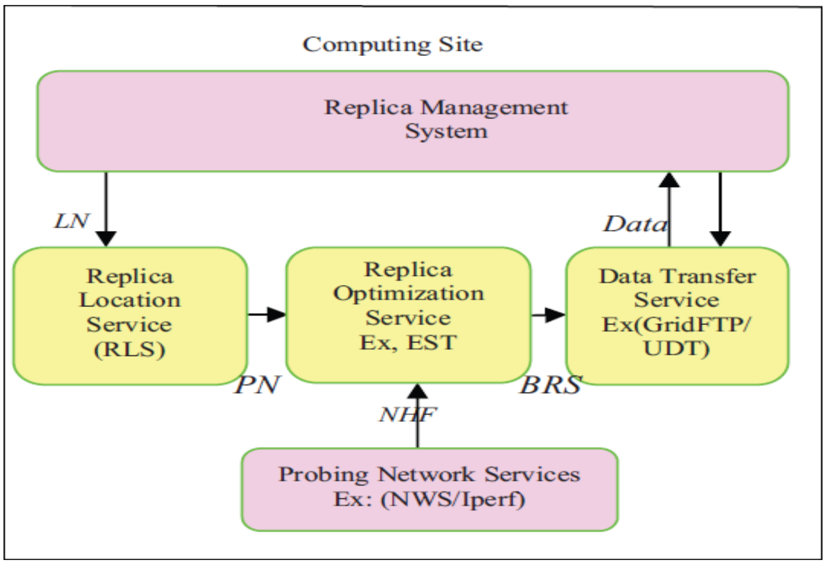

Computing Site
Replica Management System
LN
Data
Replica Location Service (RLS)
Replica Optimization Service Ex, EST
Data Transfer Service Ex(GridFTP/ UDT)
PN
BRS
NHF
Probing Network Services Ex: (NWS/Iperf)

Figure 2. 2 Functionality of Replica Management System[46]

2.6.1.1 Replica Location Service (RLS)

This service, known as Replica Location Service (RLS), is responsible for maintaining a database of all the storage nodes where replicas have been placed. Its job is to keep track of all the files that have been logged by users or services at the time those files are produced. The RLS server is then queried by users or services looking for these copies.

2.6.1.2 Replication Optimization Service (ROS)

The Optimization feature is used to speed up data access by routing requests to the most relevant copies and duplicating frequently used files in light of usage data[59]. The optimization service works to find the BRS that has the lowest network and storage access times. Data access latency information is collected by ROS from several sources, including network monitoring services like Network Weather Service (NWS) and Iperf Service [57].

2.6.1.3 Data Transfer Service (DTS)

Once DTS has the physical addresses, RMS may request that the files be sent via a high-performance, secure, and reliable data transfer protocol, such as GridFTP or UDT. The following part clarifies where our model lives and how much it affects data grid performance, building on the foundation of a clear and concise understanding of the data grid's underlying architecture[60].

2.7 Improve Selecting Score of the Replica Selection Technique

The calculation and determination of the weight are updated using the formula to increase the probability that choosing the best reproductions. The score is calculated based on data gathered from many server devices. Factors that have a direct impact on data transport include network bandwidth and latency. The data transmission rate is also somewhat affected by the CPU and I/O speeds[46][61].

2.8 Association Rule in Data Mining

Data mining's "go together" attribute discovery is called the "association problem." The job of association aims to discover criteria for measuring the link between two or more features, and is most often used in the business sector under the names affinity analysis and market basket analysis. The form of an association rule is "If antecedent, then consequent," and the strength and certainty of the rule are quantified.

Applications of association rules include catalogue design, cross-marketing, clustering, loss-leader analysis, basket data analysis, and classification. For instance, if a consumer purchases bread, he may also purchase butter. The buyer may purchase a memory card in addition to a laptop. Support and confidence are the two fundamental criteria that association rules use. It identifies the connections and rules produced by searching for common if/then patterns in data. Typically, association rules are required to simultaneously provide a user-specified minimum support and a user-specified minimum confidence [51].

A supermarket may discover, for instance, that out of a total of 1000 consumers, 200 purchased diapers and 50 purchased milk. With these numbers, the confidence in the association rule "If purchase diapers, then buy milk" would be 50% and the support would be 20% out of a possible 1000 instances[62]. Using a huge dataset, association rule mining might uncover potentially useful correlations or associations. Mining association rules from databases is becoming more appealing to various sectors as they gather and store ever-growing volumes of data Catalogue design, cross-marketing, loss-leader analysis, and other corporate decision-making processes may all benefit from the identification of intriguing association links across massive volumes of business transaction information.

The purpose of association rule discovery is to identify rules that have robust relationships between database elements. One of its main goals is the identification of connections between objects in the fin form A B, in the antecedent is the first term and the result is the second [63].

The amount of item sets that may be examined for rule generation is limited by the minimum support restriction. The strength of a set is measured by how often its components appear in the data collection. Pepsi's support, for instance, would be equal to 0.25 if 25 out of 100 transactions (assuming a collection from the database consisted of 100 transactions) included Pepsi. Regular sets are those that meet the minimal support requirement. These collections serve as the basis for the rules that are subsequently created.

Continuing with the previous example, Pepsi would be considered a common item if the user specified the minimum support to be 0.2 [64].

Take the rule "Pepsi chips" as an example.

40 percent oper centurchases made by customers include Pepsi if o Support (Pepsi) = 0.4. To illustrate, if o Support (Pepsi chips) = 0.2, then 20% of all client transactions include both Pepsi and chips. The established rule set might then serve as the basis for the user's decision to impose further limitations. Thus, several unnecessary regulations will be eliminated. Association rule discovery often generates rules with a certain level of confidence in them. Among the user-specified metrics of interest are the following:

$$\text{Confidence } (A \Rightarrow B) = \text{support}(A \Rightarrow B) / \text{ support } (A) \qquad (2.1)$$

There are three steps to finding association rules:

- First, do a data spaces search to identify all item sets (or even just one item) with higher minimum support than the user entered. These sets of items are the most common ones to find together.
- Second, use the most common collections to generate engaging rules. To be considered intriguing, a rule must have a confidence greater than the user's required minimum confidence.
- The third step is to "prune" the rule set by getting rid of the uninteresting rules.

2.8.1 Apriori algorithm

Apriori is one of the best algorithms for learning association rules. The Apriori method is often used to uncover such association rules. Apriori is first suggested by Agrawal and Srikant. The algorithm has established itself as the go-to method for discovering new associations and has revolutionized the association rule mining field.

Apriori is intended to run on databases that contain transactions, such as collections of items purchased by customers or details of a website's frequency or Ipv4. The generation of rules in Apriori is a two-stage procedure [65]:

A. Identify all common collections of things. The frequency data (backing) for the set of items are kept. Taking this action will reduce the swath of possible item sets used in the antecedent and consequent of the rules.

B. Second, the sets of frequently occurring elements are used to create association rules. Minimum support testing has been performed on all objects in the database. A candidate set of item sets may be built from the discovered 1-item sets or seed set. It is possible candidate sets with k items by joining sets with k-1 frequent items. Next, checking whether the candidate set (k item set) meets minimal support, and if so, use it as a starting point for the subsequent round of testing.

This algorithm extends frequent subsets one item at a time, which is known as the candidate generation process. The data is then used to test groups of candidates. Apriori employs a breadth-first technique and a hash tree structure to efficiently count candidate item sets. It finds the most frequently occurring individual items in the database and expands them to larger and larger item sets as long as those item sets appear frequently enough in the database.

The Apriori algorithm generates frequent item sets that can be used to generate association rules that highlight broad patterns in the database.

Algorithm (2.1) : Apriori Algorithm

```
Input:
(1) $LHF$ (2) min-conf (3) min-supp
Output: $ARS$
Begin
Initialize (1) $k = 1$, (2) $C_I$ = all the l-itemsets /* set of sets with single replica site*/
1: Read $LHF$ to count the support of $C_I$ to determine $L_l$, $L_1 = \{\forall$(1-itemsetes ) $\in C_l$ and $s$ (1-itemset ) $\geq$ min supp
2: k $= 2/*$ k represents the pass number*/
3: Read $MR/*$ Maximum number of replicas $*/$
4: While ($L_{k-1} \neq \emptyset$ and $k + 1 \leq MR$) do
5: begin
6: $C_k = \emptyset/$ start generating candidate-itemsets %
7: for all itemsets $l_l \in L_{k-l}$ do
8: for all itemsets $L_j \in L_{k-I}$ do
9: if $\big(l_i[1] = l_j[1] \wedge l_i[2] = l_j[2] \wedge \ldots \wedge l_i[k-2] = l_j[k -$
, then ($c = l_i[1], l_i[2], \ldots, l_i[k-1], 2] \wedge l_i[k-1] < l_j[k-1]$
$l[(k-1])$
10: $C_k = C_k \cup \{c\}$
11: for end
12: for end /end of generating candidate-itemsets/
13: for all candidate itemsets $c \in C_k/*$ start pruning process $*/$
14: for all $\{(k-1)$-subsets of $c\}$ do
15: if ($d \notin L_{k-1}$) then $C_k = C_k - \{c\}/*$ Delete $c$ from $C_k */$
16: for end / end pruning process/
17: for all transactions $t \in T$ do
18: $L_K$ = All candidates in $C_k$ with minimum support
19: $k = k + 1$
20: While end
21: end $ART$
22: Return $ARS = U_\kappa L_k$
End Apriori
```

Algorithm (2.1) Steps of Apriori Algorithm [65].

2.9 Task Scheduling

To reduce overall production time, we examine the two-machine flow shop scheduling issue. N jobs are waiting to be processed by two machines, and both machines must complete their respective tasks before moving on to the next job. In this case, the goal function is equal to reducing the mean time it takes to do a project. If you want to minimize the mean or total completion time in a deterministic two-machine flow shop, you'll want to use a permutation schedule, since this is the most common kind of schedule. This means that the ideal schedule may be determined by simply comparing the order of tasks on both computers[66]. It is well-established that permutation schedules are preeminent for the two-machine flow shop issue when processing durations are completely random.

Schedules based on permutations are particularly well-suited to the situation at hand since we assume that processing times are independent random variables within certain limitations. If (a) each machine can only handle one work at a time, and (b) each job may be broken down into many tasks that can be allocated to different machines, then the goal is to discover an assignment of all the tasks of n jobs to m machines that maximizes the total utility under these conditions[67].

2.10 Task Scheduling in the Data Grid

To share data and resources, the Grid uses an integrative design to link together hundreds of computers and storage facilities situated in various parts of the globe. Data-intensive scientific applications are in high demand from academics all around the globe. Examples include High Energy Physics (HEP) data processing, climate modeling observation. The potential petabyte-scale of data that must be accessible by this application presents several difficulties for the Data Grid. A data Grid is often used for data-heavy applications. Some data management tasks include copying or moving data between certain locations. Data management difficulties become crucial to scheduling and efficient resource management in Data Grids as the number of data-intensive operations increases. Because of its importance in allocating resources to tasks in light of application needs and resource utilization performance, the Grid scheduler is a crucial part of any resource management system[68].

Resource discovery, system selection, and task execution are the three stages that make up the Data Grid scheduling method. When the Data Grid scheduler has to locate a collection of acceptable resources for task execution, the process of "resource discovery" identifies those resources that can be employed with their capabilities. Data requirements for individual jobs and the effect of the replication method on scheduling efficiency are two factors to think about.

In data-intensive applications, the system selection and performance are profoundly affected by the locations of essential data by the job [69]. Time spent transferring data is a major contributor to jobs being late in large-scale data-intensive systems. It is crucial to integrate data replication and scheduling algorithms. Which files should be duplicated, when new copies should be created, and where they should be stored are all decisions made by the replication system[70].

New scheduling and replication techniques are developed in this study to speed up data transport and complete tasks more quickly. The new scheduling mechanism we've implemented thinks about things like where data has to go, where bandwidth needs to go, and how much computational power each node has. Bandwidth is taken into account while choosing and replacing replicas in the suggested replication strategy. The likelihood of using data stored on a neighbouring a cluster grid is similarly improved. In data-intensive applications, data transmission latency causes job execution delays and job data locations affect Grid scheduling and performance. Our Scheduling Strategy incorporates cluster- and site-level data for scheduling jobs. The SS finds the optimum cluster and location [71].

The best cluster (and site) stores most of the job's needed data (in size). This reduces transmission time and intercommunications. This will decrease the number of intra-communications. Local RC at each cluster reduces task scheduling decision time. In this structure, instead of determining the amount of necessary data at each site in the cluster, we may use RC to select the optimum cluster in the DG. When a work is accepted to Data Grid, the Grid Scheduler delivers job data to cluster RCs. Each RC compares the needed data with the list of available data in the cluster and delivers the result to Grid Scheduler. Grid Scheduler chooses the cluster with the greatest data (by size) for job execution. The scheduling Strategy chooses the appropriate cluster and location for task execution [72].

2.11 Task scheduling algorithms

The same method may be used to classify the various task scheduling methods into one of three distinct categories in a four-level tree model. Scheduling scopes include within-site, within-cluster, and within-grid. When a job is submitted to a "star" grid environment, the grid manager uses its algorithm to nominate a resource from the grid environment to host the submitted task, and the work is scheduled to execute across a single scope. It's almost the same as the range of the intra-site scheduling technique in the four-level tree model. Algorithms for intra-site scheduling may take into account a wide variety of criteria. These considerations may or may not be made by the algorithm. Job scheduling in real-time is the main topic of this thesis. In contrast, elements of the system that are static, such as the speed of the processor and the kind of operating system, might be taken into account by this dynamic scheduling. It might also take into account dynamic parameters like the current CPU usage level [73].

The requirements of the job (hardware, network, and utilization requirements) and the capabilities of the available resources are always taken into account by the algorithm. When providing the two inputs, they should both take into account the same criteria (from the task and resources points of view).

Information on what's required to complete a job has been included in the data that's been sent in.

In the star grid architecture, the site's resource broker (or grid manager) looks through all the data about the available resources to select the best one. Both meta-heuristic approximation solutions and precise combinational solutions exist for the scheduling algorithm (which is, after all, a search algorithm). Artificial intelligence methods, such as the genetic algorithm, are used to approximate answers. Search techniques, such as linear and exhaustive ones, are used to get precise answers. The cluster will move the job to another server if it cannot handle it. If the cluster is unable to assign the job to one of its child sites, it will raise the issue to the grid. On the intra-grid level, the grid manager entrusts any cluster with the assignment, hoping that it would schedule the necessary resources from the sites with which it is associated. If the grid has trouble scheduling the job, it will decide that the job cannot be completed with the current set of resources [74].

2.11.1 Johnson's Algorithm

Johnson's algorithm determines a manufacturing or production system's best sequence or shortest pathways. Johnson's rule schedules tasks in two work center minimize makespan [39]. It eliminates downtime between work centers that outcomes aren't always ideal, particularly for a few tasks. Johnson's method reduces the time between starting work on one computer and finishing it on another. Job scheduling uses heuristics. Johnson's algorithm finds the best order. Johnson Algorithm is a flawless shortest route algorithm (SPP). Johnson Algorithm may solve combinatorial issues and can be used for processing n tasks on two or more computers.

According to Johnson, understanding the various pathways workflow process and projecting the least length from source to destination is vital in addressing the shortest-path issue. Johnson’s method may be employed in real value problems where the distance between two locations is examined, particularly when positive and negative weights are provided.

Scheduling assigns start and end times to machine shop orders and projects. Scheduling increases shop efficiency by providing a schedule for processing operations. It's arranging tasks on machines to improve performance measurements. In flow shop scheduling, 'n' jobs require m machines. The process sequence is the order in which machines process a work. All employment processes are the same. But machine processing times may vary. If a job's operation is missing, its processing time is presumed to be 0. Job shops make unique items in small amounts. Most workshop items need a specific set-up and processing sequence [21].

```
Algorithm (2.2): Johnson Algorithm
Input: Matrix of processing time for two servers (S1, S2)
Output: Sequences for all the processes
Begin
Step 1: For each task TT1, TT2, ..., TTn, determine the S1 and S2 times.
Step 2: Establish two queues, Q1 at the beginning of the schedule and Q2 at the end of the schedule.
Step 3. Examine all the S1 and S2 times, and determine the smallest.
Step 4. If the smallest time is S1, then insert the corresponding task at the first of Q1.
Otherwise, the smallest time is S2, and the corresponding task is inserted at the end of Q2.
Step 5: Delete that task from further consideration.
Step 6. Repeat steps 3-5 until all tasks have been assigned.
Step 7: Compute the total elapsed time (the time between beginning the first work and finishing the final job) and idle time.
End
```

Algorithm (2.2) Johnson's Sequencing Algorithm [75].

2.11.2 Shortest Job First scheduling

The process with the shortest time among those currently available in the ready queue will be scheduled next in SJFS scheduling. The SJFS (Shortest Job First) scheduling algorithm in which the CPU executes the job first has the shortest execution time. Also, the burst time is an important factor in SJF scheduling.

```
Algorithm (2. 3): Shortest First Job scheduling
Input: Matrix of processing time for two servers (S1, S2)
Output: Sequences for all the processes
Step 1: Sort all the processes according to their arrival time.
Step 2: Select the process with a minimum arrival time as well as minimum burst time.
Step 3: After completion of the process, select from the ready queue the process which has the minimum burst time.
Step 4: Repeat the above processes until all processes have finished their execution.
End
```

Algorithm (2.3) Pseudo steps of SJFS Algorithm[76].

2.11.3 First come first served (FCFS)

First Come First Serve (FCFS) or also known as First In First Out (FIFO) is the simplest and the most fundamental of grid scheduling that involves client-server interaction. In grid scheduling, FCFS policy manages the jobs based on their arrival time, which means that the first job will be processed first without other biases or preferences. The jobs are simply scheduled using the arrival time as the basis. The first job in the ready queue will receive the CPU first [77].

```
Algorithm (2.4): First come first served (FCFS)
Input: Matrix of processing time for two servers (S1, S2)
Output: Sequences for all the processes
Begin
Step 1: Input the processes along with their Burst time
Step 2: Find the waiting time (wt.) for all processes.
Step 3: As the first process that comes needs not to wait so
waiting time for process 1 will be 0
Step 4: Find waiting time for all other processes
End
```

Algorithm (2.4) First come first served (FCFS) [78].

2.12 Performance metrics

In this section, the three performance measures to choose the best replica will be explained :

2.12.1 CPU utilization

The server The CPU commonly referred to as the server processor, is an essential component of the server that handles commands and instructions. It is in charge of locating and carrying out instructions, processing data, and carrying out operations including providing web pages, executing database queries, and executing other computational and programming tasks.

The most crucial component of a server, the CPU, makes a significant contribution to the computing power of servers. The CPU has a big impact on how quickly a server responds. Use of the central unit CPU usage can increase significantly. In particular, if they can each entity interacts with the other in the same space, the allowed number of entities. When loaded, the system unexpectedly slows down the rate of sending updates to updates and responding to user input [79].

2.12.2 Bandwidth

The amount of data that can be transferred between a website, its clients, and its servers is referred to as bandwidth. It refers to the speed of the connection between the server and the website user. MB/s (Megabytes per Second) or GB/s (Gigabytes per Second) are the bandwidth units (Gigabytes per Second). Bandwidth is usually generated as part of monthly hosting services. A provider, for example, may offer 1GB of bandwidth per month or 5GB of bandwidth per 30 days.

As a shared resource, bandwidth and data access time are closely related. decreased network usage, which results in decreased data transfer and bandwidth usage in the distributed system. Data replication, which tries to provide proximity for data access to eliminate data transfer, is suggested as a way to reduce bandwidth use [80].

2.12.3 Memory size

Some computer memory bottlenecks do not have RAM Fast and/or not enough RAM. This reduces the speed of data delivery to the CPU, as a result, slowed the execution of processes. When the system is not enough, the stored data will be transferred to the hard disk through the swapping and swapping process to keep the processes running, which will greatly slow down the system. If the RAM cannot deliver data to the CPU fast enough, the device will experience a slowdown, so the CPU will usually enter the idle phase [81].

2.12.4 Waiting Time

The total time spent by the process in the ready state waiting for the CPU is referred to as waiting time [76].

$$\text{Waiting time} = \text{Turnaround time} - \text{Burst time} \qquad (2.2)$$

Since , Burst time is the total time taken by the process for its execution on the

CPU and the Turnaround time is the total amount of time spent by the process from coming in the ready state for the first time to its completion.As shown in Figure (2.6).

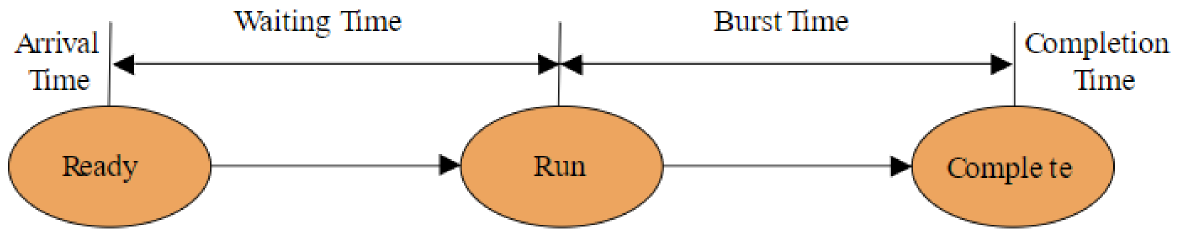


Figure 2.6 Waiting time associated with the Process[74].

2.12.5 Idle time

It is a time when an asset (a machine or an employee) is ready and available but not performing any work tasks. Every job that runs on a computer system consumes a certain amount of CPU processing time. If the CPU has finished all of its jobs, it is idle [82].

Idle time = Total elapsed time –Total busy time (2.3)

Chapter Three

The Proposed System Design

3.1 Introduction

This chapter describes the methodology and system design used for solving the described problem in chapter one. It presents the implementation of two factors. First, Best replica selection by using one of association rule algorithm. Second, Job scheduling between two servers by using one of scheduling algorithms. The adopted mechanism provides good performance on the servers in terms of replica selection, job scheduling in the data grid environment.

3.2 Architecture of the Proposed Methodology

The architecture of the proposed methodology involves two phases in data grid. The first is best replica selection phase which consists of sets main stages. These stages will be explained in detail. The second phase is job scheduling, which aims to perform all tasks on servers in the least amount of time. These steps are illustrated in Figure (3.1).

The proposed methodology was implemented using the Python programming language on the Google Colab environment. The work of the proposed methodology has been simulated programmatically.

3.3 Datasets

Database is IPv4 address networks from [83]. The type (format) is a string. It consists of 1500 IPv4(stable and unstable). Ping command applied to this dataset for 40 hours and got a set of IPv4 that worked at that time. Only the average response time has been taken Round Tirp Time(RTT) . The last update of this datasets is in 2019.

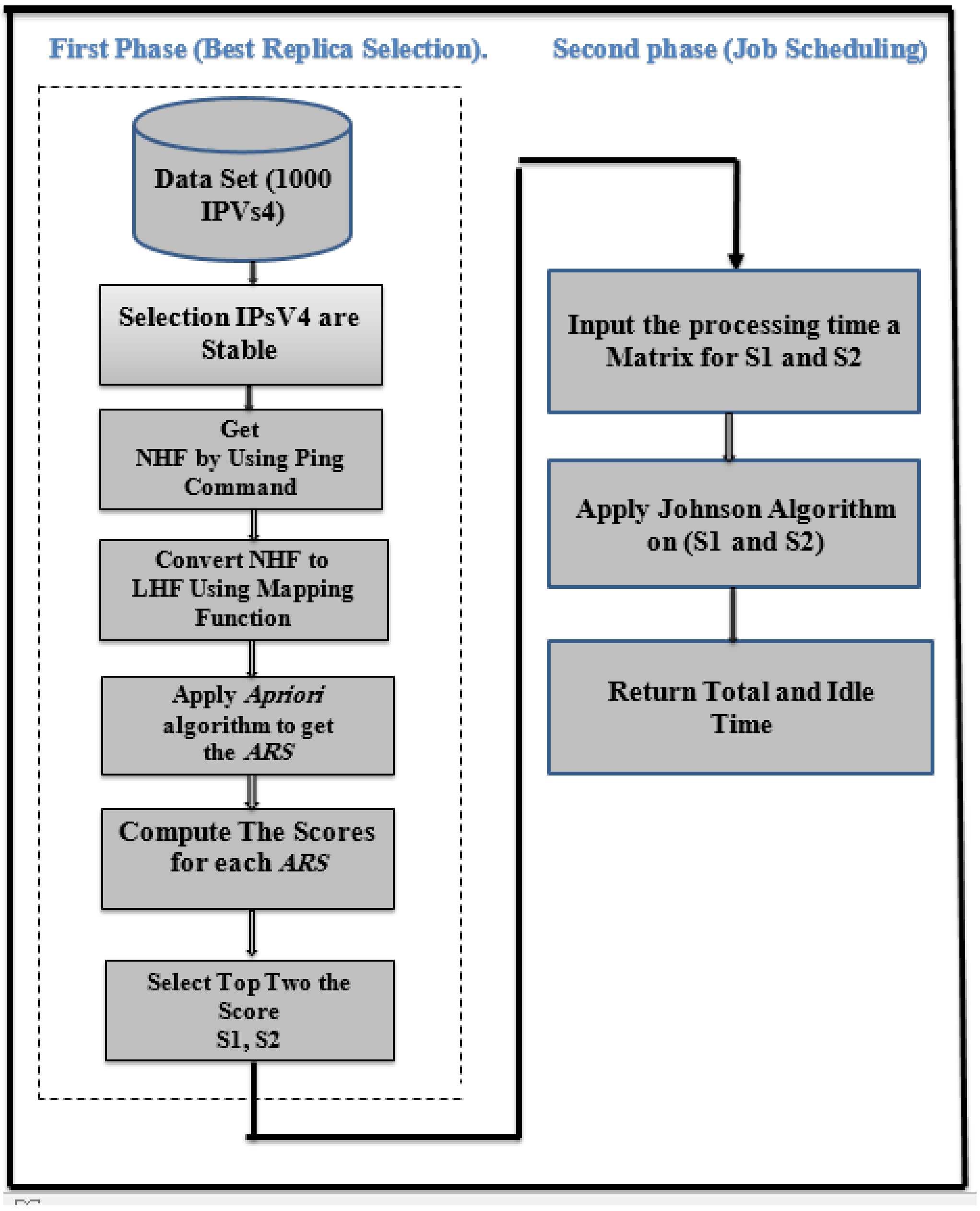
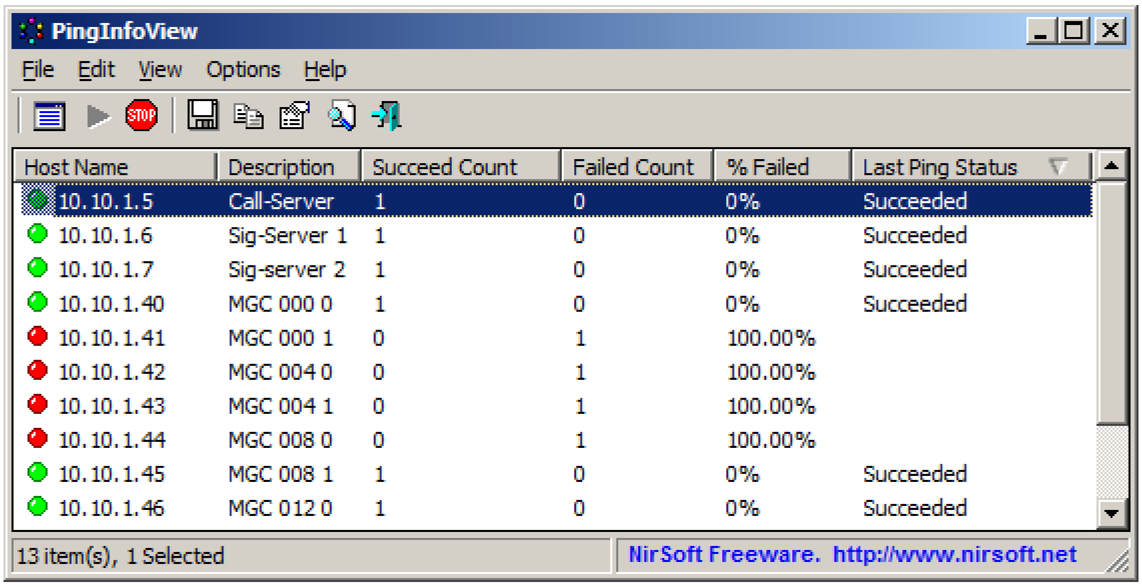

First Phase (Best Replica Selection).
Second phase (Job Scheduling)
Data Set (1000 IPVs4)
Selection IPsV4 are Stable
Get NHF by Using Ping Command
Convert NHF to LHF Using Mapping Function
Apply Apriori algorithm to get the ARS
Compute The Scores for each ARS
Select Top Two the Score S1, S2
Input the processing time a Matrix for S1 and S2
Apply Johnson Algorithm on (S1 and S2)
Return Total and Idle Time

Figure (3.1) General overview of the proposed system

3.4 Selection IPV4 (Sable).

PingInfoview tool was used to obtain the IPV4 stable. When 1500 IPV4 were examined at that time 41 stable (Succeeded) ones were obtained.



Host Name	Description	Succeed Count	Failed Count	% Failed	Last Ping Status
10.10.1.5	Call-Server	1	0	0%	Succeeded
10.10.1.6	Sig-Server 1	1	0	0%	Succeeded
10.10.1.7	Sig-server 2	1	0	0%	Succeeded
10.10.1.40	MGC 000 0	1	0	0%	Succeeded
10.10.1.41	MGC 000 1	0	1	100.00%	
10.10.1.42	MGC 004 0	0	1	100.00%	
10.10.1.43	MGC 004 1	0	1	100.00%	
10.10.1.44	MGC 008 0	0	1	100.00%	
10.10.1.45	MGC 008 1	1	0	0%	Succeeded
10.10.1.46	MGC 012 0	1	0	0%	Succeeded

13 item(s), 1 Selected
NirSoft Freeware. http://www.nirsoft.net

Figure 3. 2 PingInfoView Tool

3.5 Network History File (NHF)

After getting 41stable, ping is applied to each IP address to get (RTT). The ping command is iterated 30 times for each IP to get an array (NHF) consisting of 41 IPv4 (Column) and 30 (Row) RTT. The computing site receives periodic RTT from all replica sites before the selection process begins, and keeps the most recent ones in a log file called the Network History File (NHF).

3.5.1 Round Trip Time (RTT)

RTT is the time calculated in ms when a client starts sending a request and when it receives a reply from the server. It is an important performance metric for web applications and one of the most important factors in determining page loading times and network response time. RTT is measured with a ping, which is a command-line tool that calculates the time it takes to reach a user's device.

The study has 41 replica sites spread out over several locations. according to Table (3.1). It is necessary to calculate the RTT of the 41 IP servers first as shown in algorithm (3.1), where Si=S1, S2, S3, S4, S5, S6, S7, ... and S41 is the replica sites. After obtaining a matrix of RTTs for different periods, then saving the RTTs matrix in a file. After applied Algorithm 5 the result is as shown in Table (3.1).

```
Algorithm (3.1): Ping for IP address
Input: File of IPV4(1500)
Output: Network History File (NHF)
Step 1: Read each IP form file.txt
Step 2: ping for each IP
Step 3: return Round Trip Time(RTT) for each IP
Step 4: repeat ping 30 times for each IP
Step 5: save the result in file .xlsx
End Ping
```

Table 3. 1 Network History File (NHF)

Si no. / RTT	Si1	Si2	Si3	Si4	Si5	Si6	Si...	Si40	Si41
1	725	414	290	242	263	262	…	235	265
2	375	382	327	266	272	268	…	232	241
3	309	386	262	262	259	234	…	255	245
4	387	396	264	270	284	283	…	241	251
5	365	380	287	261	286	251	…	252	250
6	384	426	296	245	259	243	…	237	236
7	382	402	273	210	263	235	…	214	243
8	397	396	279	267	257	252	…	243	243
9	366	424	263	441	265	263	…	238	239
10	393	393	253	253	280	265	…	243	237
11	502	395	275	241	268	27	…	248	231
12	346	400	278	238	263	218	…	230	230
13	475	406	270	279	245	250	…	269	255
14	354	369	289	279	318	273	…	246	214
15	1187	391	337	254	266	261	…	252	250
16	386	367	241	237	294	283	…	275	230
17	375	404	283	277	265	271	…	227	235
19	342	405	258	214	307	260	…	221	236
.		.	.	.	.	.	.	.	.
.		.	.	.	.	.	.	.	.
30	1136	384	251	261	262	251	…	241	255

3.6 Mapping Function

It is a set of statistical measures consisting of statistical operations. Its main use is to analyze all the data in order to help get meaningful results. Statistical procedures have an important and effective role in scientific research, as they are used as basic rules to ensure that the results are consistent with the main objective.

```
Algorithm ( 3.2): Mapping Algorithm
Input: Network History File (NHF)
Output: Logical History File (LHF)
Begin
Step 1: Calculate the Mean of $RTT_{i,j}$ /* $i$ is number of RTT and $j$ is number of IP (site or replica)*/
    $MRTT_{i,j} = \sum_{k=j}^{l-1+j} \left(RTT_{i,j}\right)/2$                    (3.1)
Step 2: Calculate the Standard deviation of $RTT_{i,j}$ :
STDEV$_{i,j}$ $= \sqrt{\sum_{k=j}^{l-1+j} RTT_{i,j}\left(RTT_{i,k} - MRTT_{i,j}\right)^2/l}$        (3.2)
Step 3: Calculate the Variation
$Qi,j = STDEVi,j/MRTTi,j * 100$                    (3.3)
Step 4: Find a Coefficient Variation of Replica provider, $j$ using:
$AV_j = \sum_{i=1}^{M}\left(Q_{i,j}\right)/M$ /* $AV_j$ is the average of $Q_{i,j}$ */        (3.4)
Step 5: Classify links into stable and unstable using the condition:
$IF\left(AV_j < Q_{i,j}\right)$ then $\left(LV_{i,j} = 0\right)$
Otherwise,
$\left(LV_{i,j} = 1\right)$
  /* $LV$ represents a mapped Boolean Value of a $RTT$*/
6: LHF: = LV;
End NHF-Mapping
```

Algorithm 3. 2 Steps of the NHF-Mapping Algorithm.

RTT values are converted to logical values by using a mapping function, and saving the outcome in a Logical History File (LHF) and saved in the file **XLSX,** as illustrated in Table (3.2).

Table 3. 2 Logical Value LV

Si no. / **RTT**	**Time**	**Si1**	**Si2**	**Si3**	**Si4**	**Si5**	**Si6**	**Si...**	**Si49**	**Si41**
1	1	1	1	1	1	1	1	…	1	1
2	2	1	1	1	1	1	1	…	1	1
3	3	1	1	1	1	1	1	…	1	1
4	4	1	1	1	1	1	1	…	1	1
5	5	1	1	1	0	1	1	…	1	1
6	6	1	1	1	1	1	1	…	1	1
7	7	1	1	1	1	1	1	…	1	1
8	8	1	1	1	1	1	1	…	1	1
9	9	1	1	1	1	1	1	…	1	1
10	10	1	1	1	1	1	1	…	1	1
11	11	1	1	1	1	1	1	…	1	1
12	12	1	1	1	1	1	1	…	1	0
13	13	1	1	1	1	1	1	…	1	1
14	14	1	1	1	1	1	1	…	1	1
15	15	1	1	1	1	1	1	…	1	1
16	16	1	1	1	0	1	1	…	1	1
17	17	1	1	1	1	1	1	…	1	1
19	19	1	1	1	0	1	1	…	1	1
.	.							…		
.	.							…		
30	30	1	1	1	1	1	1	…	1	1

3.7 The Discovery of Association Rules

The Apriori algorithm is used in the mining process to obtain association rules Association Replica Sites(ARS). It is used to find similar locations associated with working on multiple projects simultaneously shorten the total time it takes to transfer the require file. So that the faster side will not be forced to wait for the slowest one. Apriori algorithm required to convert (NHF) to (LHF).

After applying the Apriori Algorithm (2.1) on Table (3.2). The output is (ARS) 26 IPS servers saved in the file **XLSX.**

3.8 Calculate the Score for ARS

The function of the scope is shown below based on the first three as shown in this equation;

$$\mathbf{Score\ i} = \mathbf{pi}^{\mathbf{BW}} \times \mathbf{wi}^{\mathbf{BW}} + \mathbf{pi}^{\mathbf{CPU}} \times \mathbf{wi}^{\mathbf{CPU}} + \mathbf{pi}^{\mathbf{I/0}} \times \mathbf{wi}^{\mathbf{I/0}} \qquad \textbf{(3.5)}$$

$$\mathbf{Wi}^{\mathbf{BW}} + \mathbf{Wi}^{\mathbf{CPU}} + \mathbf{Wi}^{\mathbf{I/0}} = \mathbf{1} \qquad \textbf{(3.6)}$$

Where

Score i : The score of the selection model corresponding to the server where $1 \leq i \leq n$ (n number of Jobs)

pi^{BW} : Calculating the amount of bandwidth that is accessible from server i to the computing site.

wi^{BW} : Network to Bandwidth ratio of a replica of server i which has been determined by the Organization's Director.

$\mathrm{pi}^{\mathrm{CPU}}$: The percentage of the CPU of the replica Server i.

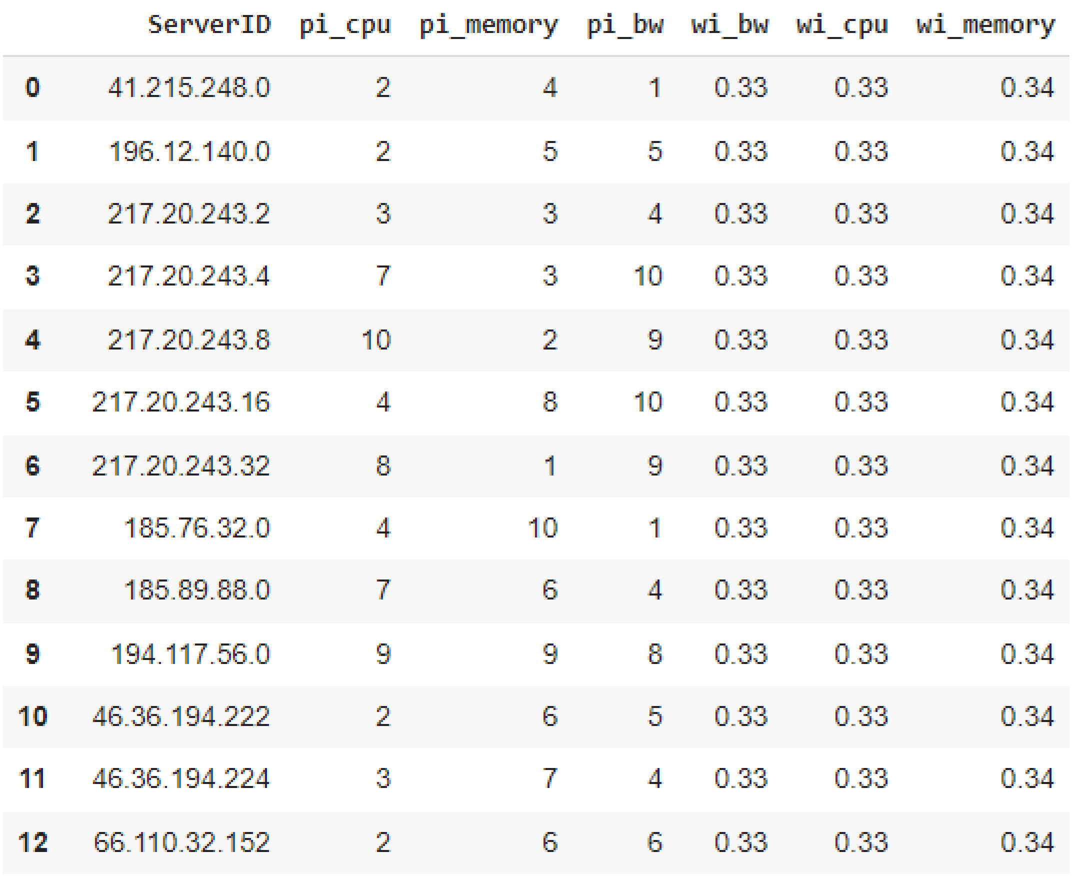
wi^{CPU} : The CPU load percentage determined by the Director of the Organization.

$\mathrm{pi}^{\mathrm{I/o}}$: percentage of free memory space of server i.

$wi^{1/0}$: The free memory percentage as determined by a representative of the data grid network.

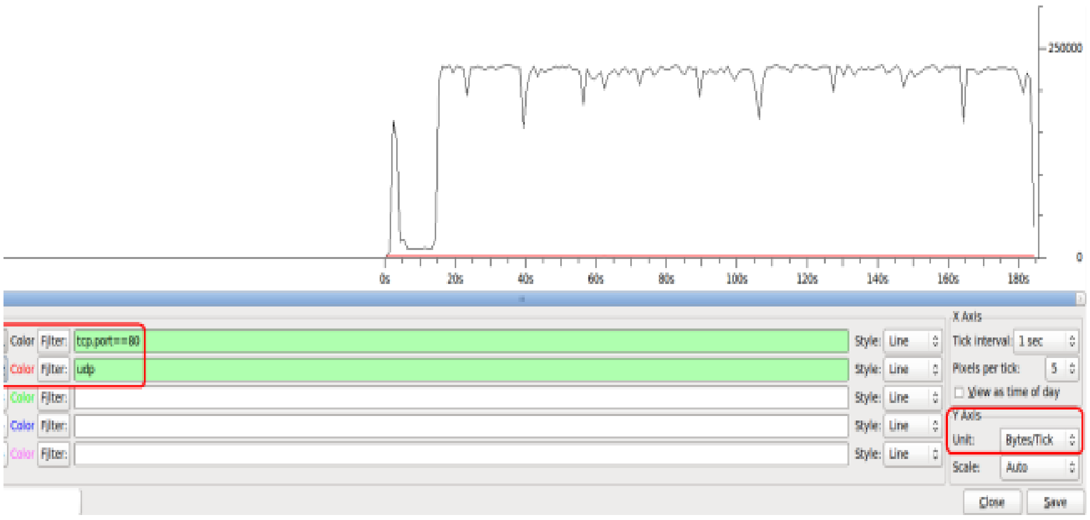
These Factors affecting represent the $wi^{\mathrm{BW}}, wi^{\mathrm{CPU}}$, and $wi^{\mathrm{I/0}}$ which represent the weights of all of the Bandwidth, CPU load, and I/O memory as shown in equation (3.5). All weights specified by the data network enterprise administrator according to his need.

Calculate (Memory size, Bandwidth, and CPU speed) for 26 IPS servers according to equation (3.5). The weights of (Memory size, and CPU speed) have been proposed because no server is giving these values for Security reasons. The CPU and memory size values are fixed for each server. As shown in Figure (3.3). The bandwidth was calculated real values by using Wireshark which is a free and open-source program for network eavesdropping and packet analysis and finding bandwidth. They are used for educational purposes. and used the one way to find bandwidth as shown in Figure (3.4).



	ServerID	pi_cpu	pi_memory	pi_bw	wi_bw	wi_cpu	wi_memory
0	41.215.248.0	2	4	1	0.33	0.33	0.34
1	196.12.140.0	2	5	5	0.33	0.33	0.34
2	217.20.243.2	3	3	4	0.33	0.33	0.34
3	217.20.243.4	7	3	10	0.33	0.33	0.34
4	217.20.243.8	10	2	9	0.33	0.33	0.34
5	217.20.243.16	4	8	10	0.33	0.33	0.34
6	217.20.243.32	8	1	9	0.33	0.33	0.34
7	185.76.32.0	4	10	1	0.33	0.33	0.34
8	185.89.88.0	7	6	4	0.33	0.33	0.34
9	194.117.56.0	9	9	8	0.33	0.33	0.34
10	46.36.194.222	2	6	5	0.33	0.33	0.34
11	46.36.194.224	3	7	4	0.33	0.33	0.34
12	66.110.32.152	2	6	6	0.33	0.33	0.34

Figure (3.3) The value and Weights for the three factors (CPU speed, memory size, Bandwidth).



Color Filter: tcp.port==80 Style: Line
Color Filter: udp Style: Line
Color Filter: Style: Line
Color Filter: Style: Line
Color Filter: Style: Line
Tick interval: 1 sec
Pixels per tick: 5
☐ View as time of day
Unit: Bytes/Tick
Scale: Auto

Figure 3. 4 Wireshark (method 3)

Then the score has been calculated for each job. Finally, find the average score for the server.

Suppose the primary agent works to provide services to his clients through his experience. Let the CPU speed be 30 %, and the memory size is 30 % but the bandwidth size of 40 % of what is available is sufficient. Table (3.3) Server 217.20.243.16 shows the processing time, bandwidth and memory capacity between each job (clients) and server.

Table 3. 3 Server 217.20.243.16

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	9	9	9	9	9	9	9	9	9
Memory capacity in GB	6	6	6	6	6	6	6	6	6
Bandwidth	45	89	110	190	95	140	100	53	90
Result *Score i*= Using Equation (3.5)	12	16.4	17.5	26.5	17	21.5	17.5	12.8	16.5
AVG Score i	**31.9**								

Table 3. 4 Server 66.110.32.152

Job	0	1	2	3	4	5	6	7	8
CPU speed	5	5	5	5	5	5	5	5	5
Memory capacity in GB	11	11	11	11	11	11	11	11	11
Bandwidth	300	200	132	150	195	100	120	98	270
Result *Score i*=Using Equation (3.5)	35.7	25.7	22.3	20.7	25.2	15.7	17.7	15.5	32.7
AVG Score i	**23.6**								

The Max of two Scores **(23.6, and 31.9)** are the best replica servers.

3.10 The processing time on the two servers.

In this part, the testing process is carried out on the basis of the assumption that there are 9 tasks and the appropriate processing time for each job is taken into account in an array, and the following is what the matrix looks like. Johnson's algorithm is applied and then calculates the total, idle and waiting time between the two servers.

Table 3. 5 The processing times

Task	Processing time on S1 66.110.32.152	Processing time on S2 217.20.243.16
J0	T01	T02
J1	T11	T12
J2	T21	T22
J3	T31	T32
J4	T41	T42
J5	T51	T52
J6	T61	T62
J7	T71	T72
J8	T81	T82

The processing time on S1 is ranging between 1 at the minimum and 77 at the maximum overall provided jobs. The processing time on S2 is ranging between 4 at a minimum and 71 as the maximum overall provided jobs. These processing times are generated automatically according to server specifications . A shown in Table (3.6).

Table 3. 6 Initial Processing Values

Task	Processing time on S1	Processing time on S2
J0	77	56
J1	33	4
J2	23	31
J3	62	40
J4	3	62
J5	26	71
J6	59	12
J7	6	32
J8	46	62

3.11 Applied Johnson Algorithm on two servers.

Apply the Johnson Sequencing Algorithm to two of the servers. This section discusses the flow of the system, which involves the Johnson Sequencing Algorithm and two servers.

To put into practice, the Johnson Algorithm, the following have been taken into consideration:

The first consideration to take into consideration: S1 and S2 will work through N tasks or requests in order on their respective servers (S1 and S2). The second consideration is that not for use on more than one task at once. Consideration number three: Once a work has been begun, it must be finished in its entirety. The fourth consideration to take is that each work order is in a state of readiness that enables it to be collected and processed. The fifth consideration to take is that moving work from one server to another scarcely takes any time at all. The design of our system details a total of 9 tasks and the associated processing times.

There are jobs 0,2, 3, …. and 9 to be processed through two servers (S1, S2). The goal is to come up with a workable solution that takes the least amount of time overall. This technique offers answers for n job 2 servers. Applied Johnson Algorithm (2.2) on the two servers.

The total elapsed time (the amount of time between the commencement of the first job and the end of the last work) and idle time (the amount of time the server is idle during the total elapsed time) should then be calculated.

When this matrix is obtained, the Johnson sequencing Algorithm is assumed to be used to get the optimal sequence.

Figure(3.5) contains two servers, one of which depends on the other. The out time for the first server is the entry for the second server. It also shows the server's idle time and waiting time for each process based on Johnson Algorithm.

Figure(3.5) Optimal Sequence based on Johnson Algorithm

Tasks sequences	**J4**	**J7**	**J2**	**J5**	**J8**	**J0**	**J3**	**J6**	**J1**

One way to determine total and idle time at the two servers is to construct chant chart. Thus , the group of jobs will takes 373 hours on the two servers to complete all jobs .The first server will take 335 hours.The second server is will idle 3 hours for its first job. The first job in the first server is no waiting. The fourth Job in the first server is started with 0 hour and out with 3 hours. The fourth Job in the second server is started with 3 hours and out with 65 hours. As shown in the Figure (3. 6) Gantt chart using the Johnson algorithm.

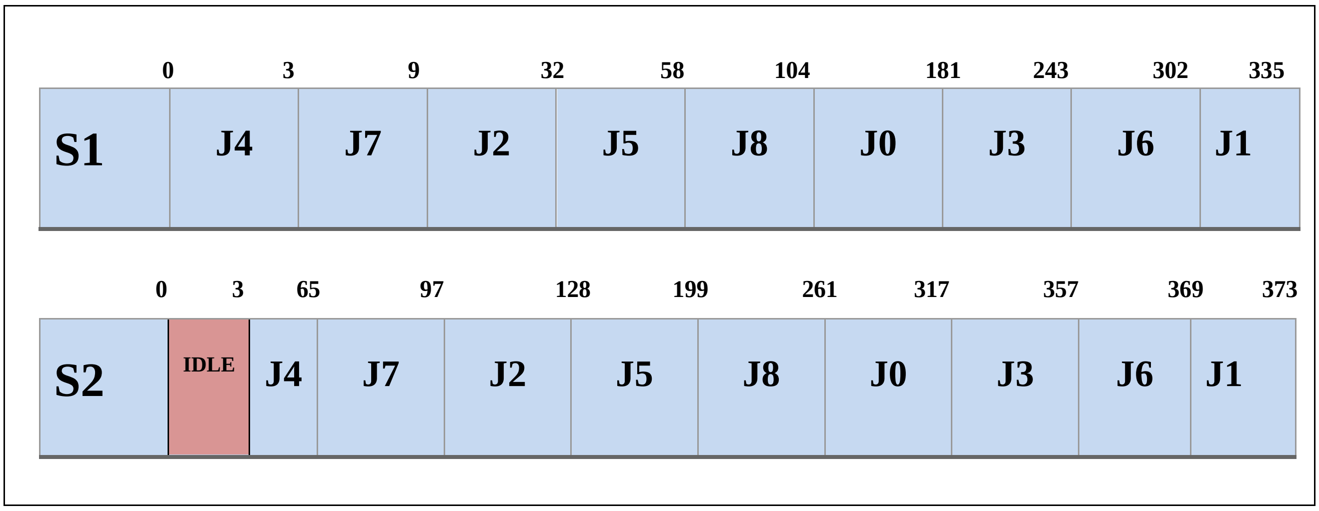


Figure 3. 6 Gantt chart using the Johnson algorithm (Idle Time)

Table (3.7) contains two servers, one of which depends on the other. The out time for the first server is the entry for the second server. It also shows the server's idle time and waiting time for each process based on Johnson Algorithm.

Table 3.7 Waiting and Idle Time for Servers S1 and S2 (Johnson Algorithm)

Tasks sequence	**Server S1**		**Server S2**		**Waiting time for S2**	**Idle time for Server S2**
	Time in	***Time out***	***Time in***	***Time out***		
J4	0	3	3	65	None	3
J7	3	9	65	97	56	0
J2	9	32	97	128	65	0
J5	32	58	128	199	70	0
J8	58	104	199	261	95	0
J0	104	181	261	317	80	0
J3	181	243	317	357	74	0
J6	243	302	357	369	55	0
J1	302	**335**	369	**373**	34	0

Total Flow Time: 373

Appling using equation (2.3)

=373-335

Idle time for server S1 =38

Another way: = 373-(77+33+23+62+3+26+59+6+46)

=373-335

=38

Appling using equation (2.3)

=373-(56+4+31+40+62+71+12+32+62)

=373+370

S2 Idle Time =3

Average Idle Time: 4.5

Waiting time for S1= 0+3+9+32+58+104+181+243+302

=932

Waiting time for S2=529

Average Waiting Time: 162.3

Chapter Four

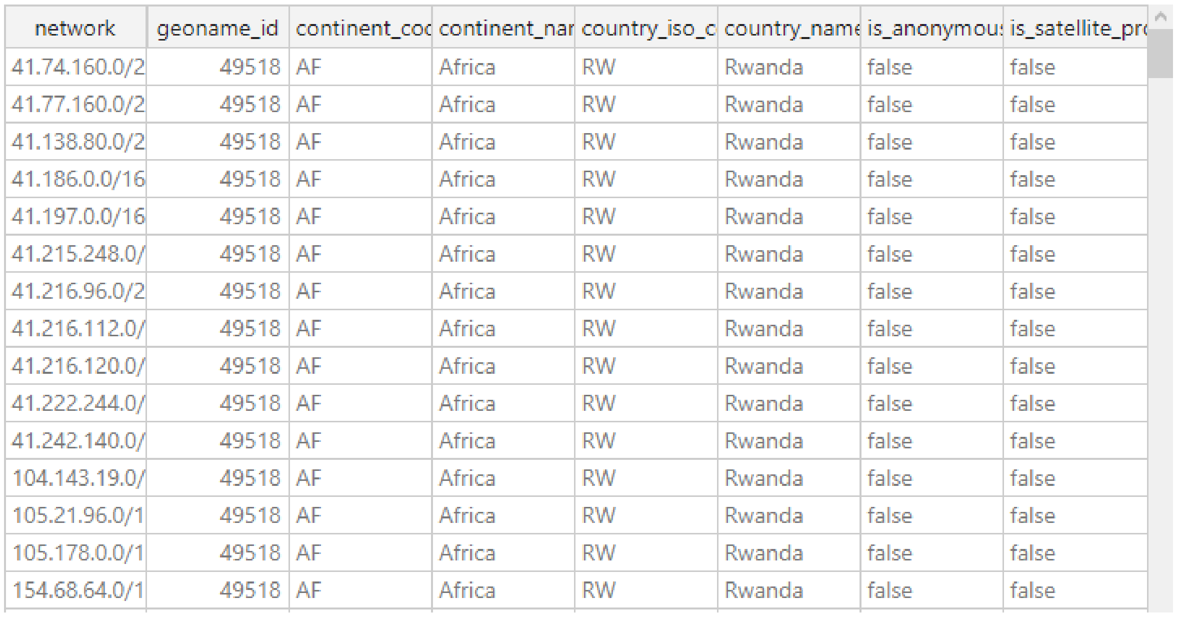
Results and Discussion

4.1 Introduction

This chapter presents the outcomes of the suggested system stages presented. It shows the role of implementing this system in the grid environment and seeing the results. The results of the experiments are detailed and discussed in this chapter.

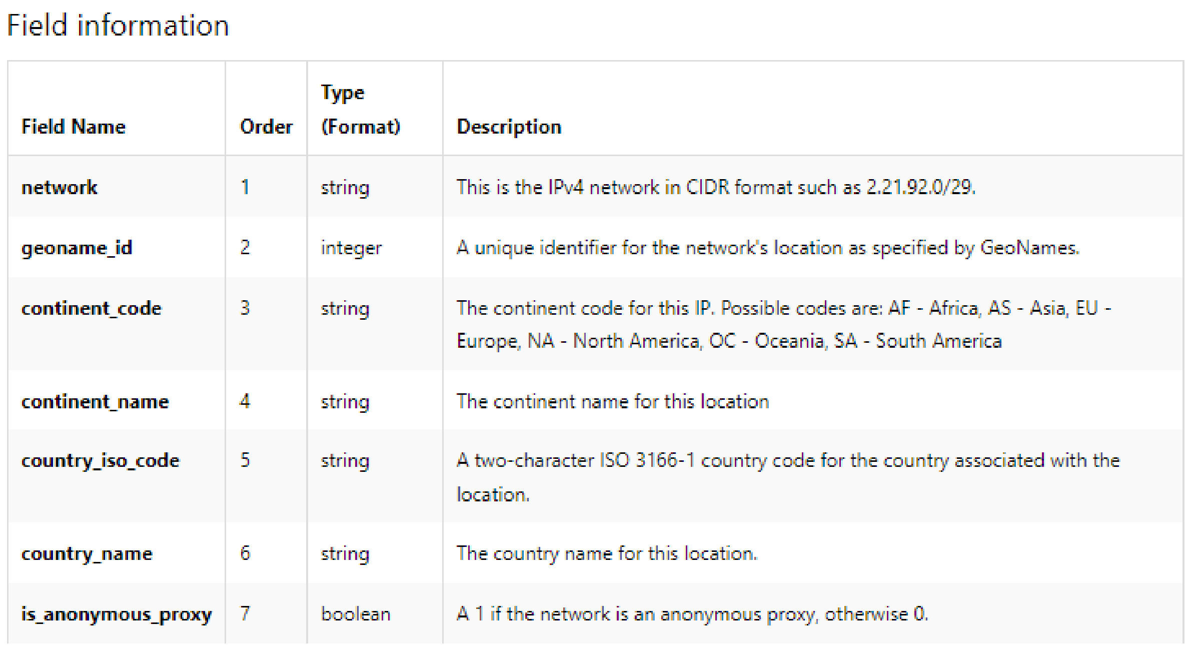
4.2 Description of IPV4 Dataset

Database of IPv4 address networks from [83]. Its real dataset. The type (format) is a string. It consists of 1500 IPv4. IPv4 address networks with the geographical location of each of them. It contains more than one field of information as shown in Figure (4.1). The network field was taken only because it is needed in this thesis. Its last update is in 2019.



network	geoname_id	continent_coc	continent_nar	country_iso_c	country_name	is_anonymou:	is_satellite_pr
41.74.160.0/2	49518	AF	Africa	RW	Rwanda	false	false
41.77.160.0/2	49518	AF	Africa	RW	Rwanda	false	false
41.138.80.0/2	49518	AF	Africa	RW	Rwanda	false	false
41.186.0.0/16	49518	AF	Africa	RW	Rwanda	false	false
41.197.0.0/16	49518	AF	Africa	RW	Rwanda	false	false
41.215.248.0/	49518	AF	Africa	RW	Rwanda	false	false
41.216.96.0/2	49518	AF	Africa	RW	Rwanda	false	false
41.216.112.0/	49518	AF	Africa	RW	Rwanda	false	false
41.216.120.0/	49518	AF	Africa	RW	Rwanda	false	false
41.222.244.0/	49518	AF	Africa	RW	Rwanda	false	false
41.242.140.0/	49518	AF	Africa	RW	Rwanda	false	false
104.143.19.0/	49518	AF	Africa	RW	Rwanda	false	false
105.21.96.0/1	49518	AF	Africa	RW	Rwanda	false	false
105.178.0.0/1	49518	AF	Africa	RW	Rwanda	false	false
154.68.64.0/1	49518	AF	Africa	RW	Rwanda	false	false

Figure (4.1) Sample of Dataset IPV4



Field information

Field Name	Order	Type (Format)	Description
network	1	string	This is the IPv4 network in CIDR format such as 2.21.92.0/29.
geoname_id	2	integer	A unique identifier for the network's location as specified by GeoNames.
continent_code	3	string	The continent code for this IP. Possible codes are: AF - Africa, AS - Asia, EU - Europe, NA - North America, OC - Oceania, SA - South America
continent_name	4	string	The continent name for this location
country_iso_code	5	string	A two-character ISO 3166-1 country code for the country associated with the location.
country_name	6	string	The country name for this location.
is_anonymous_proxy	7	boolean	A 1 if the network is an anonymous proxy, otherwise 0.

Figure (4.2) Field Information for Dataset IPV4 description

4.3 Developing Environment

For the purpose of developing and testing the project, the Python 3 programming language is selected due to its ease of coding, and the availability of free and wide libraries and packages. Furthermore, Google Colaboratory is selected as the developing environment due to its higher performance than regular and commercial personal computers since it gives a high level of CPU and GPU resources.

The libraries used in this work are pandas, CSV, numpy, apriori, association_rules, pip, os, time, xlsxwriter and openpyxl.

4.4 Best Replica Selections

Replication is the most significant technique for enhancing high data quality, dependability, and access speed in distributed systems. It is used to lower access time, expedite data transfer, and alleviate congestion caused by a large number of requesters. Because there are several copies of the requested file accessible at the time of the operation, selecting the most appropriate group of replicas is very advantageous.

41 IPsV4 out of 1000 IPV4 were obtained by examining them with the PingInfoView tool. The ping command is applied to obtain an RTT was applied. As shown in Figure (4.3)

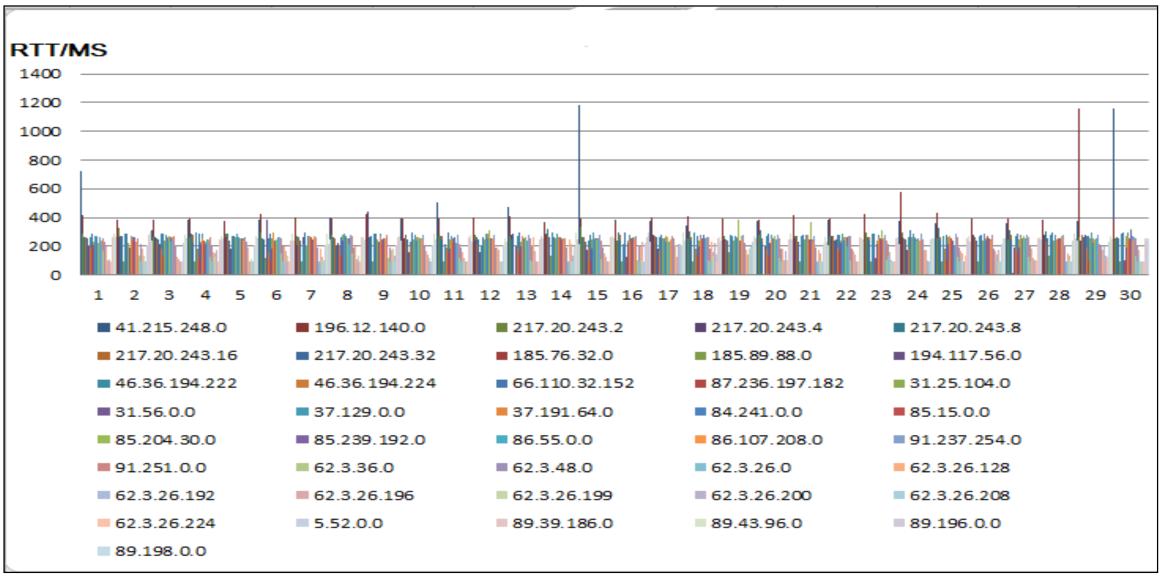


Figure 4. 3 Network History File(NHF) of 41 data providers(Stable).

4.5 Performance Improvement the Replica Selection Strategy

To reduce data retransmission and increase performance, Apriori Algorithm was used to obtains lists of sites with low latency (uncrowded lines). After applying the Apriori algorithm on 41 IPV4, Associated Replicas Sites was obtaining and their number are 26 IPV4.

To improve the selection mechanism in the data network architecture management system, the score equations (3.5) is applied on ARS based on three factors included in the selection criteria: bandwidth, CPU speed, and availability of memory factors. The result from the above explanation is the score for each IP after applying a score equation (3.5). As shown in the following Tables.

Table 4. 1 Server 196.12.140.0

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	11	11	11	11	11	11	11	11	11
Memory capacity in GB	9	9	9	9	9	9	9	9	9
Bandwidth	27	29	50	78	20	150	100	95	120
Result *Score i*=Using Equation (3.5)	10.5	12.4	14.5	17.3	11.5	24.5	19.5	19	21.5
AVG Score i	**7.2**								

Table 4.2 Server 217.20.243.2

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	3	3	3	3	3	3	3	3	3
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	155	150	7	8	2	3	17	170	166
Result *Score i*=Using Equation (3.5)	18.6	18.1	3.8	3.9	3.3	3.4	4.8	20.1	19.7
AVG Score i	**9.7**								

Table 4. 3 Server 217. 20.243.8

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	7	7	7	7	7	7	7	7	7
Memory capacity in GB	12	12	12	12	12	12	12	12	12
Bandwidth	20	39	66	145	195	100	85	29	30
Result *Score i*=Using Equation (3.5)	9.3	14.8	13.9	21.8	26.8	17.3	15.8	12.5	7.8
AVG Score i	**15.65**								

Table 4. 3 Server 41.215.248.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	2	2	2	2	2	2	2	2	2
Memory capacity in GB	4	4	4	4	4	4	4	4	4
Bandwidth	1.6	1.1	1.8	1.9	7.3	2.4	1.8	2.1	1.7
Result *Score i*=Using Equation (3.5)	1.0	3.4	2.5	2.5	9.6	2.7	2.5	2.6	2.4
AVG Score i	**3.24**								

Table 4.5 Server 31.25.104.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	3	3	3	3	3	3	3	3	3
Memory capacity in GB	2	2	2	2	2	2	2	2	2
Bandwidth	25	24.1	7.1	8.2	2.5	3.7	1.7	17	16.6
Result *Score i*=Using Equation (3.5)	10.1	9.7	4	4.3	2.4	2.8	2.1	7.3	7.2
AVG Score i	**5.54**								

Table 4.6 Server 87.236.197.182

Job	0	1	2	3	4	5	6	7	8
CPU speed	2	2	2	2	2	2	2	2	2
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	12	12.9	8.2	14	2.7	3.5	11	12.3	16
Result *Score i*=Using Equation (3.5)	6.3	6.6	5	7	2.2	3.4	6	6.4	7.7
AVG Score i	**5.6**								

Table 4.7 Server 46.36.194.224

Job	0	1	2	3	4	5	6	7	8
CPU speed	3	3	3	3	3	3	3	3	3
Memory capacity in GB	4	4	4	4	4	4	4	4	4
Bandwidth	17	16.1	18.3	20.5	2	3	17	10	36
Result *Score i*=Using Equation (3.5)	8	7.7	8.5	9.2	2.9	3.2	8	5.7	14.5
AVG Score i	**7.84**								

Table 4.8 Server 46.36.194.222

Job	0	1	2	3	4	5	6	7	8
CPU speed	3	3	3	3	3	3	3	3	3
Memory capacity in GB	7	7	7	7	7	7	7	7	7
Bandwidth	20.7	3	7	8.1	12	3	19	17	37
Result *Score i*=Using Equation (3.5)	10.3	4.3	5.6	6	7.3	4.3	9.7	9	15.8
AVG Score i	**8**								

Table 4.9 Server 46.36.194.222

Job	0	1	2	3	4	5	6	7	8
CPU speed	10	10	10	10	10	10	10	10	10
Memory capacity in GB	2	2	2	2	2	2	2	2	2
Bandwidth	33.1	15.2	4	4.9	3.5	7	13	18.1	23.6
Result *Score i* =Using Equation (3.5)	15.1	9	5.2	5.5	5	6.2	8.3	10	10.5
AVG Score i	**8.3**								

Table 4.10 Server 194.117.56.0

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	4	4	4	4	4	4	4	4	4
Memory capacity in GB	8	8	8	8	8	8	8	8	8
Bandwidth	22.5	16	2	1.5	3	4	2	1.7	1.2
Result *Score i* =Using Equation (3.5)	11.5	9.3	4.5	4.4	4.9	5.2	4.5	9.6	7.9
AVG Score i	**6.86**								

Table 4.11 Server 91.237.254.0

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	2	2	2	2	2	2	2	2	2
Memory capacity in GB	10	10	10	10	10	10	10	10	10
Bandwidth	37.1	13.2	1	4	2	3	12	1.8	15
Result *Score i* =Using Equation (3.5)	16.5	8.3	4.2	5.2	4.5	4.9	7.9	4.5	9
AVG Score i	**6.6**								

Table 4.12 Server 86.107.28.0

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	8	8	8	8	8	8	8	8	8
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	29	22.1	1.7	8	2	3	1.5	8.2	10
Result *Score i* =Using Equation (3.5)	14	11.7	4.7	6.9	4.8	5.2	4.7	6.9	7.9
AVG Score i	**7.4**								

Table 4.13 Server 85.239.192.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	8	8	8	8	8	8	8	8	8
Memory capacity in GB	2	2	2	2	2	2	2	2	2
Bandwidth	15	100	7.1	8.3	22	3	11	17	16
Result *Score i* =Using Equation (3.5)	8.4	37.3	5.7	6.1	10.7	4.3	7	9	9.9
AVG Score i	**10.93**								

Table 4.14 Server 85.204.30.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	4	4	4	4	4	4	4	4	4
Memory capacity in GB	10	10	10	10	10	10	10	10	10
Bandwidth	1.9	8.3	2.7	18	2.1	3.9	36	4.3	6
Result *Score i* =Using Equation (3.5)	18.6	18.1	3.8	3.9	3.3	3.4	4.8	20.1	19.7
AVG Score i	**9.7**								

Table 4.15 Server 85.15.0.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	7	7	7	7	7	7	7	7	7
Memory capacity in GB	6	6	6	6	6	6	6	6	6
Bandwidth	66.5	12	7	3	2	13	2.1	3.2	7
Result *Score i* =Using Equation (3.5)	27.2	8.6	6.9	5.6	5.2	9	5.3	5.6	6.9
AVG Score i	**8.9**								

Table 4.16 Server 84.241.0.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	9	9	9	9	9	9	9	9	9
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	1.99	15	7.3	12.4	2.5	4.1	33.1	2.9	77
Result *Score i* =Using Equation (3.5)	5.2	9.7	7	8.8	5.4	9	38	5.5	30.7
AVG Score i	**13.24**								

Table 4.17 Server 37.129.64.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	6	6	6	6	6	6	6	6	6
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	11.3	10	99	38.9	30.5	45.2	17	4	1
Result *Score i* =Using Equation (3.5)	7.4	7	37.2	16.8	13.9	18.9	9.3	4.9	3.9
AVG Score i	**13.25**								

Table 4.18 Server 37.129.0.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	1	1	1	1	1	1	1	1	1
Memory capacity in GB	7	7	7	7	7	7	7	7	7
Bandwidth	12.3	2.9	37	52	4	27.1	100	12.9	19.5
Result *Score i* =Using Equation (3.5)	6.7	3.5	15.1	20.2	3.9	11.8	36.6	6.9	9
AVG Score i	**12.6**								

Table 4.19 Server 31.156.0.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	2	2	2	2	2	2	2	2	2
Memory capacity in GB	6	6	6	6	6	6	6	6	6
Bandwidth	25	1	12.3	6.2	2.5	3	17	17.5	9
Result *Score i* =Using Equation (3.5)	11.1	2.9	6.7	4.7	4.4	3.6	8.3	8.5	5.6
AVG Score i	**6.2**								

Table 4.20 Server 185.89.88.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	9	9	9	9	9	9	9	9	9
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	88.1	30.2	13.3	4.1	3.2	3.4	42.4	7.3	55
Result *Score i* =Using Equation (3.5)	34.5	14.8	9.1	5.9	5.6	5.7	19	7	23.3
AVG Score i	**13.9**								

Table 4.21 Server 217.20.240

Job	0	1	2	3	4	5	6	7	8
CPU speed	7	7	7	7	7	7	7	7	7
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	11.9	122	7	4.2	2	3	2.3	20.5	4
Result *Score i* =Using Equation (3.5)	7.9	45.3	6.2	5.3	4.5	4.9	6.4	10.8	5.2
AVG Score i	**10.7**								

Table 4.22 Server 41.215.248.0

Job	0	1	2	3	4	5	6	7	8
CPU speed	6	6	6	6	6	6	6	6	6
Memory capacity in GB	7	7	7	7	7	7	7	7	7
Bandwidth	111	15	6.1	7.3	20	32	11	19	16.1
Result *Score i* =Using Equation (3.5)	41.9	9.3	6.2	6.6	11	15	7.9	10.6	9.6
AVG Score i	**9.7**								

Table 4.23 Server 217.20.243.32

Job	0	1	2	3	4	5	6	7	8
CPU speed	4	4	4	4	4	4	4	4	4
Memory capacity in GB	5	5	5	5	5	5	5	5	5
Bandwidth	40.9	22	7.3	86	20.1	30.8	12	11	12
Result *Score i* =Using Equation (3.5)	16.8	10.3	26.8	32.1	9.7	13.3	6.9	20.1	6.9
AVG Score i	**14.2**								

Table 4.24 Server 217.20.243.2

Job	0	1	2	3	4	5	6	7	8
CPU speed	3	3	3	3	3	3	3	3	3
Memory capacity in GB	8	8	8	8	8	8	8	8	8
Bandwidth	12	17.1	7	6.2	21	3.9	2	17	2.3
Result *Score i* =Using Equation (3.5)	7.6	9.4	5.9	5.7	10.7	4.9	4.2	9.3	4.8
AVG Score i	**6.9**								

Table 4.25 Server 217.20.243.16

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	9	9	9	9	9	9	9	9	9
Memory capacity in GB	6	6	6	6	6	6	6	6	6
Bandwidth	45	89	110	190	95	140	100	53	90
Result *Score i*=Using Equation (3.5)	12	16.4	17.5	26.5	17	21.5	17.5	12.8	16.5
AVG Score i	**31.9**								

Table 4.26 Server 66.110.32.152

Job	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
CPU speed	5	5	5	5	5	5	5	5	5
Memory capacity in GB	11	11	11	11	11	11	11	11	11
Bandwidth	300	200	132	150	195	100	120	98	270
Result *Score i*=Using Equation (3.5)	35.7	25.7	22.3	20.7	25.2	15.7	17.7	15.5	32.7
AVG Score i	**23.6**								

As shown in Figure (4.4) The score results for each ARS.

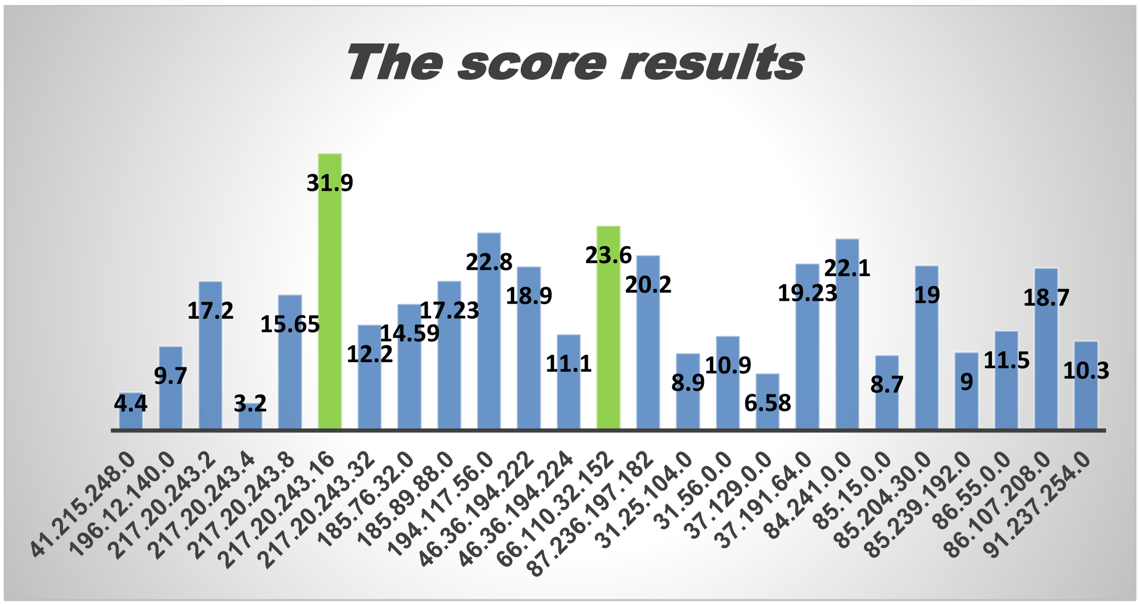


Figure 4. 4 The Scores Results of IPs contain Top Two Score (31.9,23.6)

4.6 Job Scheduling

One of the greatest obstacles is knowing how to arrange work effectively and efficiently. The main objective of Job scheduling is to reduce task completion and idle times and increase system output.

Because the current work requires two servers, the top two scores for the servers are chosen (217.20.243.16,66.110.32.152). Configuration of the two servers was done programmatically. An array of 9 tasks, each task has a processing time on server1 and a processing time on server2. To reduce total and idle time for all 9 jobs Johnson algorithm was applied to obtain the optimal sequence to execute all jobs in the least time. Figure (4.5) *shown Processing times on server1 and server2*.

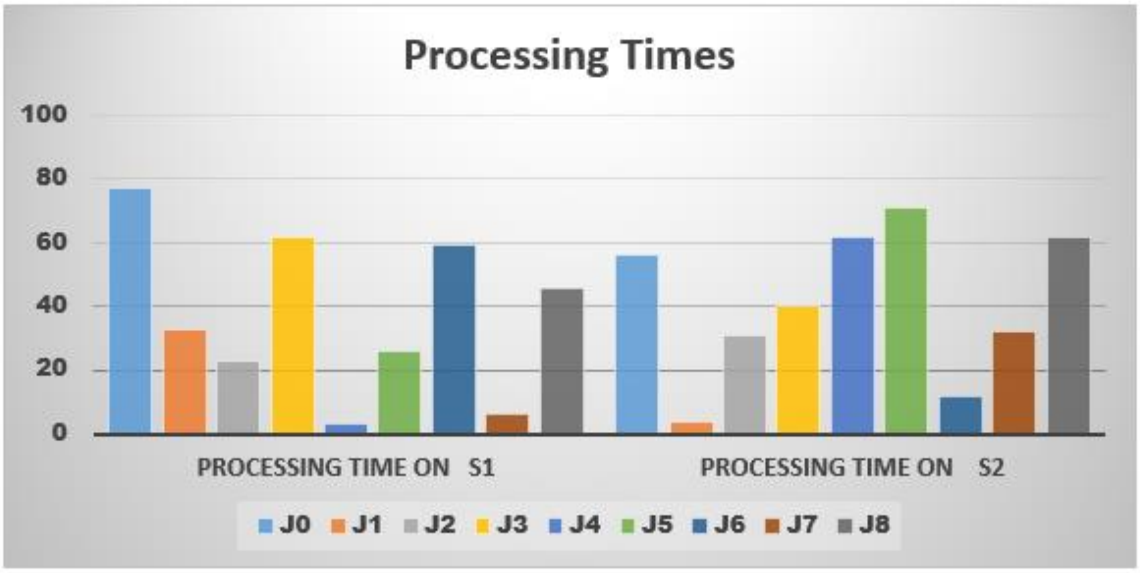


Figure 4. 5 Processing times on server1 and server2

Johnson's algorithm was applied for job scheduling to obtain the least total and idle time for jobs on the two servers. As shown in Figure 4. 6

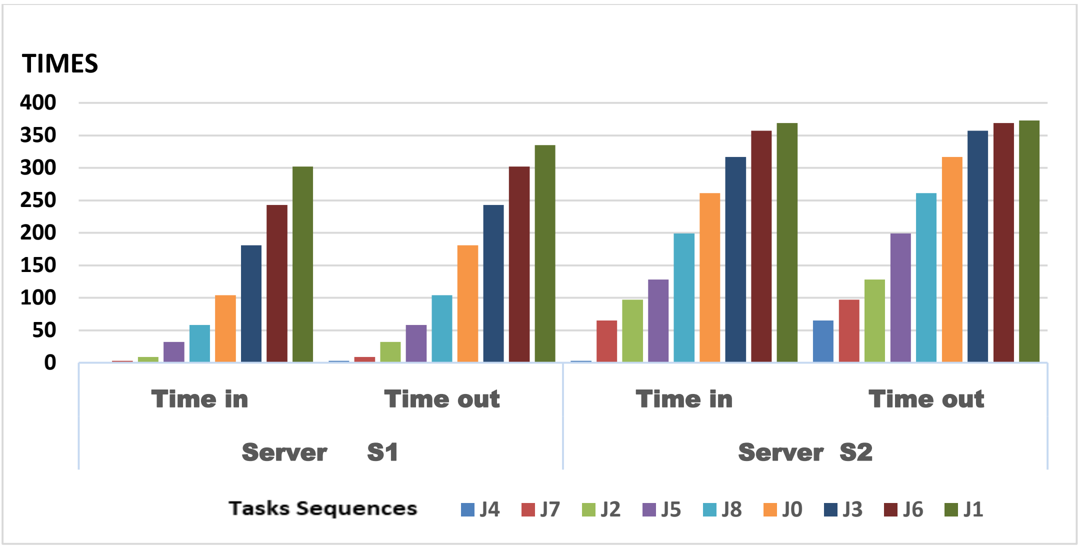


Figure 4. 6 Interferences Time between Two Servers Using Johnson Algorithm

To prove that Johnson's algorithm is the best for finding the minimum total time and idle time between two servers compare it with the SJFS algorithm.

4.6.1 Job Secheduling using Shortest First Job scheduling

The process with the shortest time among those currently available in the ready queue will be scheduled next in SJF scheduling. After applied algorithm (2.3) the results are the the following: Based on Figure (4.7) the following sequence according to SJF scheduling:

Figure (4.7) The Sequence based on SJF Algorithm

Tasks Sequence	**J4**	**J7**	**J2**	**J5**	**J1**	**J8**	**J6**	**J3**	**J0**

One way to determine total and idle time at the two servers is to construct chant chart. Thus , the group of jobs will takes 391 hours on the two servers to complete all jobs .The first server will take 335 hours.The second server is will idle 3 hours for its first job. The first job in the first server is no waiting. The fourth Job in the first server is started with 0 hour and out with 3 hours. The fourth Job in the second server is started with 3 hours and out with 65 hours. As shown in the Figure (4.8) Gantt chart using the (SJFS) algorithm.

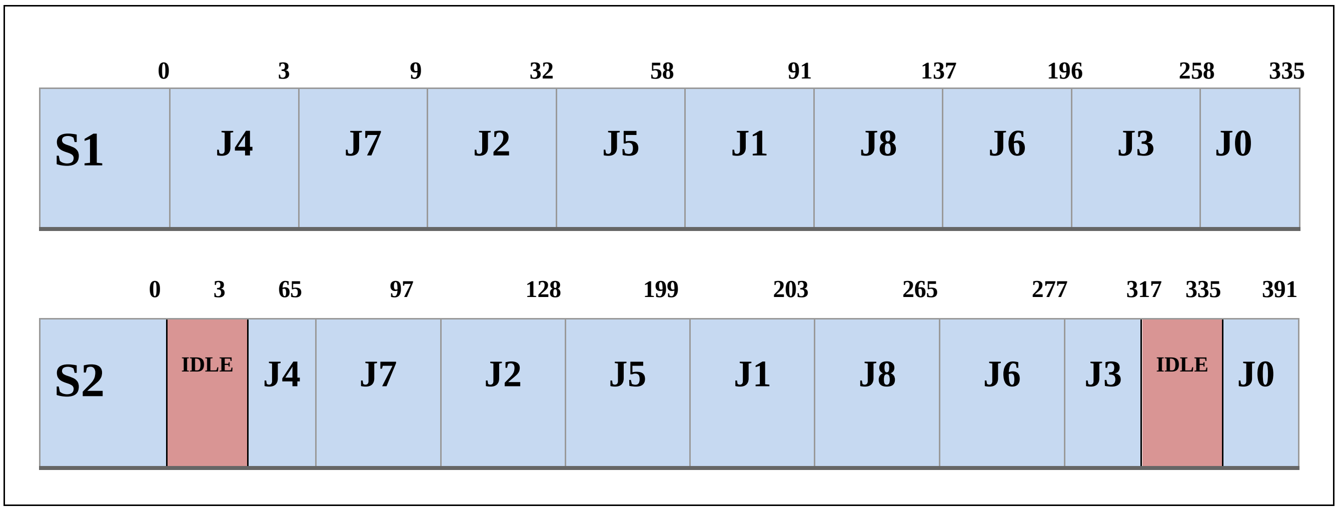


Figure (4.8) Gantt chart using the (SJFS) algorithm (Idle Time)

Table (4.27) contains two servers, one of which depends on the other. The out time for the first server is the entry for the second server. It also shows the server's idle time and waiting time for each process based on SJF Algorithm.

Table 4.27 Waiting and Idle Time for Servers S1 and S2 using the (SJFS) algorithm

Tasks sequence	**Server S1**		**Server S2**		**Waiting time for S2**	**Idle time for Server S2**
	Time in	***Time out***	***Time in***	***Time out***		
J4	0	3	3	65	None	3
J7	3	9	65	97	56	0
J2	9	32	97	128	65	0
J5	32	58	128	199	70	0
J1	58	91	199	203	108	0
J8	91	137	203	265	66	0
J6	137	196	265	277	69	0

J3	196	258	277	317	19	0
J0	258	**335**	335	**391**	0	18

Total Flow Time: 391

Appling using equation (2.3)

=391-335

S1 Idle Time: =56

Appling using equation (2.3)

  = 391-370

S2 Idle Time: =21

Average Idle Time: 8.5

Waiting time for S1= (0+3+9+32+58+91+137+196+258)

 =784

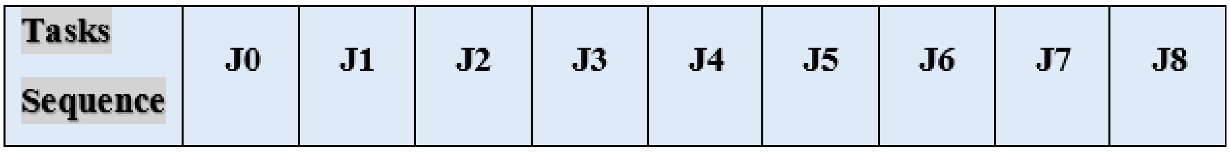
Waiting time for S2=453

Average Waiting Time: 137.4

4.6.2 Job Secheduling using First come first served (FCFS)

The jobs are simply scheduled using the arrival time as the basis. The first job in the ready queue will receive the CPU first.After applied algorithm (2.4) the results are the the following:

Figure 4.9 The sequencing based on FCFS



Tasks Sequence	J0	J1	J2	J3	J4	J5	J6	J7	J8

One way to determine total and idle time at the two servers is to construct chant chart. Thus , the group of jobs will takes 474 hours on the two servers to complete all jobs .The first server will take 335 hours.The second server is will idle 77 hours for its first job. The first job in the first server is no waiting. The fourth Job in the first server is started with 0 hour and out with 77 hours. The fourth Job in the second server is started with 77 hours and out with 133 hours. As shown in the Figure (4.10) Gantt chart using the (FCFS) algorithm.

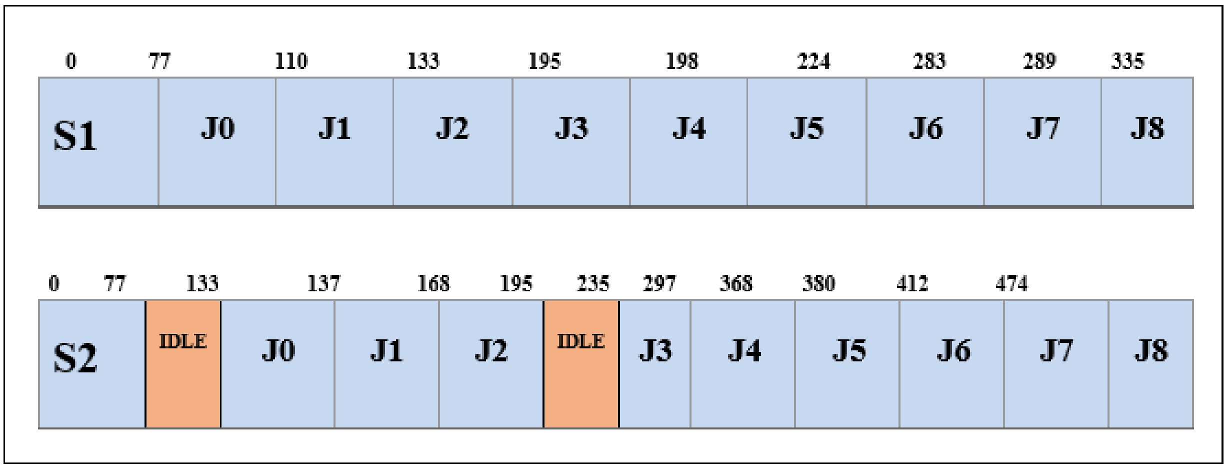


Figure 4.10 Gantt chart using the (FCFS) algorithm (Idle Time)

Table (4.28) contains two servers, one of which depends on the other. The out time for the first server is the entry for the second server. It also shows the server's idle time and waiting time for each process based on FCFS Algorithm.

Table 4.28 Waiting and Idle Time for Servers S1 and S2 by (FCFS) algorithm

Tasks sequence	**Server S1**		**Server S2**		**Waiting time for S2**	**Idle time for Server S2**
	Time in	***Time out***	***Time in***	***Time out***		
J0	0	77	77	133	None	77
J1	77	110	133	137	23	0
J2	110	133	137	168	4	0
J3	133	195	195	235	0	27
J4	195	198	235	297	37	0
J5	198	224	297	368	73	0
J6	224	283	368	380	85	0
J7	283	289	380	412	91	0
J8	289	335	412	474	77	0

Total Flow Time: 474

Appling using equation (2.3)

=474-335

S1 Idle Time: =139

Appling using equation (2.3)

S2 Idle Time: =104

Average Idle Time: 27

Waiting time for S1= (0+77+110+133+195+198+224+283+289)

=1509

Waiting time for S2=208

Average Waiting Time: 190.7

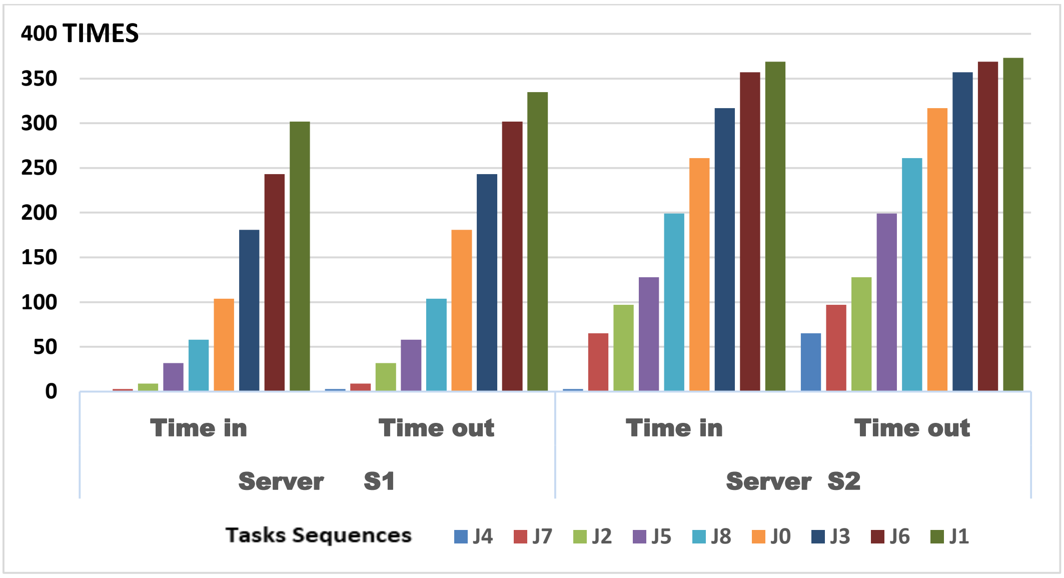


Figure 4.11 Interferences Time between Two Servers Using SJFS Algorithm

Through the Figure 4.11, it was found that the total and idle time are higher than the Johnson algorithm. Although the waiting time is less than Johnson's, the idle time has more importance in our calculations than the waiting time. Then it was compared with the FCFS algorithm. The result in the Figure (4.12):

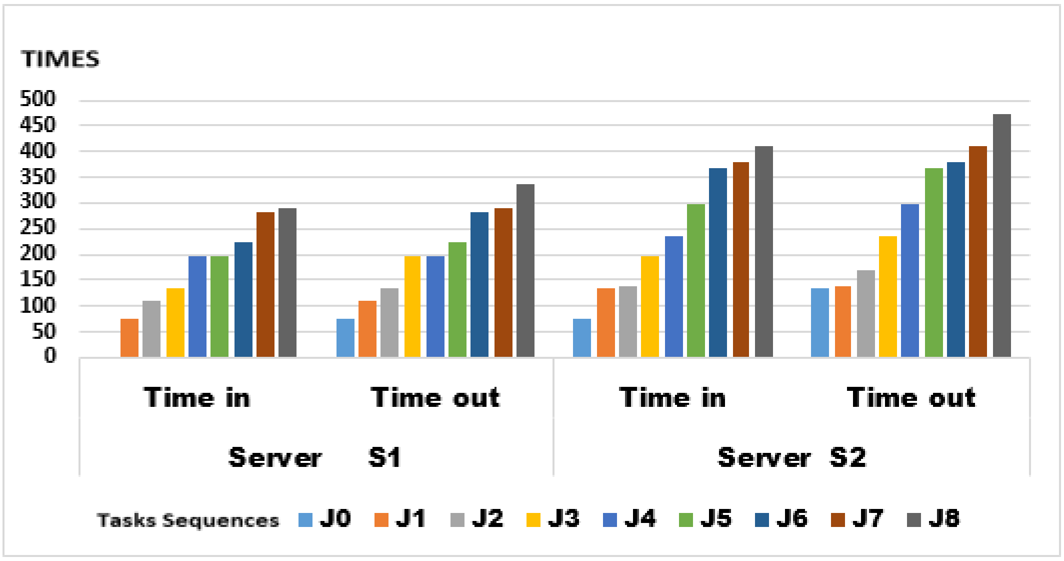


Figure 4. 12 Interferences Time between Two Servers Using (FCFS)

Through the work of the results, it became clear that the Johnson algorithm is the best in reducing the total time and the idle time.

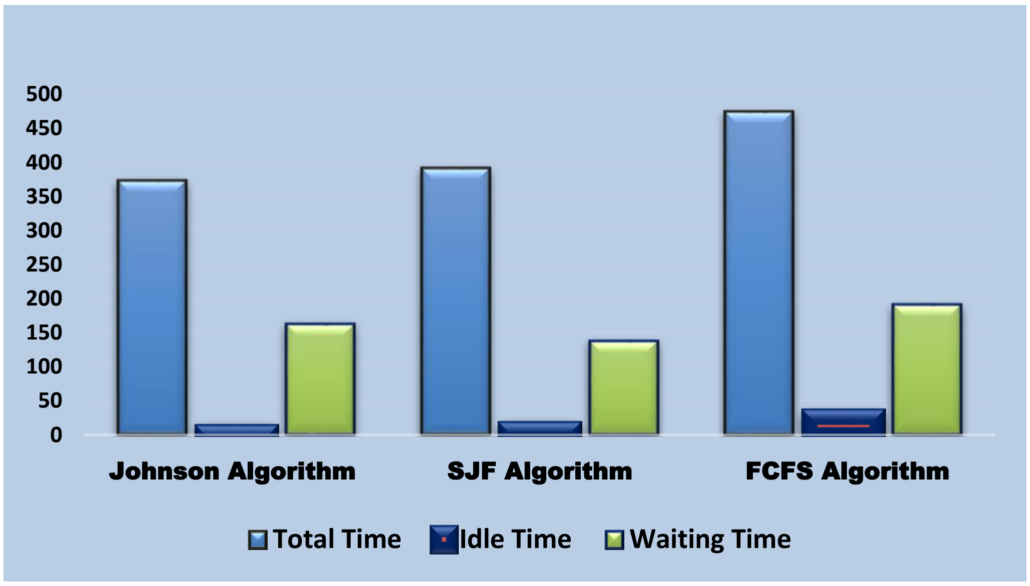
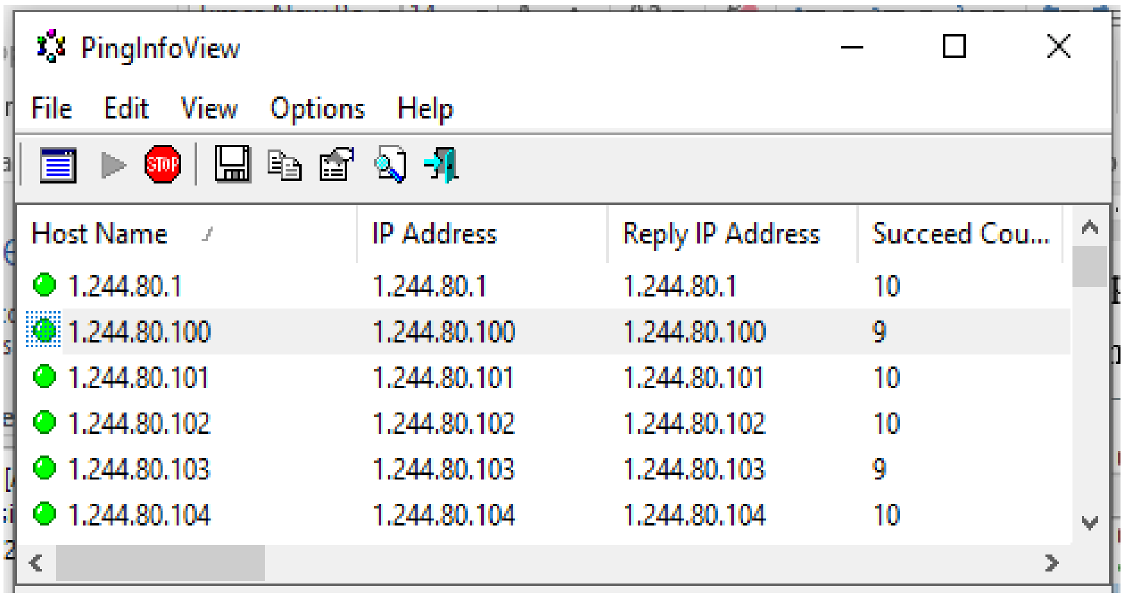


Figure 4. 13 Final Results on two servers.

4.7 Results of the Second Experiment to Select the Best Replication

A second experiment was conducted on 1500 IPV4 later time. Ping was applied to each IP 25 times to obtain RTT. The IP address was checked with the PingInfoview tool as shown in figure (4.16).



Host Name	IP Address	Reply IP Address	Succeed Cou...
1.244.80.1	1.244.80.1	1.244.80.1	10
1.244.80.100	1.244.80.100	1.244.80.100	9
1.244.80.101	1.244.80.101	1.244.80.101	10
1.244.80.102	1.244.80.102	1.244.80.102	10
1.244.80.103	1.244.80.103	1.244.80.103	9
1.244.80.104	1.244.80.104	1.244.80.104	10

Figure (4.14) The PingInfoview tool Results

58 IPv4 stables were found. As shown in Figure (4.15).

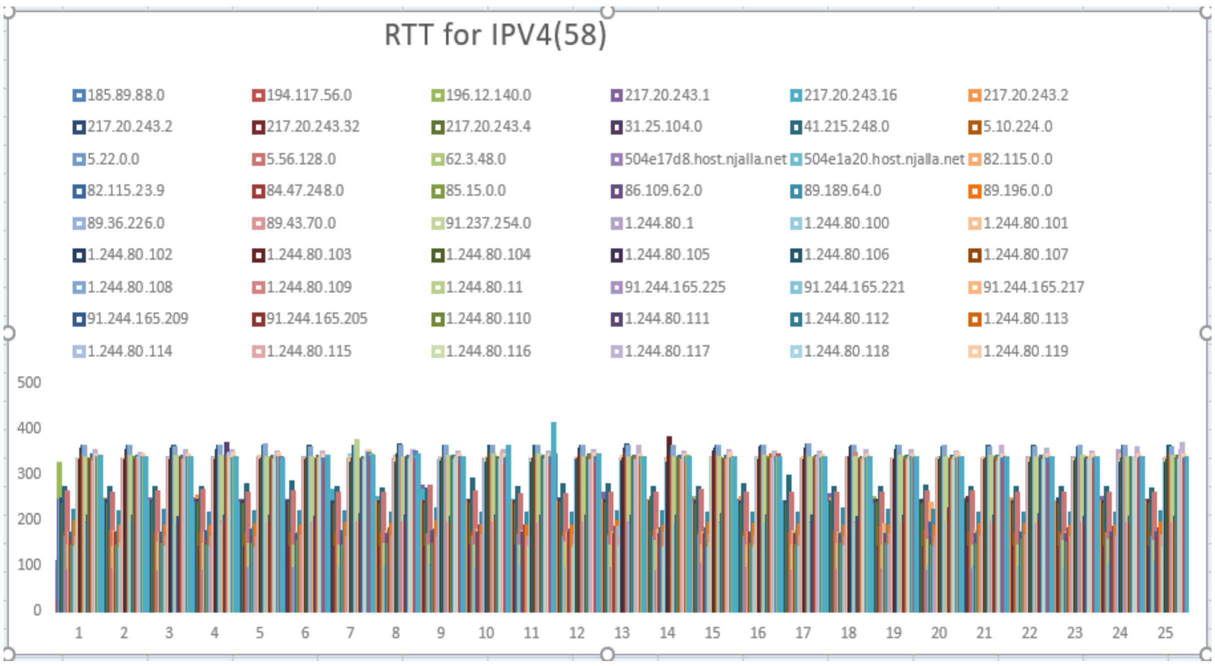
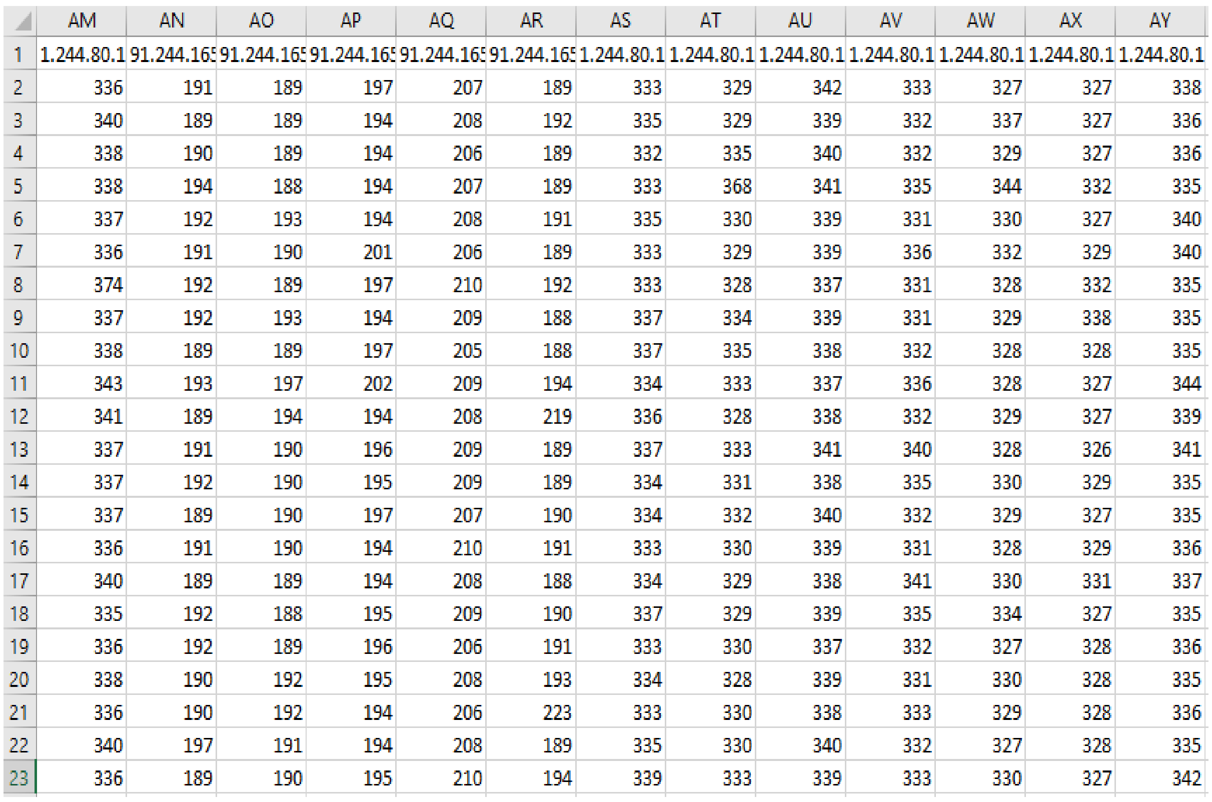


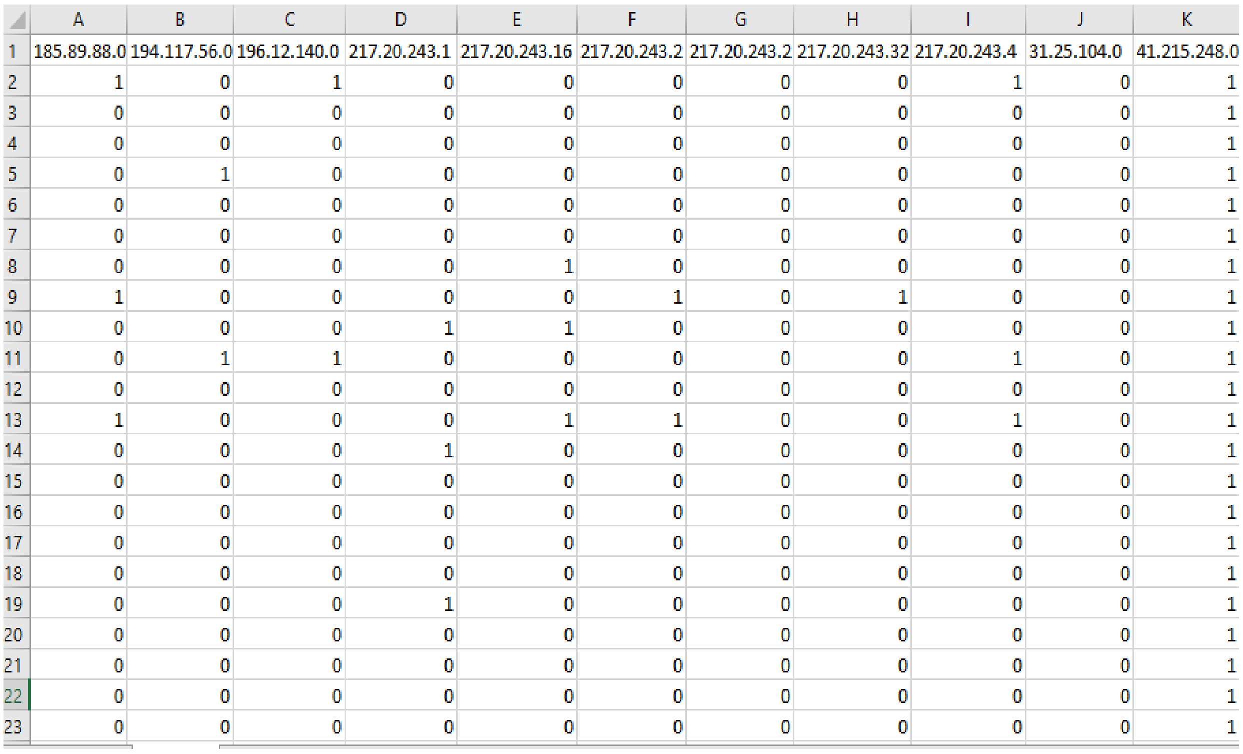
Figure (4.15) Network History File(NHF) of 58 data providers(Stable).



	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX	AY
1	1.244.80.1	91.244.165	91.244.165	91.244.165	91.244.165	91.244.165	1.244.80.1	1.244.80.1	1.244.80.1	1.244.80.1	1.244.80.1	1.244.80.1	1.244.80.1
2	336	191	189	197	207	189	333	329	342	333	327	327	338
3	340	189	189	194	208	192	335	329	339	332	337	327	336
4	338	190	189	194	206	189	332	335	340	332	329	327	336
5	338	194	188	194	207	189	333	368	341	335	344	332	335
6	337	192	193	194	208	191	335	330	339	331	330	327	340
7	336	191	190	201	206	189	333	329	339	336	332	329	340
8	374	192	189	197	210	192	333	328	337	331	328	332	335
9	337	192	193	194	209	188	337	334	339	331	329	338	335
10	338	189	189	197	205	188	337	335	338	332	328	328	335
11	343	193	197	202	209	194	334	333	337	336	328	327	344
12	341	189	194	194	208	219	336	328	338	332	329	327	339
13	337	191	190	196	209	189	337	333	341	340	328	326	341
14	337	192	190	195	209	189	334	331	338	335	330	329	335
15	337	189	190	197	207	190	334	332	340	332	329	327	335
16	336	191	190	194	210	191	333	330	339	331	328	329	336
17	340	189	189	194	208	188	334	329	338	341	330	331	337
18	335	192	188	195	209	190	337	329	339	335	334	327	335
19	336	192	189	196	206	191	333	330	337	332	327	328	336
20	338	190	192	195	208	193	334	328	339	331	330	328	335
21	336	190	192	194	206	223	333	330	338	333	329	328	336
22	340	197	191	194	208	189	335	330	340	332	327	328	335
23	336	189	190	195	210	194	339	333	339	333	330	327	342

Figure (4.16) Sample of Network History File)

After applying the mapping function algorithm () on NHF, the result as shown in Figure (4.17)



	A	B	C	D	E	F	G	H	I	J	K
1	185.89.88.0	194.117.56.0	196.12.140.0	217.20.243.1	217.20.243.16	217.20.243.2	217.20.243.2	217.20.243.32	217.20.243.4	31.25.104.0	41.215.248.0
2	1	0	1	0	0	0	0	0	1	0	1
3	0	0	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	0	0	0	1
5	0	1	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0	0	0	1
8	0	0	0	0	1	0	0	0	0	0	1
9	1	0	0	0	0	1	0	1	0	0	1
10	0	0	0	1	1	0	0	0	0	0	1
11	0	1	1	0	0	0	0	0	1	0	1
12	0	0	0	0	0	0	0	0	0	0	1
13	1	0	0	0	1	1	0	0	1	0	1
14	0	0	0	1	0	0	0	0	0	0	1
15	0	0	0	0	0	0	0	0	0	0	1
16	0	0	0	0	0	0	0	0	0	0	1
17	0	0	0	0	0	0	0	0	0	0	1
18	0	0	0	0	0	0	0	0	0	0	1
19	0	0	0	1	0	0	0	0	0	0	1
20	0	0	0	0	0	0	0	0	0	0	1
21	0	0	0	0	0	0	0	0	0	0	1
22	0	0	0	0	0	0	0	0	0	0	1
23	0	0	0	0	0	0	0	0	0	0	1

Figure (4.17) Sample of logical history file (LHF)

Then the Apriori algorithm (2.1) was applied and it was found that 32 IPV4 servers were working simultaneously. As shown in Table (4.29).

Table 4.29 IPV4 (ARS)

No.IP	Ipv4 servers
1	1.244.80.1
2	1.244.80.100
3	1.244.80.101
4	1.244.80.102
5	1.244.80.103
6	1.244.80.104
7	1.244.80.105
8	1.244.80.106
9	1.244.80.107
10	1.244.80.108
11	1.244.80.109
12	1.244.80.11
13	91.244.165.225
14	91.244.165.221
15	91.244.165.217
16	91.244.165.209
17	91.244.165.205
18	1.244.80.110
19	1.244.80.111
20	1.244.80.112
21	1.244.80.113
22	1.244.80.114
23	1.244.80.115
24	1.244.80.116
25	1.244.80.117
26	1.244.80.118
27	1.244.80.119
28	1.244.80.120
29	1.244.89.193
30	1.244.89.194
31	1.244.89.195
32	1.244.89.196

4.8 Scheduling on Three servers

The processing time on S1 is ranging between 1 at the minimum and 77 at the maximum overall provided jobs. The processing time on S2 is ranging between 4 at a minimum and 71 as the maximum overall provided jobs. These processing times are generated automatically according to server specifications . A shown in Table (4.30).

Table 4.30 Initial Processing Values

Jobs	Server1	Server2	Server3
J1	8	3	8
J2	3	4	7
J3	7	5	6
J4	2	2	9
J5	5	1	10
J6	1	6	9

It contains three servers. The output of the first server is an input of the second server, and the output of the second server is an input of the third server. Table (4.31) shows the entry and exit times of the processes in the three servers and also shows the total time for executing all the processes using the Johnson Algorithm.

Table 4.31 Waiting and Idle Time for Servers S1 and S2 (Johnson Algorithm)

Tasks sequence	***Server S1***		***Server S2***		***Server S3***		**Waiting time for S2**	**Idle time for Server S2**
	Time in	***Time out***	***Time in***	***Time out***	***Time in***	***Time out***		
J4	0	2	2	4	4	13	None	2
J5	2	7	7	8	13	23	0	3
J6	7	8	8	14	23	32	0	0
J2	8	11	14	18	32	39	3	0
J1	11	19	19	22	39	47	0	1
J3	19	26	26	31	47	53	0	4

Total elapsed time: 53

Idle time for three servers:

Idle time for S1 Appling using equation (2.3)

- S1= 53 – 26 = 27
- For S2: 10+(53-31)=32
- For S3: 4
- Average idle time = 10.5
- Waiting Time**:**

J4:0 , J5:13−8=5

J6:23−14=9 ,J1:39−22=17

J2:14−11+32−18=17

J3:47−31=16

Average waiting time =10.6

Chapter Five

Conclusions and Future Works

5.1 Conclusion

From the test results and the evaluations of the proposed system presented in the previous chapter, the most important characteristics that have been discovered through the implementation of the proposed methodology and the discussion of its results are as follows:

1. The proposed methodology has proven effective in selecting the best replica sites in the data network by using data mining methods to improve the selection of replicas.
2. The proposed methodology reduces the transmission time in the data network management system by replica selection method.
3. The Apriori algorithm has proven effective in extracting sites that run simultaneously; It was 26 IPV4 out of 41 in the first experiment, while in the second experiment, it was 32 out of 58 IPV4.
4. The proposed methodology proved effective in improving the selection of replicas, through the score equation, which included the three factors:(CPU speed, memory size and bandwidth). The Max of two Scores (23.6, and 31.9) are the best replica servers.
5. The proposed methodology proved effective in improving performance on two and three servers by using Johnson algorithm which is the best scheduling algorithm. The Johnson algorithm showed that the total time to execute 9 jobs is 373 hours and the idle time is equal to 4.5 hours. While SJF algorithm, the total time to execute 9 jobs is 391 hours and the idle time is equal to 8.5 hours. FCFS shows is the total time to execute 9 jobs is 474 hours and the idle time is equal to 27 hours.

5.2 The Future Work

In this section some future directions are suggested as an extension of the proposed system in this thesis:

1. Applying the proposed model to larger dataset, to be 2000-3000 IPV4.
2. Improving the replica selection process by involving the users in determining their preferences.
3. Proposing a new replication strategy to support replicas selection such as artificial intelligence algorithms.
4. Other factors can be used in the score equation, such as file size, response time, and cost
5. Use of modern scheduling algorithms.

References

[1] R. M. Almuttairi, R. Wankar, A. Negi, and C. R. Rao, “Smart replica selection for data grids using rough set approximations (RSDG),” *Proc. - 2010 Int. Conf. Comput. Intell. Commun. Networks, CICN 2010*, no. December, pp. 466–471, 2010, doi: 10.1109/CICN.2010.94.

[2] K. Omer and G. M. T. Abdalla, “Dynamic Algorithms Replication Using Grid Computing,” *2018 Int. Conf. Comput. Control. Electr. Electron. Eng. ICCCEEE 2018*, pp. 1–6, 2018, doi: 10.1109/ICCCEEE.2018.8515859.

[3] T. Hamrouni, S. Slimani, and F. Ben Charrada, “A survey of dynamic replication and replica selection strategies based on data mining techniques in data grids,” *Eng. Appl. Artif. Intell.*, vol. 48, no. February 2018, pp. 140–158, 2016, doi: 10.1016/j.engappai.2015.11.002.

[4] M. Beigrezaei, T. H. Abolfazl, and R. K. Hamidreza, “A new fuzzy based dynamic data replication algorithm in data grids,” *13th Iran. Conf. Fuzzy Syst. IFSC 2013*, no. January 2013, 2013, doi: 10.1109/IFSC.2013.6675676.

[5] R. M. Almuttairi, R. Wankar, A. Negi, C. R. Rao, A. Agarwal, and R. Buyya, “A two phased service oriented Broker for replica selection in data grids,” *Futur. Gener. Comput. Syst.*, vol. 29, no. 4, pp. 953–972, 2013, doi: 10.1016/j.future.2012.09.007.

[6] A. Jaradat, H. Alhussian, A. Patel, and S. M. Fati, “Multiple users replica selection in data grids for fair user satisfaction: A hybrid approach,” *Comput. Stand. Interfaces*, vol. 71, no. July, 2020, doi: 10.1016/j.csi.2020.103432.

[7] M. Beigrezaei, A. Toroghi Haghighat, and S. Leili Mirtaheri, “Minimizing data access latency in data grids by neighborhood-based data replication and job scheduling,” *Int. J. Commun. Syst.*, vol. 33, no. 15, pp. 1–33, 2020, doi: 10.1002/dac.4552.

[8] R. M. Almuttairi, M. S. Almhanna, S. Q. Muhamed, and M. Q. Mohammed, “Promote Replica Management based on Data Mining Techniques,” no. January, 2019, doi: 10.14419/ijet.v7i4.19.28006.

[9] R. M. Almuttairi, R. Wankar, A. Negi, R. R. Chillarige, and M. S. Almahna, “New replica selection technique for binding replica sites in data grids,” *EPC-IQ01 2010 - 2010 1st Int. Conf. Energy, Power Control*, vol. 6, no. 2, pp. 187–194, 2010, doi: 10.37917/ijeee.6.2.16.

[10] Z. Challal and T. Bouabana-Tebibel, “A priori replica placement strategy in data grid,” *2010 Int. Conf. Mach. Web Intell. ICMWI 2010 - Proc.*, pp. 402–406, 2010, doi: 10.1109/ICMWI.2010.5647925.

[11] N. Mansouri, M. M. Javidi, and B. Mohammad Hasani Zade, “Using data mining techniques to improve replica management in cloud environment,” *Soft Comput.*, vol. 24, no. 10, pp. 7335–7360, 2020, doi: 10.1007/s00500-019-04357-w.

[12] N. Mansouri and M. M. Javidi, “A review of data replication based on meta-heuristics approach in cloud computing and data grid,” *Soft Comput.*, vol. 24, no. 19, pp. 14503–14530, 2020, doi: 10.1007/s00500-020-04802-1.

[13] M. Jemmali, I. Agrebi, H. Alquhayz, and T. Ladhari, “Optimal algorithm for a two-machine flowshop scheduling problem with release dates and blocking constraints,” *J. Chinese Inst. Eng. Trans. Chinese Inst. Eng. A*, vol. 44, no. 5, pp. 440–447, 2021, doi: 10.1080/02533839.2021.1919560.

[14] Y. Song, H. Ni, and X. Zhu, “An enhanced replica selection approach based on distance constraint in ICN,” *Electron.*, vol. 10, no. 4, pp. 1–24, 2021, doi: 10.3390/electronics10040490.

[15] W. S. W. Awang, M. M. Deris, O. F. Rana, M. Zarina, and A. N. M. Rose, *Af fi nity Replica Selection in Distributed Systems*, no. May 2022. Springer International Publishing, 2019. doi: 10.1007/978-3-030-25636-4.

[16] A. Jaradat, “Replica Selection Algorithm in Data Grids: the Best-Fit Approach,” *Adv. Sci. Technol. Res. J.*, vol. 15, no. 4, 2021.

[17] M. Bsoul, A. E. Abdallah, K. Almakadmeh, and N. Tahat, “A Round-based Data Replication Strategy,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 1, pp. 31–39, 2016, doi: 10.1109/TPDS.2015.2388449.

[18] L. Zou, T. Ma, and J. Yang, “A replica selection algorithm in data grid,” *Information Technology Journal*, vol. 13, no. 1. pp. 32–41, 2014. doi: 10.3923/itj.2014.32.41.

[19] N. Sasikaladevi, “Minimum Makespan Task Scheduling Algorithm in Cloud Computing,” vol. 9, no. 11, pp. 61–70, 2016.

[20] C. Comp, “ADAPTATION OF JOHNSON SEQUENCING ALGORITHM FOR JOB SCHEDULING TO MINIMISE THE AVERAGE WAITING TIME IN”.

[21] M. Okwu and I. Emovon, “Application of Johnson’s algorithm in processing jobs through two-machine system,” *J. Mech. Energy Eng.*, vol. 4, no. 1, pp. 33–38, 2020, doi: 10.30464/jmee.2020.4.1.33.

[22] Y. Xiong, S. Huang, M. Wu, S. Member, J. She, and S. Member, “Transactions on Cloud Computing A Johnson ’ s-Rule-Based Genetic Algorithm for Two-Stage-Task Scheduling Problem in Data-Centers of

Cloud Computing," vol. 13, no. 9, 2017, doi: 10.1109/TCC.2017.2693187.

[23] S. A. Mansouri, E. Aktas, and U. Besikci, "Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption," *Eur. J. Oper. Res.*, vol. 248, no. 3, pp. 772–788, 2016, doi: 10.1016/j.ejor.2015.08.064.

[24] T. C. E. Cheng, C. C. Wu, J. C. Chen, W. H. Wu, and S. R. Cheng, "Two-machine flowshop scheduling with a truncated learning function to minimize the makespan," *Int. J. Prod. Econ.*, vol. 141, no. 1, pp. 79–86, 2013, doi: 10.1016/j.ijpe.2012.03.027.

[25] P. Grzybowski, C. Mazurek, and M. Wolski, "Data Management System for grid and portal services," vol. 2002.

[26] T. Hamrouni, S. Slimani, and F. Ben Charrada, "A critical survey of data grid Replication strategies based on data mining techniques," *Procedia Comput. Sci.*, vol. 51, no. 1, pp. 2779–2788, 2015, doi: 10.1016/j.procs.2015.05.434.

[27] R. M. Almuttairi, R. Wankar, A. Negi, and C. R. Rao, "Smart replica selection for data grids using rough set approximations (RSDG)," *Proc. - 2010 Int. Conf. Comput. Intell. Commun. Networks, CICN 2010*, pp. 466–471, 2010, doi: 10.1109/CICN.2010.94.

[28] B. Segal, L. Robertson, F. Gagliardi, and F. Carminati, "Grid computing: The European data grid project," *IEEE Nucl. Sci. Symp. Med. Imaging Conf.*, vol. 1, no. February, 2000, doi: 10.1109/nssmic.2000.948988.

[29] L. Banica and C. Åžtefan, "From Grid Computing To Cloud Infrastructures," *Int. J. Comput. Technol.*, vol. 12, no. 1, pp. 3187–3194, 2013, doi: 10.24297/ijct.v12i1.3372.

[30] R. M. Almuttairi, "New Replica Selection Technique for Binding Cheapest Replica Sites in Data Grids," Jul. 2012, pp. 295–312. doi: 10.5121/csit.2012.2329.

[31] G. R. Vijay and A. R. M. Reddy, "Data Replication System in Cloud based on Data Mining Techniques," vol. 2, no. 11, 2013.

[32] N. Mostafa, W. H. F. Aly, S. Alabed, and Z. Al-Arnaout, "Intelligent Replica Selection in Edge and IoT Environments Using Artificial Neural Networks," *Electron.*, vol. 11, no. 16, 2022, doi: 10.3390/electronics11162531.

[33] T. Hamrouni, S. Slimani, and F. Ben Charrada, "A survey of dynamic replication and replica selection strategies based on data mining techniques

in data grids," *Eng. Appl. Artif. Intell.*, vol. 48, no. February, pp. 140–158, 2016, doi: 10.1016/j.engappai.2015.11.002.

[34] W. Ben Abid, M. Ben Ahmed Mhiri, E. Bouazizi, and F. Gargouri, "Ontological data replication in a distributed real-time database system," *Front. Artif. Intell. Appl.*, vol. 337, no. November, pp. 567–580, 2021, doi: 10.3233/FAIA210054.

[35] Y. Zhao and Y. Hu, "GRESS - a Grid Replica Selection Service," *16th ISCA Int. Conf. Parallel Distrib. Comput. Syst. 2003, PDCS 2003*, pp. 423–429, 2003.

[36] H. Hamidpour, Y. Liu, and A. Raza, "Data Mining Techniques for Web Mining: A Survey," no. October, 2022, doi: 10.47852/bonviewXXXXXXXXX.

[37] S. Agarwal, *Data mining: Data mining concepts and techniques*. 2014. doi: 10.1109/ICMIRA.2013.45.

[38] A. S. Osman, "Data mining techniques: Review," *Int. J. Data Sci. Res.*, vol. 2, no. 1, pp. 1–4, 2019.

[39] S. Bagga and A. Sharma, "Big Data and Its Challenges: A Review," *Proc. - 4th Int. Conf. Comput. Sci. ICCS 2018*, no. November, pp. 183–187, 2019, doi: 10.1109/ICCS.2018.00037.

[40] L. Chen, H. Hu, X. He, and M. Lyu, "The Development Path and Data Mining Mode of Rural Tourism under the Background of Big Data," *Wirel. Commun. Mob. Comput.*, vol. 2022, 2022, doi: 10.1155/2022/3031169.

[41] A. Sulistiyawati and E. Supriyanto, "Implementasi Algoritma K-means Clustring dalam Penetuan Siswa Kelas Unggulan," *J. Tekno Kompak*, vol. 15, no. 2, p. 25, 2021, doi: 10.33365/jtk.v15i2.1162.

[42] M. D. S. Anarase, "SURVEY OF CLUSTRING ALGORITHMS IN WSN Ms.," *GIS Sci. J.*, no. September, 2020.

[43] T. Berteloot, R. Khoury, and A. Durand, "Cambrian Explosion Algorithm for Multi-Objective Association Rules Mining," 2022, [Online]. Available: http://arxiv.org/abs/2211.12767

[44] C. Jiang, C. Jiang, D. Chen, and F. Hu, "Densely Connected Neural Networks for Nonlinear Regression," *Entropy*, vol. 24, no. 7, 2022, doi: 10.3390/e24070876.

[45] R. K. Grace, "GA Based Replica Selection in Data Grid," no. February 2015, 2014.

[46] R. M. Almuttairi, R. Wankar, A. Negi, R. R. Chillarige, and M. S. Almahna, “New replica selection technique for binding replica sites in data grids,” *EPC-IQ01 2010 - 2010 1st Int. Conf. Energy, Power Control*, no. January, pp. 187–194, 2010, doi: 10.37917/ijeee.6.2.16.

[47] S. E. E. Profile and S. E. E. Profile, “Adaptive Replica Selection in Mobile Edge Environments Adaptive Replica Selection in Mobile Edge Environments,” no. May, 2022, doi: 10.1007/978-3-030-94822-1.

[48] A. A. Eid, “REPLICATION STRATEGIES BASED ON MARKOV CHAIN MONTE CARLO AND REPLICATION STRATEGIES BASED ON MARKOV CHAIN,” no. March, 2022.

[49] H. Chan, “Response Time Optimization for Replica Selection Service in Data Grids,” no. June 2008, 2015, doi: 10.3844/jcssp.2008.487.493.

[50] M. Radi, “Fussy logic replica selection algorithm for cloud storage system ׀ Fussy logic replica selection algorithm for cloud storage system,” no. February 2016, 2017.

[51] T. A. Kumbhare Santosh V Chobe, “An Overview of Association Rule Mining Algorithms,” *Int. J. Comput. Sci. Inf. Technol. Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 1, pp. 927–930, 2014, [Online]. Available: www.ijcsit.com

[52] A. Multim and L. Manouba, “Replication in Data Grids : Metrics and Strategies Mémoire de Synthèse Habilitation Universitaire en Informatique,” 2017.

[53] Q. Rasool, “Replica Placement in Multi-tier Data Grid Replica Placement in Multi-tier Data Grid,” no. December 2009, 2014, doi: 10.1109/DASC.2009.60.

[54] S. Lim, B. Sharma, G. Nam, E. K. Kim, and C. R. Das, “MDCSim : A Multi-tier Data Center Simulation Platform,” no. May, 2014, doi: 10.1109/CLUSTR.2009.5289159.

[55] R. M. Almuttairi, M. S. Almhanna, M. Q. Mohammed, and S. Q Muhamed, “Promote Replica Management based on Data Mining Techniques,” *Int. J. Eng. Technol.*, vol. 7, no. 4.19, p. 838, 2018, doi: 10.14419/ijet.v7i4.19.28006.

[56] A. Ciuffoletti, A. Congiusta, G. Jankowski, M. Jankowski, O. Krajiček, and N. Meyer, “Grid infrastructure architecture: A modular approach from CoreGRID,” *Lect. Notes Bus. Inf. Process.*, vol. 8, no. 4, pp. 72–84, 2008, doi: 10.1007/978-3-540-68262-2_6.

[57] Y. Cunjiang, Z. Huaxun, and Z. Lei, “Architecture Design For Smart Grid,” *Energy Procedia*, vol. 17, pp. 1524–1528, 2012, doi: 10.1016/j.egypro.2012.02.276.

[58] M. Hasnain, M. H. Malik, A. Raoof, S. Ullah, and A. Dad, “Architecture for Grid Computing,” vol. 14, no. 11, pp. 84–90, 2016.

[59] S. Pardeshi, C. Patil, and S. Dhumale, “Grid Computing Architecture and Benefits,” *Ijsrp.Org*, vol. 3, no. 8, pp. 3–6, 2013.

[60] M. K. D. R. Dhivya and M. C. Sunitha, “Article on Grid Computing Architecture and Benefits,” pp. 1070–1074, 2015.

[61] H. Dafaalla *et al.*, “Deep Learning Model for Selecting Suitable Requirements Elicitation Techniques,” *Appl. Sci.*, vol. 12, no. 18, 2022, doi: 10.3390/app12189060.

[62] R. Rathan, “International Journal of Advanced Research in Association Rule- Spatial Data Mining Approach for Exploration of Endometrial Cancer Data,” no. November 2013, 2016.

[63] B. M. Patel, V. H. Bhemwala, and D. A. R. Patel, “Analytical Study of Association Rule Mining Methods in Data Mining,” *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, no. June, pp. 818–831, 2018, doi: 10.32628/cseit1833244.

[64] H. Pang and L. Zhou, “Application and Analysis of Hypergraph Association Rule Redundancy Algorithm in Data Mining,” *Mob. Inf. Syst.*, vol. 2022, 2022, doi: 10.1155/2022/1193586.

[65] N. Mikail and A. Cig, “IMPLEMENTATION OF APRIORI ALGORITHM IN SURVEY ANALYSIS,” no. August, 2021.

[66] M. Hayashi and S. Song, “Two-Server Oblivious Transfer for Quantum Messages,” pp. 1–13, 2022, [Online]. Available: http://arxiv.org/abs/2211.03308

[67] R. Benmansour and A. Sifaleras, “Scheduling in Parallel Machines with Two Servers: The Restrictive Case,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12559 LNCS, no. March, pp. 71–82, 2021, doi: 10.1007/978-3-030-69625-2_6.

[68] T. Kokilavani and D. I. George Amalarethinam, “Reduced makespan task scheduling algorithm for Grid Computing,” *Int. J. Control Theory Appl.*, vol. 9, no. 27, pp. 71–76, 2016.

[69] W. Viyakulamary and T. Kokilavani, “A study on Task Scheduling

Algorithms in Grids," *Int. J. Eng. Comput. Sci.*, no. August 2015, 2015, doi: 10.18535/ijecs/v4i8.56.

[70] P. Kokkinos, K. Christodoulopoulos, A. Kretsis, and E. Varvarigos, "Data consolidation: A task scheduling and data migration technique for grid networks," *Proc. CCGRID 2008 - 8th IEEE Int. Symp. Clust. Comput. Grid*, no. June, pp. 722–727, 2008, doi: 10.1109/CCGRID.2008.18.

[71] L. Zheng, "Optimization of Agricultural Machinery Task Scheduling Algorithm Based on Multiobjective Optimization," *J. Sensors*, vol. 2022, 2022, doi: 10.1155/2022/5800332.

[72] A. S. S. Navaz, C. Prabhadevi, and V. Sangeetha, "Data Grid Concepts for Data Security in Distributed Computing," vol. 61, no. 13, pp. 6–11, 2013.

[73] S. Behzad, R. Fotohi, and M. Effatparvar, "Queue based Job Scheduling algorithm for Cloud computing," no. October, 2013.

[74] J. V. Gautam, H. B. Prajapati, V. K. Dabhi, and S. Chaudhary, "Empirical study of job scheduling algorithms in hadoop mapreduce," *Cybern. Inf. Technol.*, vol. 17, no. 1, pp. 146–163, 2017, doi: 10.1515/cait-2017-0012.

[75] S. M. Johnson, "Optimal two- and three-stage production schedules with setup times included," *Nav. Res. Logist. Q.*, vol. 1, no. 1, pp. 61–68, 1954, doi: 10.1002/nav.3800010110.

[76] T. D. Putra, "Analysis of Preemptive Shortest Job First (SJF) Algorithm in CPU Scheduling," *Ijarcce*, vol. 9, no. 4, pp. 41–45, 2020, doi: 10.17148/ijarcce.2020.9408.

[77] C. Yaashuwanth *et al.*, "Performance Comparison of Priority Rule Scheduling Algorithms Using Different Inter Arrival Time Jobs in Grid Environment," *Int. J. Grid Distrib. Comput.*, vol. 6, no. 4, pp. 157–168, 2012.

[78] N. Y. Hidayah, M. Syafrizal, and M. Darmawan, "Analysis of textile dye production scheduling using FCFS, CDS and Heuristic Pour methods," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 847, no. 1, 2020, doi: 10.1088/1757-899X/847/1/012006.

[79] H. Liu, M. Bowman, R. Adams, J. Hurliman, and D. Lake, "Scaling virtual worlds: Simulation requirements and challenges," *Proc. - Winter Simul. Conf.*, no. January, pp. 778–790, 2010, doi: 10.1109/WSC.2010.5679112.

[80] A. Shakarami, M. Ghobaei-Arani, A. Shahidinejad, M. Masdari, and H. Shakarami, *Data replication schemes in cloud computing: a survey*, vol. 24, no. 3. Springer US, 2021. doi: 10.1007/s10586-021-03283-7.

[81] M. Hans and V. Jogi, “Peak load scheduling in smart grid using cloud computing,” *Bull. Electr. Eng. Informatics*, vol. 8, no. 4, 2019, doi: 10.11591/eei.v8i4.1687.

[82] F. Della Croce, A. Grosso, and F. Salassa, “Minimizing total completion time in the two-machine no-idle no-wait flow shop problem,” *J. Heuristics*, vol. 27, no. 1–2, pp. 159–173, 2021, doi: 10.1007/s10732-019-09430-z.

[83] “IPv4 geolocation - Dataset - DataHub - Frictionless Data.”

المستخلص

يسمح الهيكل التنظيمي لحوسبة الشبكة لمنشآت الشبكة بإدارة ملفات البيانات ونسخها في جميع أنحاء العالم. تم تطوير تقنية Data Grid لمشاركة البيانات عبر العديد من المواقع في مواقع جغرافية مختلفة لتحسين الوصول إلى البيانات وزيادة سرعة نقل البيانات. عندما تحتفظ مواقع مختلفة بنسخة من الملفات، تتحقق فوائد كبيرة عند اختيار أفضل مواقع النسخ المتماثلة التي تتعاون لتسريع عملية نقل الملفات.

في هذه الأطروحة، يتم استخدام تقنية استخراج البيانات، وهي طريقة قواعد الارتباط باستخدام خوارزمية Apriori لاستكشاف الخصائص المشتركة للمواقع لتحديد مواقع النسخ المتماثلة غير المزدحمة. تؤثر بعض العوامل مثل سعة الذاكرة وأداء وحدة المعالجة المركزية وعرض النطاق الترددي المتوفر على أداء الخادم. بناءً على هذه المعلمات، يتم اختيار أفضل الخوادم لتقليل الوقت اللازم لنقل جميع البيانات الهامة وتحسين أداء النظام.

جدولة المهام هي واحدة من أكثر العقبات صعوبة في الحوسبة الشبكية. عند طلب العديد من المهام، يجب أن تنتظر هذه المهام قبل تعيينها للخوادم، مما قد يتسبب في زيادة قائمة الانتظار وزيادة وقت الانتظار. بعد اختيار أفضل خادمين بناءً على أعلى الدرجات، يتم تحسين جدولة الوظائف باستخدام إحدى خوارزميات الجدولة وهي خوارزمية Johnson Scheduling لتقليل إجمالي إكمال العمل ووقت الخمول وتحسين أداء النظام.

قاعدة البيانات المستخدمة في هذا العمل هي (1500) IPv4 مستقرة وغير مستقرة. أظهرت نتائج العمل أن الطريقة المقترحة في الجزء الأول من المشكلة كانت (26 IPv4) من أصل(41) IPv4مستقرة. تم حساب النتائج لـ (26 IPv4) وتم اختيار الخادمين اللذين يمثلان أقصى درجتين (23.6 ، 31.9) هما أفضل خوادم متماثلة. أظهرت نتائج الجزء الثاني من المشكلة أن خوارزمية جونسون وجدت أن إجمالي وقت الانتهاء لجميع المهام بين خادمين هو 373 ساعة وإجمالي وقت الخمول 4.5 ساعة. كما هو موضح في نتائج الجزء الثاني من المشكلة، فإن خوارزمية جونسون هي أفضل خوارزمية جدولة من ناحية الأداء مقارنة بخوارزميات الجدولة الأخرىFCFS و SJFS.

جمهورية العراق

وزارة التعليم العالي والبحث العلمي

جامعة بابل

كلية تكنلوجيا المعلومات

قسم البرمجيات

تحسين الأداء لاختيار أفضل النسخ المتماثلة باستخدام قاعدة الاقتران وجدولة العمل في شبكة البيانات

رسالة مقدمة الى

مجلس كلية تكنلوجيا المعلومات – جامعة بابل كجزء من متطلبات نيل درجة الماجستير في تكنلوجيا المعلومات / قسم البرمجيات

من قبل

رؤى ستار جبار محمد

بأشراف

أ.م.د مهدي صالح نعمة موسى

1444هـ 2023م