

STATUS OF THESIS

Title of thesis

GREEDY SINGLE USER AND FAIR MULTIPLE USERS REPLICA
SELECTION DECISION IN DATA GRID

I AYMAN KAMEL SALIM JARADAT,

hereby allow my thesis to be placed at the Information Resource Center (IRC) of Universiti Teknologi PETRONAS (UTP) with the following conditions:

1. The thesis becomes the property of UTP
2. The IRC of UTP may make copies of the thesis for academic purposes only.
3. This thesis is classified as


Confidential

Non-confidential

If this thesis is confidential, please state the reason:

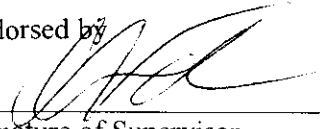
The contents of the thesis will remain confidential for _____ years.

Remarks on disclosure:


Signature of Author

Permanent address: Mafrq Po Box 900, Jordan.

Date : 17.5.2013

Endorsed by 
Signature of Supervisor

Dr. Mohamed Nordin B Zakaria

Date : 17.5.2013

UNIVERSITI TEKNOLOGI PETRONAS

GREEDY SINGLE USER AND FAIR MULTIPLE USERS REPLICA SELECTION

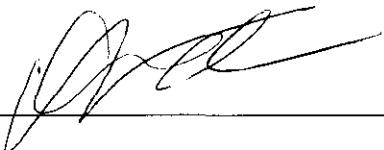
DECISION IN DATA GRIDS

by

AYMAN JARADAT

The undersigned certify that they have read, and recommend to the Postgraduate Studies Programme for acceptance this thesis for the fulfillment of the requirements for the degree stated.

Signature:



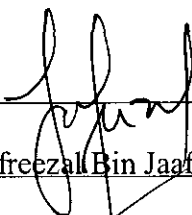
Main Supervisor:

Dr. Mohamed Nordin B Zakaria

Signature:

Co-Supervisor:

Signature:



Head of Department:

Dr. Jafreezal Bin Jaafar

Dr. Jafreezal Bin Jaafar
Head
Department of Computer & Information Sciences,
Universiti Teknologi PETRONAS
31150 Tronoh
Jalor, Perak, Malaysia MALAYSIA

Date:

GREEDY` SINGLE USER AND FAIR MULTIPLE USERS REPLICA SELECTION
DECISION IN DATA GRID

by

AYMANJARADAT

A Thesis

Submitted to the Postgraduate Studies Programme

as a Requirement for the Degree of

DOCTOR OF PHILOSOPHY

INFORMATION TECHNOLOGY

UNIVERSITI TEKNOLOGI PETRONAS

BANDAR SERI ISKANDAR,

PERAK

MAY 2013

DECLARATION OF THESIS


Title of thesis

GREEDY SINGLE USER AND FAIR MULTIPLE USERS REPLICA
SELECTION DECISION IN DATA GRID

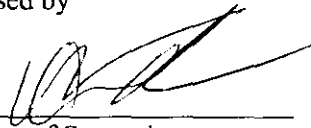
I AYMAN KAMEL SALIM JARADAT,

hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTP or other institutions.

Witnessed by



Signature of Author



Signature of Supervisor

Permanent address: Mafraq Po Box 900, Jordan.

Dr. Mohamed Nordin B Zakaria

Date: 17-5-2013

Date: 17-5-2013

DEDICATION

This Thesis is dedicated

To

My Beloved Parents

ACKNOWLEDGEMENT

All praise is to be to Allah, the Cherisher and Sustainer of the world. My deepest hearted appreciation goes to my supervisors Dr. Mohamed Nordin B Zakaria and Dr. Anang Hudaya Bin Muhamad Amin who kindly and gently have taken me throughout the rigors of this research work bringing it to a logical conclusion. My appreciation also goes to the Universiti Teknologi PETRONAS, and all the staff and the employees for whom I could not find the perfect words to acknowledge my profound thank to all what have they done to me. I would be honored to say: You will remain dear to my heart and I will forever remain grateful.

To all the respondents and anonymous commentators, I say a VERY BIG THANK YOU for providing your valuable time and useful suggestions for improving this thesis. You have added a priceless value to my life in no small measure. I am immensely indebted to my dearest parents for all the support rendered selflessly to me. To my wife and Son, I must commend your high level of understanding and tolerance during this study period.

I shall not fail to pay a very glowing tribute to all my very dear friends, specifically Dr 'Ahmed', and all colleagues who have shown a deep concern and assistance and have proved that they are truly real friends and one is honored to have them. I do pray Allah that this friendship is kept and sustained as long as we live. The memories of the time we shared together shall ever remain in my mind.

ABSTRACT

Replication in data grids increases data availability, accessibility and reliability. Replicas of datasets are usually distributed to different sites, and the choice of any replica locations has a significant impact. Replica selection algorithms decide the best replica places based on some criteria. To this end, a family of efficient replica selection systems has been proposed (RsDGrid). The problem presented in this thesis is how to select the best replica location that achieve less time, higher QoS, consistency with users' preferences and almost equal users' satisfactions. RsDGrid consists of three systems: A-system, D-system, and M-system. Each of them has its own scope and specifications. RsDGrid switches among these systems according to the decision maker. The first two systems improve the existing single user replica selection systems by incorporating new features, while M-system presents a new direction focusing on simultaneous multiple users' selection.

A-system integrates site availability QoS parameter in the selection process as a very important parameter, because the absence of this parameter could leads to excessive delays. Simulations have been performed in the OptorSim grid simulator and the results have proved that A-system outperformed the other systems in the literature. D-system handles several heterogeneous QoS parameters and tries to find the replica location having the best ratings for all QoS parameters or the one that better matches the user's preferences. Simulation results have proved its efficiency.

M-system is an upgrade of D-system and the others by facilitating multiple users' selection to achieve almost equal satisfactions of the grid users. In all previous systems, users' requests are fulfilled one by one; this could satisfy the first users in the scheduling queue in comparison to those who are at the back. Considering all the users' requests simultaneously to gain fair user's satisfaction can obtain better results; however, it is a challenging task because it requires a huge search. Therefore, Genetic Algorithm has been hybridized with D-system to overcome the huge searches. Simulation results have proved that M-system solves this complex problem better than the previous works.

ABSTRAK

Replikasi dalam grid data dapat meningkatkan ketersediaan, kemudahan dan kebolehpercayaan data. Replika dataset-dataset biasanya diagihkan ke tapak-tapak yang berlainan, dan pemilihan mana-mana lokasi replika mempunyai impak yang ketara. Algoritma pemilihan replika akan menentukan lokasi replika terbaik berdasarkan beberapa kriteria. Tesis ini mencadangkan satu sistem pemilihan replika yang berkesan (RsDGrid). Masalah yang dikemukakan di dalam tesis ini adalah bagaimana untuk memilih lokasi replika terbaik yang mengambil masa yang kurang, lebih tinggi QoS, konsisten dengan kepuasan 'pilihan pengguna dan pengguna hampir sama'. RsDGrid terdiri daripada tiga sistem: Sistem-A, Sistem-D, dan Sistem-M. Setiap dari mereka mempunyai skop dan spesifikasi yang tersendiri. RsDGrid beralih diantara system-system ini mengikut keputusan pengguna. Dua sistem pertama memperbaiki sistem pemilihan replika tunggal yang sedia ada dengan menggabungkan ciri-ciri baru, manakala Sistem-M memberi hala tuju baru dengan menumpu kepada pemilihan serentak pelbagai pengguna. Sistem-A mengintegrasikan ketersediaan tapak parameter QoS dalam proses pemilihan sebagai parameter yang sangat penting, kerana ketiadaan parameter ini boleh membawa kepada kelewatan yang terlalu. Simulasi telah membuktikan bahawa Sistem-A mengatasi system-system lain yang ada. Sistem-D mengendalikan beberapa parameter QoS yang berlainan dan cuba untuk mencari lokasi replika yang mempunyai penilaian yang terbaik untuk semua parameter QoS atau satu lokasi yang lebih sepadan dengan pilihan pengguna. Keputusan simulasi telah membuktikan kecekapannya. Sistem-M memudahkan pelbagai pilihan pengguna untuk mencapai kepuasan yang hampir sama oleh pengguna grid. Dalam semua sistem sebelumnya, permintaan pengguna dipenuhi satu demi satu, ini cuma dapat memuaskan pengguna pertama dalam barisan penjadualan. Memandangkan semua permintaan pengguna serentak dan untuk mendapatkan kepuasan pengguna yang saksama. Oleh itu Algoritma Genetik telah digabung dengan Sistem-D untuk mengatasi carian besar. Sistem-M menyelesaikan masalah yang kompleks ini dengan lebih baik daripada hasil yang sebelumnya.

In compliance with the terms of the Copyright Act 1987 and the IP Policy of the university, the copyright of this thesis has been reassigned by the author to the legal entity of the university,

Institute of Technology PETRONAS Sdn Bhd.

Due acknowledgement shall always be made of the use of any material contained in, or derived from, this thesis.

© Ayman Kamel Salim Jaradat, 2013

Institute of Technology PETRONAS Sdn Bhd

All rights reserved.

TABLE OF CONTENTS

UNIVERSITI TEKNOLOGI PETRONAS	iv
GREEDY SINGLE USER AND FAIR MULTIPLE USERS REPLICA SELECTION DECISION IN DATA GRIDS	iv
DECLARATION OF THESIS	iv
DEDICATION.....	v
ACKNOWLEDGEMENT	vi
ABSTRACT	vii
ABSTRAK.....	viii
TABLE OF CONTENTS	x
LIST OF TABLES.....	xiv
LIST OF FIGURES	xvi
LIST OF ABBREVIATIONS	xviii
CHAPTER 1	1
INTRODUCTION	1
1.1 Background of Grid Computing & Data Grid Motivations	1
1.2 Replication in Data Grids.....	4
1.3 Motivation for Replica Selection in Data Grids.....	6
1.4 Problem Background.....	7
1.5 Problem Statement	10
1.6 Thesis Goal	11
1.7 Thesis Objectives	11
1.8 The Scope of the Study	12
1.9 Research Framework.....	13
1.10 The Importance of the Study.....	15
1.11 Thesis Layout.....	15
CHAPTER 2	17
AN OVERVIEW AND LITERATURE REVIEW	17
2.1 Overview	17
2.2 Data Grid Architecture.....	17
2.3 Data Management in Data Grids.....	20
2.3.1 Metadata Service	22
2.3.3 Related Data-Intensive Application Domains.....	24
2.3.4 Globus Toolkit.....	25
2.4 Replica Selection Strategies.....	27
2.4.1 Heuristically & Statically Configured Replica Selection Algorithms	27

2.4.2 Dynamic Replica Selection Algorithms.....	28
2.5 An Overview of Simulation Tools and Evaluation.....	35
2.7 Genetic Algorithms.....	41
2.8 Multi-objective Optimization Model.....	46
2.8.1 Scalarization in Multi-objective Optimization.....	48
2.9 Proposed Solution.....	48
2.10 Summary.....	50
CHAPTER 3.....	51
METHODOLOGY.....	51
3.1 Overview.....	51
3.2 The Proposed Systems.....	51
3.2.1 A-system.....	52
3.2.2 D-system.....	54
3.2.3 M-system.....	56
3.3 Simulation.....	57
3.3.1 OptorSim Grid Simulator.....	57
3.3.2 Simulator Modifications.....	59
3.3.3 Dataset and Configurations.....	62
3.4 Summary.....	64
CHAPTER 4.....	65
THE IMPACT OF SITE AVAILABILITY IN REPLICA SELECTION.....	65
4.1 Overview.....	65
4.2 Theoretical Background.....	65
4.3 A-system Requirements.....	67
4.4 A-system System Design and Features.....	68
4.4.1 Rating Time.....	70
4.4.2 Rating Site Availability.....	73
4.4.3 Estimating the Best Site.....	75
4.5 Simulation Setup.....	79
4.6 Performance Metrics & Cost.....	81
4.7 Results and Discussion.....	82
4.8 Summary.....	88
CHAPTER 5.....	89
MULTI QoS PARAMETERS REPLICA SELECTION.....	89
5.1 Overview.....	89
5.2 Background.....	89
5.3 D-System Requirements.....	90
5.4 D-system Design.....	91

5.5 D-system Detailed Design	93
5.6 Security Parameter	93
5.6.1 Self-Protection Capability Calculation.....	94
5.6.2 Reputation Calculation.....	94
5.6.3 Reliability Calculation.....	96
5.6.4 Security Rating & Trust Factor Calculation.....	96
5.7 Replica Decision Maker (D-System)	97
5.8 D-System First Approach (D-SystemAp1)	99
5.8.1 Case Study.....	102
5.8.2 Simulation Setup	104
5.8.3 Performance Metrics	104
5.8.4 Results and Discussion.....	105
5.8.5 Summary of D-SystemAp1	109
5.9 D-System Second Approach (D-SystemAp2).....	109
5.9.1 Stock Market Model.....	111
5.9.2 D-SystemAp2 Detailed Procedure	112
5.9.3 Performance Metrics	115
5.9.4 Results and Discussion.....	117
5.9.5 Summary of D-SystemAp2	122
CHAPTER 6.....	123
MULTI USERS SELECTION & FAIR USERS' SATISFACTION.....	123
6.1 Overview	123
6.2 Theoretical Background.....	123
6.3 Problem Formulation	125
6.4 Permutation & Genetic Algorithm	127
6.5 Mathematical Modeling	128
6.5.1 Parameters	128
6.5.2 Decision variables	128
6.5.3 Objective functions	129
6.5.4 Scalarization.....	130
6.6 M-system Requirements and Design	131
6.6.1 Rating Cost.....	132
6.6.2 Genetic Algorithm.....	132
6.6.3 Selection of Operator	134

6.6.4 Mutation Operator	134
6.6.5 The Repair Operator.....	134
6.6.6 Hybridizing D-system with GA	136
6.6.7 System Detailed Design	136
6.7 Performance Metrics and Evaluation.....	138
6.7.1 Total User’s Satisfaction	138
6.7.2 Fair Users’ Satisfaction	139
6.7.3 Evaluation.....	139
6.8 Results and Discussion.....	140
6.8.1 Case (1) Fair Users’ Satisfactions & the Total UPQ.....	141
6.8.2 Case (2) Scalability Test and Best Replica	148
6.9 Summary	150
CHAPTER 7.....	151
CONCLUSIONS AND FUTURE WORK.....	151
7.1 Overview	151
7.2 Research Summary	151
7.3 Thesis Contributions	154
7.4 Limitation of the Thesis	155
7.5 Future Works.....	155
LIST OF PUBLICATIONS	157
REFERENCES.....	158

LIST OF TABLES

Table 1.1: Example of Sites and Their Parameters Rates' Values	9
Table 2.1: A Summary of the Algorithms Used for Replica Selection	28
Table 2.2: Features of the Simulators	36
Table 2.3: Listing of Functionalities and Features for Grid Simulators	38
Table 4.1: 10 GB and 100 GB Replicas with Different Metric Values for: Common Storage Speed and Bandwidth, Queue Waiting Time and Remaining Time.....	73
Table 4.2: Possible Paired Values of A & T & Various Values for β	78
Table 4.3: Example of Applying the Proposed System	79
Table 4.4: Workload and System Parameter Values	81
Table 4.5: Average Jobs' Time for 500 Jobs with Different Values of α & β	83
Table 4.6: Average Jobs' Time for 100 Jobs when Availability is Always 100%.	84
Table 4.7 (a): Average Jobs' Time in Seconds for 100 Jobs	84
Table 4.7 (b): Average Jobs' Time in Seconds for 500 Jobs.....	85
Table 4.7 (c): Average Jobs' time in Seconds for 1000 Jobs	85
Table 4.8: Paired Samples Statistics between A-system & OsBi.....	85
Table 5.1: Security Factors Considered for Self-Protection Capability	95
Table 5.2: Security Attributes Considered for Reputation	95
Table 5.3 & 5.4: 10 GB Replicas with Different Parameters & Sub Parameters'	103
Table 5.5: Workload & System Parameter Values	104
Table 5.6: TAS Descriptive Statistics for M-system, Random and OsBi.....	108
Table 5.7: The Results of Multivariate Tests based on TAS	108
Table 5.8: The Results of Tests Between-Systems Effects based on TAS.....	108
Table 5.9: TAS Pairwise Comparisons.....	109

Table 5.10: Time Metrics Calculation	116
Table 5.11: Average Values of UPQ after 150 Requests for Values of K from 2-5 .	121
Table 6.1: Pairing Requests to Sites in an Arbitrary Order	126
Table 6.2: Pairing Requests to Sites in a Managed Order	126
Table 6.3: Different Pairing Requests to Sites.....	127
Table 6.4: Experiment Setup	141
Table 6.5: Sites with their QoS Parameters' Values.....	141
Table 6.6: Users with their Preferred QoS Parameters' Values	142
Table 6.7(a): FUSs & TUPQs of the 10 Experiments Using M & D systems	143
Table 6.7(b): FUSs & TUPQs of the 10 Experiments Using M-system & AHP	143
Table 6.8: 10 Experiments Using M-system, AHP & D-system.....	144
Table 6.9: TUPQ Descriptive Statistics for M-system, D-system & AHP.....	145
Table 6.10: The Results of Multivariate Tests based on TUPQ.....	145
Table 6.11: The Results of Tests Between-Systems Effects based on TUPQ.....	146
Table 6.12: TUPQ Pairwise Comparisons.....	146
Table 6.13: FUS Descriptive Statistics for M-system, D-system & AHP.....	146
Table 6.14: The Results of Multivariate Tests based on FUS	147
Table 6.15: The results of Tests of Between- Systems Effects based on FUS.....	147
Table 6.16: FUS Pairwise Comparisons.....	148
Table 6.17(a): The Performance of 9 Experiments Using M & D systems.....	149
Table 6.17 (b): The Performance of 9 Experiments Using M-system & AHP.....	149

LIST OF FIGURES

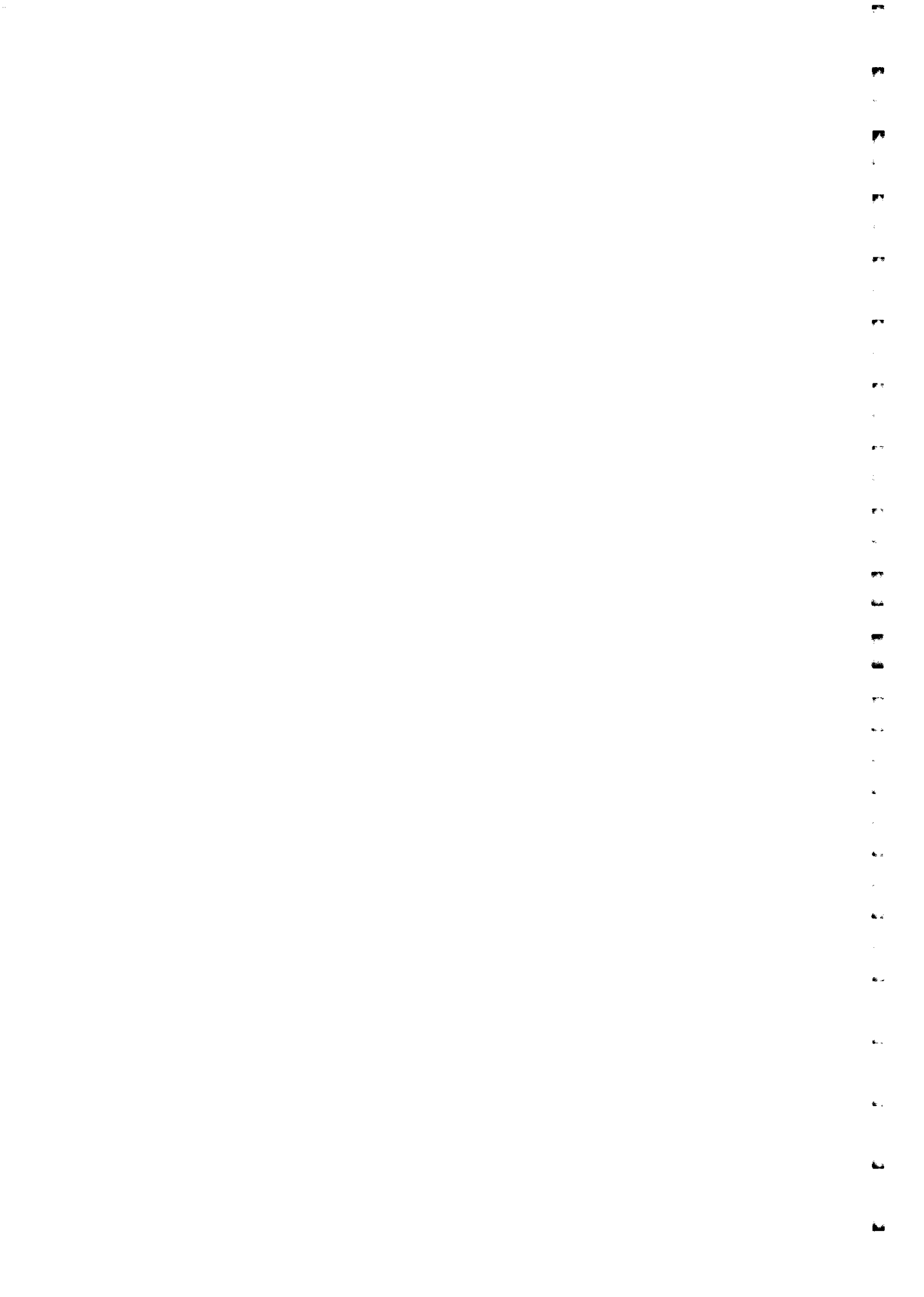
Figure 1.1: The Research Flow.....	14
Figure 2.1: Overview of a Data Grid Architecture [58]	19
Figure 2.2: Data access Model in Data Grid [1].....	23
Figure 2.3: Globus Major Components of Data Grid Architecture [94]	26
Figure 2.4: A Wide List of Simulators from [116].....	37
Figure 2.5: The EU Data Grid Architecture	40
Figure 2.6: Populations of Chromosomes in GA.	41
Figure 2.7 Pseudo-code of Genetic Algorithm.....	42
Figure 2.8: Single Point Crossover.....	43
Figure 2.9: Mutation in GA	43
Figure 2.10: The General Scheme of GA.....	46
Figure 3.1: A Flowchart of the Research Methodology	52
Figure 3.2: Visual Representations for Sites and their Model Values.....	53
Figure 3.3: D-SystemAp1 Utilizing 3 Parameters.....	55
Figure 3.4: Execution flows of the RB and Computing Element Threads [119]	59
Figure 3.5: Choices of Optimizer in OptorSim Configuration File.....	60
Figure 3.6: Grid Topology for CMS Test Bed	63
Figure 4.1: Overview of A-system.	70
Figure 4.2 (a) : Average Jobs' Total Time in Seconds for 100 Jobs.....	86
Figure 4.2 (b) : Average Jobs' Total Time in Seconds for 500 Jobs.....	87
Figure 4.2 (c) : Average Jobs' Total Time in Seconds for 1000 Jobs.....	87
Figure 5.1: Overview of D-system and other Related Entities.....	92

Figure 5.2 QoS Parameters in a Grid Distributed Between the Network and Computational Aspects.....	97
Figures 5.3 (a): Show the Performance of the 3 Systems for 20 Requests.....	105
Figures 5.3 (b): Show the Performance of the 3 Systems for 50 Requests	106
Figures 5.3 (c): Show the Performance of the 3 Systems for 100 Requests.....	106
Figure 5.4: Integrating Users' Preferences in Replica Selection System [44]	110
Figure 5.5 D-SystemAp1 Versus D-SystemAp2.....	118
Figure 5.6(a) : Random & OsBi Versus D-SystemAp2	119
Figure 5.6 (b) : Random & OsBi & D-SystemAp2 with a larger Requests	119
Figure 5.7 (a) D-SystemAp2, OsBi & the Random for 30 Requests, k=2-5.....	120
Figure 5.7 (b): D-SystemAp2, OsBi & the Random for 100 Requests, k=2-5.....	120
Figure 5.7 (c): D-SystemAp2, OsBi & the Random for 700 Requests, k=2-5.....	121
Figure 6.1 Chromosome Encoding.....	133
Figures 6.2 (a) One Point Crossover, (b) Mutation	134
Figure 6.3 Pairing Users with Sites before Mutation	135
Figure 6.4 Pairing Users with Sites after Mutation	135
Figure 6.5 Major Components and Structure of M-system	137

LIST OF ABBREVIATIONS

Analytical Hierarchy Process (AHP).....	34
Application Service Provisioning (ASP).....	2
Average Job Turnaround Time (AJTT).....	81
Bag-of-Tasks (BoT).....	2
Biomedical Information Research Network (BIRN).....	3
Center for Nuclear Research (CERN)	4
Computing Elements (CEs).....	2
Computing Site (CS)	32
Content Delivery Network (CDN)	24
Data Replication Service (DRS).....	26
Economic Model-Binomial (EB).....	82
Fair Users' Satisfactions (FUS).....	16
Fermi National Accelerator Laboratory (FNAL)	57
first come, first served (FCFS)	72
first in / first out (FIFO).....	155
Genetic Algorithm (GA).....	155
Global Grid Forum (GGF).....	26
Grid File Transfer Protocol (GridFTP).....	68
grid manager (GM).....	112
Grid Organization Manager (GOM).....	94
Grid Resource Information Service (GRIS)	68
Grid Security Infrastructure (GSI).....	17
Hewlett-Packard (HP).....	71
High Energy Physics (HEP)	39
human decision maker (DM).....	47
information technology (IT)	3
K-Nearest Neighbor (KNN)	30
Large Hadron Collider (LHC)	4
Least Recently Used (LRU)	82

local area network (LAN).....	71
MegaBytes per second (<i>MBps</i>).....	71
Meta-computing Directory Service (MDS).....	137
multiple criteria decision making (MCDM).....	47
Network Weather Service (NWS)	30
Open Science Grid (OSG).....	66
physical names of the files (PFN).....	20
quality of service (QoS).....	82
Replica File Transfer Service (RFT)	18
Replica Location Service (RLS).....	5
Replica Selection in Data Grid (RsDGrid).....	48
Replica's Site (RS)	32
Replication Manager (RM).....	39
Reputation Manager (ReMg).....	94
Resource Broker (RB)	12
Resource Broker (SRB).....	18
Self-Protection Capability (SPC).....	93
Single Trip Time (STT).....	32
standard deviation (<i>sd</i>).....	76
Storage Elements (SEs)	2
Storage Resource Broker (SRB).....	30
TAS (Time, Availability and Security)	16
The Laser Interferometer Gravitational Wave Observatory (LIGO).....	3
trust factor (TF)	93
UPQ (Users' Preferred Quality)	16
User's Preferred QoS (UPQ).....	16
Virtual Organizations (VO).....	25
Virtual Organizations (VOs)	4
wide area network (WAN)	71



CHAPZTER 1

INTRODUCTION

1.1 Background of Grid Computing & Data Grid Motivations

It has become a huge challenge to process massive datasets that have been accumulated due to the process of simulations or due to large-scale experiments in the next generation of the huge-scale scientific application that occur in different scientific disciplines, such as earth sciences, high-energy physics and molecular modeling [1-6]. It is crucial to have very high capacity resources such as, mass storage systems, supercomputers and high bandwidth networks, to manipulate huge datasets and distributing them to the scholars all over the world. Some of such applications, like resource owners' harmonization, co-operation, multi-domain applications and elimination of system boundaries, might need new techniques to manage important issues in this context. One of the important techniques in this domain is grid computing [7], in which an accumulation of huge-scale storage, networking and computing resources are employed.

The grid computing emerged to offer an environment, where globally scattered scientific communities might share their valuable resources, through different organizational and administrative domains, in order to sort out computation problems and huge-scale or data-intensive problems, in an attempt to cooperate in different fields. Consequently, grids allow the formation of virtual environments with an enormous pool of hardware and software resources, over different administrative domains. Such resources, categorized into storage or computing elements, can be transparently accessed by a large number (thousands or more) of grid users who are

geographically distributed. Grid resources [8, 9] might imply a combination of software and hardware that is placed in grid nodes, and offers specific computing or storing capabilities to the grid jobs and to users. A grid job or application is a computer software that accesses multiple grid resources, to process data and accomplish pre-determined objectives while a Grid service is a software that delivers an exact computing capability and does some errands on behalf of the users of the grid or on behalf of other grid systems [8, 10]. The Main Grid resources are: 1) Computing Elements (CEs): hardware that provide processing power to accomplish the tasks. 2) Storage Elements (SEs): hardware that stores data files. 3) Data files: electronic files that used as inputs to accomplish tasks and placed in the SEs.

The concepts utilized in grid computing are not new. They were inherited from earlier technologies such as 1) Networking and 2) Network operating systems, where both of them provide gaining access to resources crossways geographically scattered places [11]. By means of distributed systems, users can exploit extensively spread heterogeneous computers to execute jobs. These heterogeneous computers demand more resources than possibly available in local stations. The lack of resources in the local station to cater emergent needs and the demand for collaborative cost-effective and problem-solving have spurred the development of middleware. The middleware transparently offers access from dispersed resources and directs data from sources to users' applications in a smooth and reasonably scalable way. This was introduced as meta-computing, which later known as "computing on the Grid" or Grid computing [7]. Many different types of applications are potentially supported by grid computing including, intensive-computing applications, data-demanding applications and other applications that require distributed services. These applications provide the impetus for the development of several kinds of grids to support them. These different grids have been classified into six types namely: Computational Grids, Data Grids, Interaction Grids, Application Service Provisioning (ASP) Grids, Knowledge Grids and Utility Grids [12].

An example of the capabilities provided by grid systems is the ability to move files. Computational grids serve intensive-computations applications such as Monte Carlo simulations [13], and Bag-of-Tasks (BoT) applications [14], where each

application has many independent jobs to be executed. One more example of intensive-computations applications is the project known as Nimrod-G [15], which uses grids to schedule intensive-computations applications on available resources. Interaction grids lay the foundation of infrastructure and the facilities for users to act together in a real-time environment, e.g., Access Grid [16]. This type of grid is appropriate for multimedia applications, like, large-scale video conferencing. ASP grids focus on offering access to distant applications hosted on computational grids or data centers (e.g. NetSolve [17]). However, the Knowledge Grids are devoted to knowledge acquisition, data processing, and data management. Furthermore, they offer business analytical services, determined by incorporated data mining services. Common projects entitled under knowledge grids are the EU Data Mining Grid [18] and the Knowledge Grid [19]. The Utility Grid aims at delivering one or more of the aforementioned grid services to end-users, as Information Technology (IT) utilities, on the basis of pay-to-access. Well-known projects under this domain are the Utility Data Center [20] at the enterprise level, and Grid bus [21] at the global level.

Finally, Data Grids, which occupy a central position in this research, play a significant role in offering the infrastructure to access, share, move and coordinate huge datasets stored in scattered places across the world. Data grids are designed to meet the needs of the scientific research and collaborations, wherein there is a demand for analyzing huge amounts of data and sharing the results. The Laser Interferometer Gravitational Wave Observatory (LIGO) collaboration [22] hosts more than forty million files in ten locations and extensively replicates huge datasets. Experimental datasets are created at two LIGO instrument sites which afterward replicated throughout the LIGO locations in order to offer local access for scientists to access data. Other examples of data grids' applications can be found in several areas including astronomy [23], climate change modeling or climate change simulation [1, 3], high energy physics [24], clinical trials [25], Earth Observation [26], Biomedical Information Research Network (BIRN) [27, 28], and TeraGrid [29].

Significant amounts of datasets are required to explore any natural phenomenon and to carry out experiments in the aforementioned applications to obtain huge datasets outputs. These outputs must be gathered and properly stored in

geographically scattered data centers to facilitate their retrieval for additional processing at other sites [6, 23, 30-32].

Further examples can be listed in this concern. Among which are the experiments in high energy particle physics, such as those occurring at the European Center for Nuclear Research (CERN) (e.g., the ATLAS and CMS) [23, 33-35] and experiments taking place at the Large Hadron Collider (LHC). Both of these created and accumulated amounts of data which are usually processed and analyzed by several thousands of scientists across the world. Data and results are used by scientists as tools of measurement in any forthcoming experiments. These experiments can be fulfilled by providing several essential technologies in an organized manner to consolidate data-intensive petabyte-scale applications.

1.2 Replication in Data Grids

The main function of data grids is to offer services and infrastructure for scattered data-intensive applications. These applications require accessing, moving and amending enormous datasets that are located in globally distributed storage servers. The main goal of data grids is to serve as a concrete base that offers users the following facilities: a) the capability to manage various identical copies of their data, b) the capability of searching over a huge number of available datasets to find the most appropriate data location, c) the capability of moving huge datasets amongst resources in suitable time, and d) the capability to manage access permissions for the datasets, based on the local policies that should be applied on Virtual Organizations (VOs). Hence, pairing high-performance networking and wide-area storage management techniques is a primary objective of data grids.

In order to realize these capabilities, a data grid is required to provide services to handle collaborative access to datasets, like the data grid used to administer authentications, to grant permissions and authorizations to users across the world. Other main services offered by the data grid are: a) Data replication, in which duplicated copies of the same dataset are produced and stored at a number of disseminated sites to confirm accessing the needed datasets in the fastest and/or low-

cost way. b) Search algorithms, to allow users finding out the needed datasets that could be available within the collaboration. c) Resources' management services to enable the users and the applications to use the infrastructure efficiently at which datasets are accessed at the less loaded resources which in turn could provide better turnaround times and low costs.

In general, sharing is one of the main functions of data grids. It is a demanding problem in grid environment due to the following points: a) Massive quantity of datasets to be shared, b) The network bandwidth and the storage space are limited and c) The heterogeneity of the resources. Moreover, the resources are assorted, as they are acquired from various administrative domains in a distributed environment. It is impracticable for all users to access a single instance of data, located at a single organization, because this leads to a drastic increase of the data access latency. On the other hand, a single organization alone might not be able to hold such a huge volume of data. One solution to the problem is data replication, where, duplicated copies of each datasets are produced and circulated to several different grid sites to increase data accessibility, availability and reliability.

For instance, let's consider the data distribution that follows a hierarchical structure such as the experiments of CMS and ATLAS, where, the data grid organization spans universal scattered locations, and is arranged in "Tiers". Tier 0 denotes the main site located at CERN (European Center for Nuclear Research), Tier 1 comprises national centers, Tier 2 characterizes regional centers that cover either one small country or one region of a large country such as a state in the USA, Tier 3 presents workgroup servers, and Tier 4 which represents the (thousands of) desktops or workstations used by the scholars or scientists. In this scenario, all datasets are gathered at CERN, situated in Geneva, Switzerland. They are preprocessed online and saved in the CERN computer center (Tier 0), in the data grid hierarchy. Subsets of that datasets are next replicated at national centers in Canada, France, the USA, Germany, Italy, and so on. While, smaller subsets of the datasets are replicated at individual foundations, like individual universities. For example, some of the services offered by the grid is the Replica Location Service (RLS) [5] which allows the users to find all replica physical locations of a given data file. A scientist in Pisa can utilize the RLS to get the locations of the data originally collected at the experiment site and

its replicas. It is a potential that, the data is already placed on several grid locations across the world, maybe one of them is very close to Pisa or at Pisa itself, which can provide faster replica transfer.

Replications have been proven to be a concrete and effective technique to attain high network performance in distributed environments [6, 36]. Producing replicas efficiently, assigns users' requests to different grid sites, and offers different access options rather than a single datasets location. Meanwhile, the load on the originating server is circulated across data grid sites. This consequently and significantly reduces the load that might be fallen on the originating servers. Furthermore, the network load is also distributed crossways various network paths, thus reducing the prospect of congestion-related performance deprivation.

1.3 Motivation for Replica Selection in Data Grids

The previous section has presented the importance of replication in data grids. However, generating multiple replicas for the datasets and distributing them to different locations introduces the replica selection problem, where each replica location has its own capabilities (QoS specifications) and each user could has his/her own QoS preferences as well.

Replica selection is a critical decision that cannot be performed manually because of the huge number of replicas, the huge sizes of the datasets and the huge number of users. On the other hand, the resources are limited and the users are competing over them. Other problem is sharing, especially millions of files (usually very huge), will be generated from scientific experiments. These data files will be accessed by researchers, developers and application providers around the world [37].

Moreover, the replicas vary from small sizes files to huge data files (terabytes or even petabytes). These replicas belong to various grid sites that dynamically changed which require complicated cost model to evaluate their statuses [38]. Sites' local policies (i.e. site availability) are also obstacles where the users are allowed to access the grid sites in specific times and for specific times. It is reported in [39] that once a

user is allowed to gain access to a resource based the access policy, the usage Service Level Agreement (SLA) determines how much of the resources the user is permitted to use.

1.4 Problem Background

The core issues addressed in this thesis is how to select the best replica location amongst many replicas scattered across the data grid sites in a manner that achieve not only efficiency and tradeoff between available QoS and users' preferences but also by attaining fair users' satisfactions. Based on [40], the executed tasks are subject to failures due to infected hardware, software vulnerability, network failure, overloaded resources and non-availability of resource. Furthermore, the same datasets may have location-dependent access costs the same as the tangible goods in economies field have [41]. It is reported in [42] that, some users attempt to optimize the mapping to the required replica depending on the billing costs of their network operators or hosting services, especially if a 95th percentile billing mechanisms are imposed for the services. Cost minimization can be achieved by decreasing the frequency of peak consumption or by taking care not to exceed their budgeted rates. Delayed accesses that may occur due to resources non-response or disconnections may cause unwanted consequences. This in turn can make the effective accesses and availability are precarious in numerous data intensive applications [5, 43]. The users need to get the best QoS parameters (like response time, security, site availability and cost) in their site selections. However, these parameters are usually conflicting, which means achieving one might negatively affect the others. As a result, integrating users' preferences is required to direct or focus the selection.

Quality has many different meaning for different users, in different environments and cases as well. This could be changed over time with accordance to the user's priorities [9]. For instance, at a given time, a user might decide that, a specific service should be executed, with the minimum cost, at the shortest time. Later on, the same user and for some reason might change his priorities, by making a new decision that the service should be accomplished with a high level of security no matter what the cost is. Quality is usually measured with respect to performance; therefore improving

quality is considered as an optimization technique. Generally, quality can be expressed by one or more of the nonfunctional characteristics such as, security, response time, cost, the duration of resource availability, and performance. Consequently, QoS becomes a significant challenge in grid environment because of the deviation in the availability of resource duration and the probability of system failures.

Ideally, any grid users need to receive their required replicas in the shortest turnaround time with the high level of QoS. Accomplishing high level of QoS to a certain user increases his/her satisfactions, which could have a negative impact on other user or users' satisfactions. However, the demand in grid environment is to attain equal users' satisfactions as much as possible. Therefore, the replica selection as critical decision should be aware how to achieve equal users' satisfactions in order to efficiently distribute replicas amongst the grid users.

The main objective is to deliver the required replicas to grid users in a suitable time duration, suitable cost, and reliable secure manner and at the same time to achieve fair users' satisfaction. Because of the network cost in transferring huge replicas through computer networks, the cost of the replica itself, the risks of security breaches, and the risks of faults due to disconnections, there are a lot of advantages from selecting the optimal replica location. The key factor to proficiently select the best replica is to precisely identify and utilize the accurate set of parameters that guide or play roles in the selection process. Replica selection [44] is a high level optimization services that targets selecting the best replica location among many scattered across the world, to fulfill and meet users' requirements. In this context, the best means, the most appropriate replica location for a specific user according to his preferences. Since each grid user has different preferences, the best replica location is different from one user to another. The replicas' locations are the grid sites that contain storage elements to hold replicas.

Thus, the parameters that integrated in the selection process are: turnaround time, site availability, security and cost. The parameters are contradictory with each other. This means that, a parameter of high quality will be considered at the expenses of the others parameter because it is very rare to find a site, which is the best with respect to

all parameters. Satisfying all users might be conflicting, because fully satisfying some users might drastically dissatisfy the others.

The QoS parameters are heterogeneous requiring different metrics to measure them. Some parameters can be measured using time metrics like turnaround-time and the site availability, which can be measured in seconds or hours. On the other hand, cost can be measured in any currency for example, Malaysian Ringgit, while security requires a new metrics to measure it. The main question that arises here is how to aggregate these parameters together, in order to make a decision?

The complication of this problem is illustrated in Table 1.1, which draws a fictional simple scenario of a data grid in which replica is held by five sites and each site has four parameters. Each parameter is rated out of (100).The best value for the parameter is (100) while the (0) represents the worst value for the parameter. The question here is: which is the best site? (Which is the best replica location?) It is obvious that, site 3 is the best in terms of availability, security, and cost, but unfortunately the worst in terms of turnaround time. Site 4 has the best turnaround time and good availability but the worst in terms of security and very low quality in terms of cost. This is an example or a case that deals only with five sites. The complexity of the case becomes a real one when the number of sites is increased to thousands which is anticipated to take place in the nearer future. Apparently, selecting the best site is a challenging and sophisticated decision that is why a concrete technique for this decision is required.

Table 1.1: Example of Sites and Their Parameters Rates' Values

Site ID	Availability rate	Security rate	Turnaround time rate	Cost rate
Site 1	80%	40%	75%	30%
Site 2	90%	40%	70%	50%
Site 3	95%	85%	40%	90%
Site 4	85%	30%	95%	60%
Site 5	80%	50%	75%	55%

1.5 Problem Statement

This research intends to help scholars in completing their computerized collaborative jobs and receiving their valuable huge data in timely manner with high QoS and to attain fairness amongst them. However, achieving these requires a perfect replica selection algorithm. The complications of the replica selection challenging problem have increased, because of various QoS parameters or factors that play roles in the selection decision. The turnaround time is the only QoS factor that has been studied by the previous researches. The observations or the investigations in this study have revealed that there are other QoS factors that concern the users and the overall grid performance. Among many QoS parameters, security, site availability and cost have been selected in addition to turnaround time, which has been already intensively addressed in previous studies [45-48]. Subsequently, a super replica selection strategy is required to improve data grids systems because the current replica selection algorithms still require more optimization techniques due to the following reasons:

1. A very limited number of studies analyzed the replica selection algorithms in data Grids in order to identify the needed QoS parameters rather than time in the selection process.
2. None of the current replica selection algorithms considers sites' local policies specifically site availability in the selection process which increases jobs' times or leads to faults.
3. A very limited number of algorithms that integrates the QoS parameters in the replica selection process in order to reduce jobs' time, security breaches and cost.
4. Little number of models that integrates QoS parameter in the replica selection process.
5. Lack of studies that integrates users' preferences in the replica selection process in order to satisfy them.

6. One study only, integrates QoS parameters rather than time with the aim to achieve users' satisfactions amongst grid users.

In consequence, replica selection challenging problem can be studied by posing four critical questions as stated below:-

1. What are the QoS parameters that should be considered when making a replica selection decision? And how to model them?
2. Does considering the site availability QoS parameter in the replica selection process decrease jobs' times?
3. How to choose the best replica location amongst lots of locations scattered crossways the data grid sites with high level of QoS?
4. How to consider and to integrate users' preferences in the replica selection process in order to satisfy data grid users?
5. How to achieve equal users' satisfactions as much as possible, amongst data grid users?

1.6 Thesis Goal

The ultimate goal of this research is to design a new family of replica selection systems that can break administrative domains and organizational barriers in order to efficiently utilize the shared and distributed resources.

1.7 Thesis Objectives

The key concern of this thesis is to increase the satisfaction level of the grid users. In typical grid environment, grid users submit their jobs to the Resource Broker (RB), which in turn locates the best grid site to carry out the jobs. Grid site [30] is a grid node in the network that contains CEs and /or SEs. Each grid site has its own power depending on the existing number of CEs, SEs in the site and their speeds, its

bandwidth and its capability to protect itself and the users against security attacks. Grid users [10] are the individuals who seek to gain access the grid, in order to use and benefit from its resources. Grid users may directly use grid resources by sending their jobs to the Resource Broker (RB), which matches the best resources to the intended jobs.

In data grids, usually each job under execution requires many huge datasets that are scattered across the grid sites. In this thesis, improved replica selection systems are proposed to deliver to users' jobs, the required replicas in suitable QoS parameters that are in line with users' preferences. Consequently, this research attempts to achieve the following objectives:

- To study and analyze the current replica selection systems in order to identify the QoS parameters required in the selection process.
- To design systems that integrates the site availability QoS parameter in the replica selection process in order to reduce jobs' times or faults.
- To design a replica selection system that integrates simultaneously several QoS parameters and users' preferences in order to reduce jobs' times, security breaches and cost, and satisfy the users.
- To design a system that integrates QoS parameters and achieve almost (or as can as possible) equal users' satisfactions amongst grid users.
- To validate the systems and check their efficiencies.

In this thesis, the terms: *site*, *grid site*, and *node* are used interchangeably, for the same meaning and as well *grid users* and *users*.

1.8 The Scope of the Study

The concentration in this research is on read-only data similar to most data grids in reality, while there are a very few dynamic updates, because grid users mostly use a "load" rather than an "update" strategy [1,14]. In this context, the data consistency is

not considered in this research. In addition to that, this thesis focuses on newly proposed QoS parameters; site availability, security, and cost, due to their high potential impact on the users' satisfactions in the context of replica selection process. Moreover, this research has been built under the assumption that the sites' QoS parameters rates are obtained from the Information Service Providers (ISP) or to be provided through the logs files, once a data grid site is logged on. In addition to that, this thesis is focused to the cases, where the number of replicas is greater than the number of users in each scheduling cycle.

1.9 Research Framework

As mentioned in the previous section, the main objectives of this research are to identify the QoS parameters required in the selection process and to design replica selection systems that consider them. This section provides a brief overview of the research framework to achieve these objectives. More details and the methodology are presented in chapter 3. To answer the first question of this research, a complete review in the literature has been undertaken on the replica selection area in data grids with the focus on the QoS parameters that should be utilized in the selection process. Next to that, the problem has been formulated and the QoS parameters have been modeled in the systems through chapters 4 to 6.

To answer the second research question the replica selection has been modeled as a multi-objective problem, and a new system has been proposed and adapted. The new system integrates site availability QoS parameter in the selection process; this method is basic and termed in this research as Site Availability Based Replica Selection System (A-system). Although the performance of (A-system) is magnificent, it lacks other QoS parameters and users' preferences. A-system deserves more research in order to answer the third and the fourth research questions. Therefore, a Decision Maker Replica Selection System (D-system) has been introduced to further answer the third and the fourth questions of the research.

The D-system is a modification to A-system by imposing some parameters and users' preferences. Some weaknesses have been revealed after implementing the D-

system related to the fair users' satisfactions. This has led the research to address the problem as multi-objective optimization problem to answer its fifth question. For the fifth research question, the Genetic algorithm has been hybridized with the D-system to improve the fairness received by the users.

Finally, a comparative analysis has been conducted to compare the three proposed systems (A-system, D-system, and M-system) with some other methods, as well as to compare the performance of the three systems among each other through chapters 4, 5 and 6. Figure 1.1 illustrates the framework of this research.

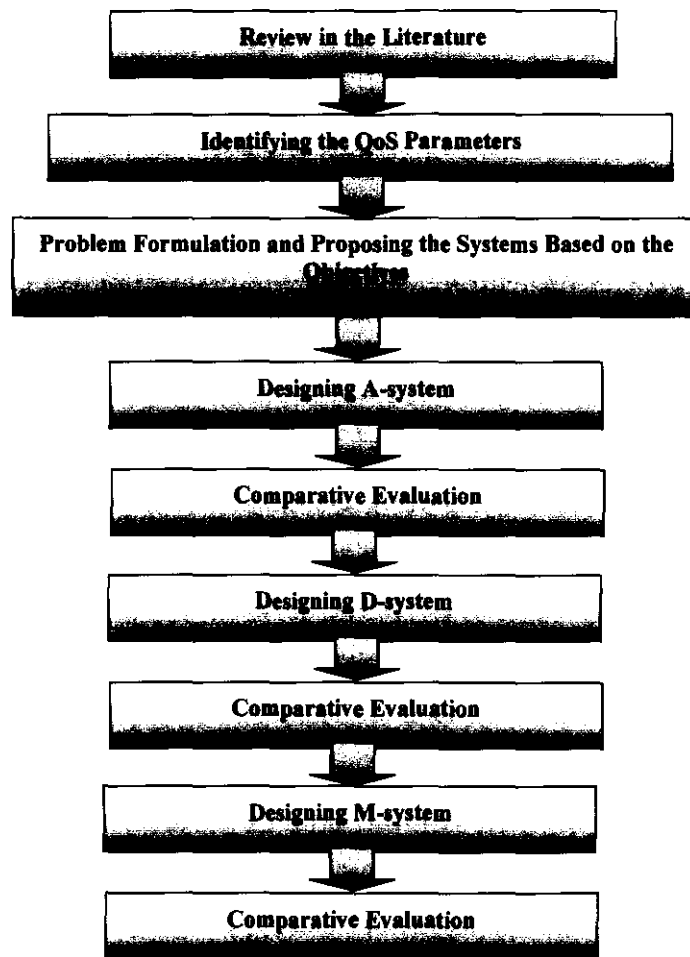


Figure1.1: The Research Flow

1.10 The Importance of the Study

The proposed replica selection systems are very useful for grid users, especially academic researchers and scientists who require sharing scattered huge datasets

collected by sensors distributed across the world or generated from scientific experiments. The systems contributed to the state of the art by delivering the required replicas to the scientists in two ways: (i) by providing the names of the replicas, directly from the scientists, and then the systems in return collect the available locations for the replicas, later, the best replica location will be selected, (ii) the academics' jobs that under execution require replicas, consequently, the systems collect the available locations, then decide the best one, after that, use other services to transfer the files to the underlying location, where the job is being executed. The proposed systems assist the academic researchers to perform their jobs transparently, fairly and effectively.

1.11 Thesis Layout

The remaining of this thesis is organized as follows:

Chapter 2 presents a brief critical survey of the relevant existing studies. The chapter is divided into four main parts: The first part explores data grid and replica management service. The second part explores the related works of replica management strategies. The third part tackles the related works to the replica selection strategies such as the clustering model, multi-objective decision making, and genetic algorithm. The fourth part is the proposed solution to the underlying research problem.

Chapter 3 defines the modeling and representation of the proposed systems, and it explains the overall methodology of this research.

Chapter 4 introduces a single user basic replica selection system. The system imposes the site availability QoS parameter in the selection process. The system requirements, design, components, and algorithms are explained. Some examples are stated, to expose the functionality of the proposed system. The performance evaluation and metrics regarding the basic replica selection system are discussed in this chapter. The job turnaround time and the grid simulator "OptorSim" original replica selection system are used as benchmarks to evaluate the efficiency of the

proposed system. The results produced from the simulation are discussed and compared with other similar systems.

Chapter 5 extends the single user basic replica selection system by imposing more QoS parameters specifically, security and users' preferences. The system requirements, design, components, and algorithms are explained. Some examples are stated, to clearly expose the functionality of the proposed system. The performance evaluation and metrics regarding the basic replica selection system are discussed in this chapter. Two new performance metrics were proposed: TAS (Time, Availability and Security) and UPQ (Users' Preferred Quality) and both of them to gather with OptorSim" original replica selection system are used as benchmarks to evaluate the system. Simulation results are discussed and compared with other similar systems in the literature.

Chapter 6 introduces a multi user replica selection system, which integrates time, availability, security and cost QoS parameters, in addition to users' preferences. The system requirements, design, components, mathematical model and algorithm are explained. Some examples are stated to measure and evaluate the functionality of the proposed system. The performance evaluation and the new proposed metrics regarding the multi user replica selection system are discussed in this chapter. The new metrics are Fair Users' Satisfactions (FUS) and the Average User's Preferred QoS (\overline{UPQ}) metric which are used as benchmarks to evaluate the proposed system. The results produced from the simulation are discussed and compared with the system proposed in chapter 4.

Chapter 7 discusses the conclusions according to the results that have been obtained from the proposed systems. The feasibility and worthy of the proposed systems are presented. Suggestions to the future works are provided.

CHAPTER 2

AN OVERVIEW AND LITERATURE REVIEW

2.1 Overview

This chapter focuses on the idea of data grid architecture elaboration and presents an overview of data management related to the domains of this research. These domains include metadata service, the needs and the motivations for replication strategies, Globus Toolkit grids and related data-intensive studies. Various types of currently available replica selection techniques are also discussed. Analysis of the features and the limitations on the state-of-the-art of the replica selection techniques are tackled in this study. The simulation survey, the K-Means model, Genetic Algorithm and multi-objective optimization model are presented. Finally, the proposed solution is presented in this chapter, while the design and the implementation of the proposed solutions are discussed in details in the following chapters.

2.2 Data Grid Architecture

In late 90's various researchers from different educational organizations and higher learning institutes have attempted to build computational and data grids of which Globus team has taken leading efforts [7, 49-51]. A structure for constructing grids depending on the service-oriented architecture has been provided by the Globus project. The design of the structures has offered the following services: 1) Securityservices,2) Information services, 3) Resource management services, and 4) Data management's services [7, 50, 51]. The toolkit with respect to the security matters depends on the Grid Security Infrastructure (GSI) that offers a lot of

security aspects which equip the users with facilitates to substantiate their communication and to provide them with the mechanism to make one log on in order to access the permitted grid resources and services. The Information Services offer information about the status of grid resource with a notification approach, where the current status is published by the resource and the updates about specific instruments, machines, or storage components are received by the resources' subscribers.

Thus, Information Services facilitates both resources monitoring and resources discovery as well. Inputs from the information services are used by components of the resource management to enable the users utilizing the existing resources and to help the system to manage resource allocations. On the other hand, the potential to manage and access data resources through the grid nodes or sites is offered by data grid [1, 6]. Furthermore, the Globus toolkit offers a lot of components to move, to make a duplicate copy (or copies) and to locate data, including GridFTP, RFT, and RLS [1, 49, 51, 52]. The GridFTP provides tools that enable secure, fast and parallel datasets movement in the data grid (or among data grid sites). Managing several GridFTP transmissions are facilitated efficiently and most reliably by the Replica File Transfer Service (RFT). RLS retains and offers access's information of the available datasets and their locations within the data grid [1, 5, 6].

Replica catalogs are registries used to maintain the records for all the shared datasets and objects in the grid sites. These catalogs are used by the RLS to register, index, and locate data. Each record of the replica catalog comprises the locations of all the object's replicas and provides mapping for the object name and its replicas. The Resource Broker (SRB), also known as Storage Resource Broker, is a client-server based middleware that offers similar service as RLS [53, 54]. SRB offers a management system for data replica and a uniform single interface. The assorted and distributed data is managed by SRB to enable the users to seamlessly access the files and database. The SRB offers an integrated transparent view of the data files that are stored in different media and locations, so that the users can feel the scattered data appear as if they are locally stored in their computes [55]. Data replication is applicable in SRB if the data is required to be much closer to the user [56]. Several forms of replicas can be created using SRB or can be obtained from outside the

system. Figure 2.1, which is reproduced from what is reported in [2, 6], illustrates the above mentioned elements of the data grid which is structured as a layered architecture,. This structure is exemplified or derived from definitions and proposals introduced in [1, 2, 6, 51, 57, 58]. It is noteworthy that the elements at the same level can collaborate to provide some specific services while those at higher levels can use the services available at the lower levels as well as on top of them.

Workstations, supercomputers, scientific tools and storage devices like disks and tapes are the components of the grid fabric. However, a lot of these above mentioned resources, which represent the physical layer, are broadly scattered and linked with high bandwidth and wide area networks. Whereas, operating systems and software, which administer these scattered heterogynous resources constitute the basic software layer of the data grid. The process of copying data from resources in the grid fabric layer is facilitated by the data transfer protocols reside, in the connectivity layer. These data transfer protocols rely on the GridFTP communication and authentication protocols to authenticate the identity of users and guarantee security and data integrity. The data grid services layer comprises the core services such as, replication and resource monitoring which offer transparent identification, location, and access to data and computer resources.

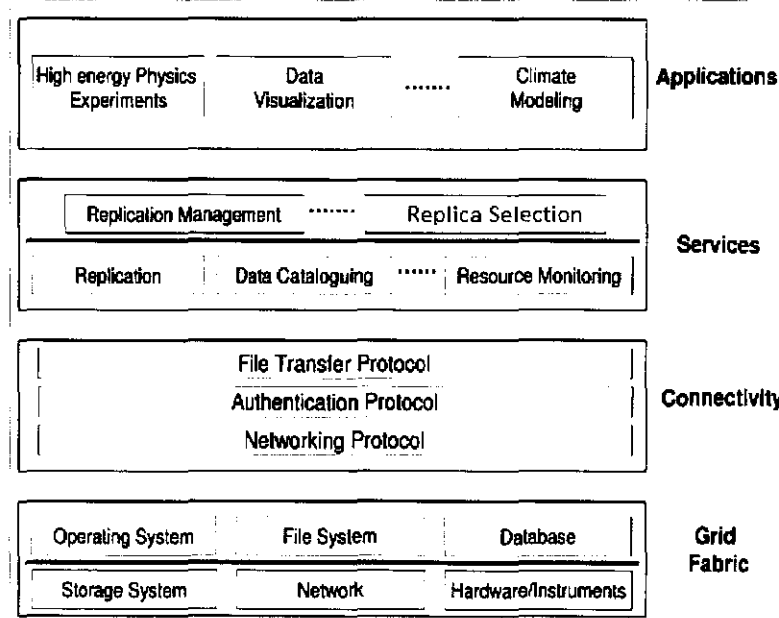


Figure 2.1: Overview of a Data Grid Architecture [58]

These core services can be employed to assist the higher services to identify and implement access policies of the resources. The monitoring service is similar to the Information Services as well as their implementation in the Globus architecture [51]. The subsequent components of the services layer offer superior level services that utilize the services of lower level to facilitate a well-organized management, provision of replicas and data resources on the data grid. The combined services offered at this layer signify the data grid middleware. The middleware abridges the grid services to facilitate administering access to resources and provide API's for users and applications in order to make a full use of utilities available in the data grid. The applications layer offers services and access interfaces, which are specifically designed to a research group, a community or a virtual organization. These services invoke and customize the services provided by the lower layers to match the target domains such as, high energy physics, biology and climate modeling.

2.3 Data Management in Data Grids

In general, data management is an important element in data grids. It does not only manage data flow among physical storage systems, but also provides metadata, replication of the application data, knowledge about the replica locations, executes data access protocols, and enforces security policies [5, 31, 35, 57, 59]. In a typical data grid, the element responsible for data management services consists of protocols and services which spans multiple resources, and belongs to the service layer [1, 2, 5, 7, 51, 59-61]. It is reported in [1, 2, 5, 59] that data management is composed of the following two major services: a) the reliable replication service and b) the metadata service. The reliable replication service provides information about the physical names of the files (PFN) and the objects the users are interested in. This service transfers these files and objects to destinations required by the user.

The metadata access service offers tools for publishing and accessing metadata, and offers systems for accessing and organizing information about the data stored in storage systems. This means that the metadata service provides information about the names of logic files or objects based on users' access attributes. Other functions of this service include: data access consistency policy, access control, authorization,

auditing, etc. The metadata service is designed to facilitate an efficient means of naming, creating, and retrieving the metadata. The detailed discussion about metadata service is presented in the next subsection.

As explained in the above section, the services layer is branched as high-level and low-level sub-layers. The former involves in upper layer services such as, replication management, replica selection optimization, and resource allocation. The low-level sub-layers are utilized by the high-level sub-layers to enhance the quality of service. The process of managing the quantity of replicas and their locations in the grid sites in order to optimize the utilization of grid resources is known as replication management service. Nevertheless, users get best replica location or the replica required by their jobs using replica selection service. The upper level gets the services such as replication, data cataloguing, and resource monitoring from the low-level services at the lower layer.

The data catalogue service deals with vital processes such as: recording all replicas and their physical locations on the grid sites, registering the newly created replicas, and removing the replicas that are intended to be eradicate from the registry by the replication management service. However, the replication management service is distinct from the replication service because the former does the function of determination, while the later does the function of execution. For instance, if the replication management determines to generate a new copy of a certain replica, the new copy of the particular dataset is created by the replication service. Afterwards, the replication management employs data transfer service to move the copy (replica) to a site that has been concluded as an essential site location by replication management services.

This thesis focuses on the high-level service layer, in particular, the data management system and the replica selection services, while other services and other layers are less highlighted. The current design of the data management system [35, 57], is not adequate to address all data replication service issues. For example, replica selection is a critical issue that significantly impacts data grid performance, but the current design lacks some elements among which models and parameters. Concerning the QoS parameters, the current design lacks site availability, cost and users'

preferences. In relation to models, the current design lacks a reliable model for security and a proper model to enhance users fairly satisfaction. These models and parameters are highly critical in data grid and require solid mechanisms in the replica selection process.

2.3.1 Metadata Service

Data grid enables the users to transparently access the globally distributed data and services through the data management service [1, 2, 5, 6, 59, 62, 63]. As reported in [1, 2, 5, 59, 62, 63], the data access model is defined as follows and is shown in Figure 2.2. An application program sends a query to the data management service for data access with particular attribute values. When queries are sent to the data management service, it consults the metadata service to find the set of interests based data objects on attributes provided in the query. The metadata service responds to the query by generating a list of logical data object names, based on the given aspects or attributes. Then, the application program again sends a request to obtain the physical locations of the files of interests. In the same manner, the replica location service responds to the query by issuing list of physical file names that matches the logical file names. Afterwards, the list of physical file names and their locations' performance measurements and predictions are sent back to the data management service. Subsequently, the replica selection service selects the best locations for accessing the files and returns the location information of the chosen replica to the requester. Finally, the application sends an access request to each data site and performs data transfer operations if necessary.

In a data grid, there might be many data sources and according to the data access model discussed above, multiple requests are issued to access data objects from different sources. Thus, selection of data objects from different data sources can be considered independently. A data selection scenario is illustrated by the Figure 2.2, where the metadata service, replica management service, and replica selection service are consulted by application to establish the most excellent source of data which corresponds with set of preferred data attributes.

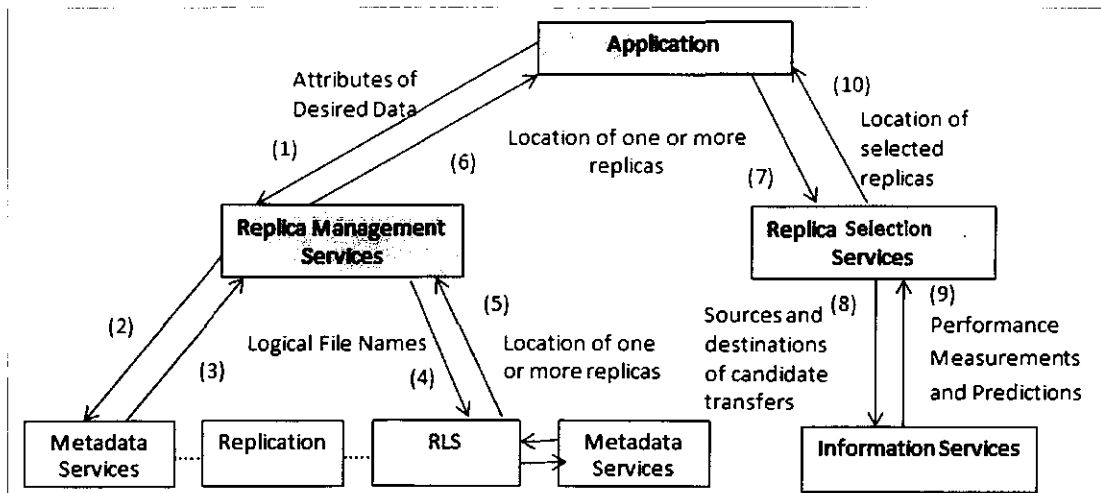


Figure 2.2: Data access Model in Data Grid [1]

2.3.2 The Needs and the Motivations for Replication Strategies

The difference between the replication management service and the replication service has been presented in the previous section. In general, the replication management service is in charge of making a decision that is executed by the replication service. In fact, usually scientific experiments or other applications, produce huge datasets that are required by many users across the world. However, accessing a single dataset by all data grid users is not feasible and requires a tool (solution) to be conducted. Data replication [64-66] is the solution where a determined number of duplications to the same dataset are produced and then distributed (Replicas) into grid sites through a technique that is called replication strategies. Replication strategy [67] is a number of policies that decides which dataset is to be replicated, how many replicas are required for each specific dataset, and where to place the newly created replicas. The terms data replication, data replication service, replications strategies, and replication management are interchangeably used in the literature with the identical meaning. One objective of data replication is to decrease response time, which in turn decreases the turnaround time of jobs.

The terms used in this thesis such as, time, response time and turnaround time, have the same meaning. Response time indicates the time [36] passed between requesting and getting the data file into the local storage, which includes: file transfer time, and waiting time. On the other hand, turnaround job time [68] depicts the time

spans between submission and accomplishment of the job. The turnaround job time comprises both, turnaround time and CPU time (processing time). The replication strategy intends to positively influence the network bandwidth at the time the number of replicas is efficiently balanced and the replicas are distributed across grid sites. However, the replication strategies negatively affect the network bandwidth when the number of replicas is not proportional to the appropriate demand of replica. In literature, replication has been comprehensively investigated, and various distributed replica management strategies have been introduced [11, 69-77]. Conventionally replication is employed to enhance the system performance by escalating its accessibility, reliability and it is utilized to ameliorate the availability of data.

Replication is applied in an extensive range of applications, such as, web services [78-80], distributed object systems [81], content distribution networks [82-84], and grid systems [44, 85]. Replication as a method can be deployed for other objects rather than the data files. For example, replication can be deployed in web services and replication scheduler [80, 86]. Indeed, the web services or the scheduler can be replicated to increase the services reliability and availability.

2.3.3 Related Data-Intensive Application Domains

Surveying interrelated research areas related to data grids, specifically, the domain of distributed data-intensive applications can boost the proposed research problem with already explored similar complications in other areas. Distributed data-intensive applications like Content Delivery Network (CDN), web applications, and distributed databases have some characteristics in common with data grids such as replication and selection databases. A Content Delivery Network (CDN) [87, 88] comprises a group of (non-origin) servers that offer contents such as: web pages, streaming media, and real-time on behalf of the content provider. Precisely, CDN comprises the academic and commercial architectures, where the former depends on client-server network such as, Akamai [89]. Nevertheless, in the academic architecture, the CDN depends on peer-to-peer such as Gnutella [90], which is based on the content providers, to become a part of servers.

Generally in web applications, data replication is employed as web caching [91]. The documents that are very often used are replicated and stored in the servers. The users who request the documents most frequently will be closer to the servers for the purpose of minimizing web response suspension and loads of server. Nevertheless, web caching handles the web pages documents, which is small in size when compared with the size of data used in data grids. Moreover, in data grids, data exhibit some or special localities that do not exist in other domains [36].

A distributed database [92] has emerged to meet the needs of big enterprisers, to augment the availability and consistency of data by duplicating the databases into various locations. Maintaining the consistency of data is one of the crucial challenges of database as the majority of the transactions are of data updates type. This type is in contrast with data grids datasets which are read-only type [58].

Basically, it may be agreed that data grids share many characteristics with other types of data intensive domains. However, the major characteristics of data grids that differ from the relative data-intensive applications and domains are: heavy computational requirements, wider heterogeneity, and huge volume of data, exhibition of special data localities, and the presence of Virtual Organizations (VO) [51].

On the other hand, establishing new various organizational disciplines and surfacing of scientific research, which generates enormous amount of read-only data files to be shared by a lot of grid users all over the world will be anticipated.

2.3.4 Globus Toolkit

As defined and explained by Ian Foster [93] Globus is:

- Communities of grid users who cooperate with each other on sharing of grid resources through collaborate, organizational, and geographic borders. Globus is also a specific group of developers, for a documented open source software in order to construct grids and grid based applications.
- These communities are supported by infrastructure like: code pools, problem tracking systems, interfaces, protocols and email lists.

- The supported software comprises a group of libraries and programs for solving common problems that occur when constructing distributed systems, services and applications.

The Globus data grid architecture [6] is divided into two essential folds: high-level and core services, as presented in Figure 2.3. The structural organization describes the options for utilizing the core services, to form the high-level service like data management services and complicated storage management systems, like SRB [49, 53, 54] that could share common low level services.

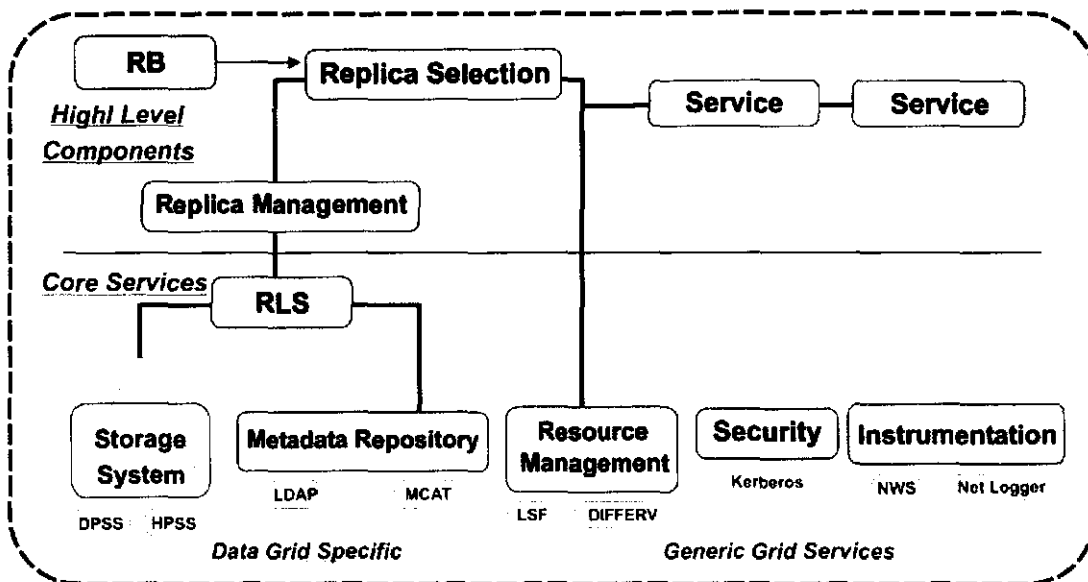


Figure 2.3: Globus Major Components of Data Grid Architecture [94]

Replications in Globus are done by RLS and Data Replication Service (DRS). RLS provides a framework for registering the physical locations of replica, which is mapped with the corresponding logical file names. The DRS ensures that the required replicas are stored in the local storage. In case those replicas are not stored, the required replicas will be cached from other grid sites. The new replicated data will be transferred by GridFTP and registered in the RLS for future decisions. GridFTP [52, 95, 96] is a data transfer protocol proposed by the Global Grid Forum (GGF) [1, 97, 98] to provide a secured data movement among grid sites in the grid environment and used by the Globus Toolkit.

2.4 Replica Selection Strategies

In a general data grid environment, multiple copies of the same dataset are produced by the replication systems and are distributed to different areas. These locations differ in their potentials, assets, and network requirements. Therefore, there is a considerable disparity in choosing a particular replica location from many widely distributed locations [48]. Thereby, it is possible to focus on the replica selection strategies for the purpose of selecting the best replica location depending on some factors. The crucial challenge of any replica selection strategy is identifying the suitable standards to establish the best replica location, and the selection algorithm employed for replica selection strategy.

In this section, the works related to various replica selection algorithms have been discussed. The literature survey in the context replica selection reveals two classes of selection algorithms. The first class comprises heuristically based and statically configured algorithms. The second class dynamically assigns the needs of clients depending on measured conditions (selection criteria set).

2.4.1 Heuristically & Statically Configured Replica Selection Algorithms

The first class covers heuristically based, statically configured algorithms. Random, round robin, and proportional algorithms are the most popular algorithms in this class [99]. The round robin algorithm rotates client request (transaction) among replicas in the grid, while the random algorithm assigns each user request to one of the replicas randomly. These two algorithms presume that, each replica location has identical processing capabilities and turnaround time. Without this assumption, the round robin and random algorithms will not work efficiently but, the advantage is no dynamic measurements are needed for either algorithm. The third type of algorithms employs a proportional algorithm, which proportionally disseminates the client requests among replicas to corresponding power ratings allocated by the system administrator. The down sides of this approach is its depending on the system administrator which might be erroneous, and it is based on information that has been

gathered(off-line) prior to the selection process, and thus, the performance of system will be very poor, especially if the information collected are inaccurate.

2.4.2 Dynamic Replica Selection Algorithms

The second class of selection algorithms dynamically assigns client requests, based on measured conditions (selection criteria set). Greedy and weighted algorithms are the most widely recognized algorithms in this class [79]. In greedy algorithms, the best replica location that exposes the best criteria values is selected as a local optimum optimization that assumes to lead to the global optimum. However, the selection exhibits a well-known undesirable behavior where transactions oscillate in groups among available replicas [81].

The weighted algorithm pairs each request to the replica, estimated to give best performance against the criteria (different systems adopt different metrics). User's requests are proportionally distributed to the likelihoods that each replica will provide acceptable performance. Some systems, first divide [99] replicas (based on estimated performance against a defined set of criteria) into two groups, available and unavailable, and then, using any one of the selection algorithms. Table 2.1 shows a summary of the replica selection strategies, based on the algorithms used for the selection process.

Table 2.1: A Summary of the Algorithms Used for Replica Selection

No.	Category	Algorithm Name	Description
1	Heuristic	Round Robin	Rotates client request (transaction) in turn among replicas in the grid.
2	Heuristic	Random	Selects the best replica location randomly
3	Heuristic	Proportional	Client requests are proportionally distributed among replicas to relative power ratings, assigned by the system administrator.
4	Selection Criteria	Greedy	The most excellent replica location that exposes the best criteria values is chosen as a local optimum optimization, which assumes to lead to the global optimum.
5	Selection Criteria	Weighted	Assigns each request to the replica estimated to give best performance against the selection criteria.

The next sub-section discusses various related works with replica selection based on criteria set. The main focus is how to define the criteria set in order to gain best selection.

2.4.2.1 Replica Selection Based on weighted algorithm

Many studies have located the best replica that has minimal turnaround time. However, the major distinction among these studies relies on estimating the turnaround time criterion. It is not possible to compute turnaround in advance [99], but some factors play a significant role in estimating the turnaround time and are classified into two groups:

a) Utilizing Static Metric Criteria

Users select the closest server that holds the needed replica through the first replica selection techniques [100] based on few static metric factors such as: geographical distance, topological distance in number of hops, and HTTP request suspension. These techniques monitor the former turnaround time experienced by the client and use this information for later forecast. Authors in [101] have used probing messages received by the clients from servers to identify the accessible resources, and then the client uses the probing messages for selecting the minimal turnaround time server. Nevertheless, the static metrics are not adequate forecasters for the estimated turnaround time for user requests due to ignoring the dynamic network conditions such as network bandwidth that changes over time. This has been pragmatically confirmed by some studies [102, 103] which prove that, the static metrics are not appropriate forecasters for the projected response time of client requests. For example authors of [104] have used traditional replica model-based catalog in which each new request, RLS is inquired to obtain the replica sites addresses and then the link of network is investigated with hop count method to choose the best replica.

b) Utilizing Dynamic Metric Criteria

The emergence of dynamic replica selection techniques [54, 85, 99] are aimed at enhancing the evaluation of the estimated user turnaround time depending on the

measures of network parameters such as: network bandwidth and server request latency. The ‘best’ replica is chosen by the smart forecast, based on the chronological historical log files. Here, the term “best” means that the replica has the least turnaround time. The resource capabilities and network status are monitored by these techniques based on or using other grid services such as the Network Weather Service (NWS) [105]. NWS carries out end-to-end network investigations (where the available network performance is measured) and then employs fast statistical models to investigate the history information to predict current performance.

In the context of replica estimation based on turnaround time, authors of [54] have considered the network bandwidth as a dynamic mechanism for choosing the suitable replica at run time. This work has adapted the bandwidth as a significant dynamically changing factor to decide the best replica location. However, this selection of replica supports only the Storage Resource Broker (SRB) datasets and not other datasets such as RLS in Globus. On the other hand, the GridFTP log file is the only prediction tool utilized by the researchers of [85] for the purpose of identifying the replica in minimum turnaround time. While authors of [106] have explained why only GridFTP is not adequate for the forecast? Therefore, a regression technique model is developed for estimating the data transfer time from the resource, to the sink, depending on three data sources such as GridFTP, NWS, I/O Disk.

On the other hand, authors of [44, 106, 107] have come up with many research outcomes. They have projected the best site holding the requested replica by using the history of previous replicas transfer information. While the dataset moves between two sites, the dataset size, the available network bandwidth, and transmission time are recorded. This record is to be utilized later in training and testing the regression model to estimate the real transmission time. The authors have illustrated that data from different sources can facilitate superior estimations better than the data from a single source. They have achieved a higher precision in predicting the file transfer throughput by utilizing data from data streams of network, dataset size, and past grid transfer information.

Consequently, a novel approach by authors of [48] has exploited a replica selection technique with the K-Nearest Neighbor (KNN) rule used to choose the best

replica based on the locally collected information. The best replica location for a dataset is selected by the KNN rule based on the preceding dataset transfer logs that indicates the history of the dataset and the similar ones. However, based on the authors' opinion, this technique has a shortcoming represented by the possibility of misclassifications in case of the large dataset transfer. This in turn will cost more than a pair of small dataset transfer misclassifications. It is noteworthy that the accuracy is very minimal in the Gaussian random access pattern. Furthermore, the need of recording all previous instances (datasets requests) for selecting the best replica site is another shortcoming of KNN. This obligates the KNN not only to consume some extra time searching the huge volume of database history but also the outcome may not be accurate as well.

In addition to that authors of [46] have proposed a Neural Network predictive technique (NN based) to approximate the transfer time among sites. The transfer time that has been forecasted can be employed as an approximate to choose the best replica site from various sites. The outcomes of simulation have demonstrated that the Neural Network predictive technique works more precisely in comparison to the multi regression model which was used before NN [44, 106, 107]. Nevertheless, NN technique does not always provide the accurate decisions because the copy of the dataset or the replica might no longer be available (this is a common occurrence in grid) in the predicted site, therefore, in this case the traditional model has to be used.

Some researchers have utilized fuzzy logic techniques. For example researchers of [108] have conceived a fuzzy logic technique to evaluate the replication "state" (i.e., negative, normal and/or positive) by using the gray prediction model to analyze the factors that affect replica selection. Their approach has archived promising results in estimating response time.

Authors of [109] have proposed new replica selection, based on ant colony optimization to improve average access time. In this algorithm the ants convey replica information and find the best replica with the shortest access time. The ants define the best replica by calculating the pheromone for each replica. The pheromone indicates the supremacy of the replica. When a file is requested by a task or a user, an ant is produced, invoked and initialized by the places of replicas based on the catalog that

contains the PFNs of all replicas. While the ant moves from one node to another in the network, the statistical information of the visited nodes is collected. Furthermore, this information is carried from one node to the other. This eventually makes the statistical information to be dropped a trail in each node. These trails will be sensed and used by other to determine the best path to reach the better pheromone. Here, the ants are the probes sent by the requester, and the information conveyed by the ants is the pheromone of each node. Ants build up a table Node Name (PFN) Pheromone which contains the nodes that the ants have visited on their way and the pheromone of each node. The ants use this table to choose the best replica. As the number of request for replicas gets augmented, the ant will consume huge bandwidth in the network. So the moving ants should be as much as straightforward and small-sized because whenever the ant reaches the best replica, it can be downloaded from the source whenever required.

Some researchers have combined static and dynamic approaches, for example a recent work [110] allows the user to prioritize among three network factors such as: bandwidth (B), distance (D), and history record (H). The system is named PU-DG Optimizer toolbox (also recognized as PU-DG Optibox) and it is a package containing some efficient techniques and algorithms. The algorithms operate as middleware on the top of data grid platforms to optimize file downloads by improving its effectiveness and performance. The users have totally six different options: BDH, BHD, DBH DHB, HBD, and HDB, from which they can choose one. The toolbox utilizes mathematical formulations for calculating download time. It is transformed into a dynamic programming problem in order to reduce the final time complexity to $O(n)$, where n is the number of candidate replica sites. The toolbox also provides manual and automatic download modes for both computing novice and nerds. It is anticipated that such an approach could decrease the problems that most users could possibly face, in operating and managing files in a data grid environment.

Another example is by authors of [45] as they have proposed a Single Trip Time (STT) approach. In STT approach, the time is taken by the small packet to travel from Replica's Site (RS) to Computing Site (CS). The delays in STT comprises hold-off in packet-transmission delay (the rate of transmission from each router and the replica site), delay in packet-propagation (the proliferation of each and every link), delays in

queuing packets in intermediate routers and switches, and delays in processing packet (the delay in processing at each router and at the replica site) for a single transmission, beginning from replica site to the computing site. This indicates that STT is the summary of all sorts of delays. They use the standard deviation of STTs as a factor to check the stability or instability of the network links [111]. The computing site obtains recurrent STTs of all the sites of replicas, prior the selection process, and saves the latest in a log file known as Network History File (NHF), they have employed the information from the network monitoring service like Iperf Service [112] and the storage element service about the respective data access latencies. SEs [118] are abstractions for any kind of storage system (e.g., disk pool or a mass storage system). Each SE offers an exact volume of storage for grid users to save their jobs and data.

Some approaches have integrated storage access latency in estimating the turnaround time. For instance, the authors of [44, 48] have included the storage access latency in estimating the turnaround time. The history of latency storage information about and data transfer time is considered as a forecaster of future time. However, it is not possible to accurately predict the future storage access latency due to periodical changes and upgrading of the grid resources such as storage. It is noteworthy that the best replica chosen from a particular storage will not remain as best forever as there are chances of changes to occur in the storage media. However, in general the techniques that depend on the historical information about the resources can be more proper and pertinent in a stable grid environment. Indeed, previous studies have not considered storage request queue and storage media speed as parameters that influence the turnaround time.

The above mentioned dynamic replica selection approaches have considered only the network bandwidth, static metrics, storage access latency and historical information, in order to estimate the turnaround time. So the main focus was only on one QoS parameter, which is turnaround time.

On the other hand, authors of [113] have proposed a replica selection that integrates three QoS parameters namely: time, security and reliability in which a multi criteria decision maker is used to select the best replica. The main objective of this

system is to achieve fair resource allocation in terms of QoS amongst users to accomplish users' satisfactions. This approach has been implemented based on a multiple criteria decision maker known as Analytical Hierarchy Process (AHP) [114] which relies totally on the historical average of the QoS attained by users. For example, if history shows that security is the least QoS given to a user then this means that the best replica for that user is the one that shows higher security. The main drawbacks of the system are: first, it targets a uniform fairness based on historical information and it does not consider the users' preferences in an attempt to satisfy the users and aims to deliver the same portion of QoS to all users regardless of what they desire or what they pay for. Second, it considers the average for the user regardless to the number of requests he has already made (the user already requested 100 replica is treated similarly to the one who requested 1000 replicas). Third, it carries out the selection one by one which also may reach a better resource allocation in terms of QoS to front-queue users comparing to the users at the back. Forth, it does not address site availability and it has been built under the assumption that users are competing for the limited data resource. Moreover approach researchers have defined security as a level to protect the data files (already secured by the grid toolkits) from unauthorized users. In fact, relying on history may hinder the system from selecting the best site because the decision maker aims at balancing the QoS rather than increasing all of them.

However, there are other QoS in addition to what has been mentioned above, which are as important as turnaround time and in some cases they are more important. Neglecting such QoS parameters could infect the turnaround time or could lead to serious issues such as faults or security breaches and expensive costs. Aspects of site availability come from the fact that each site has its local policies that allows access at certain time and for certain duration. Ignoring this fact could lead to delays or faults. On the other hand, aspects of cost come from the fact that the same dataset may have location-dependent access costs, just like any tangible goods in actual economies. Not paying attention to this fact may increases replicas' cost which in turn dissatisfy the users. Aspects of security can be represented by: the existence of hackers, viruses, and many unauthorized users attempt to violate the network. In addition to that, some users may value cost only and they do not have time constraint or security concerns.

Therefore engaging users' preferences is critical and may lead to enhance the level of users' satisfactions.

Most of the previous works related to replica selection problem have focused only on turnaround time and security (implicitly include reliability). Therefore, it is believed that there are other QoS significant for the users that can be integrated in the replica selection to improve the whole grid environment. However, there are some shortcomings in the recent replica management systems as it has been referred to in the literature review.

It becomes essential to enhance the current systems to achieve a better performance. Hence, an improved replica management system that offers better solutions and evades the existing systems weaknesses is proposed to the underlying research problem.

2.5 An Overview of Simulation Tools and Evaluation

A broad exploration of simulation tools that have been designed to cater the distributed systems and grid sorts has been carried out. The proper simulator should contain the following [115]:

- The simulator must be steady and contain the needed services for the required problem and include the most important services as well, that are delivered by the simulation tool supplier, in terms of technical support, maintenance, upgraded, and available with acceptable price.
- The simulator must be compliant to the reality as much as possible.
- The simulator should be in line with grid features in all replication management characteristics that have a number of entities that represents the grid environment like: bandwidths and network topology, replica selection and replication policies or strategies, resource broker, and data transfer mechanisms.

- The simulator must be user-friendly, easy to be utilized and understood.
- The simulator must be a discrete event driven simulator that uses a discrete event system, like distributed and parallel systems.

As shown in Table 2.2 and in Figure 2.4, a lot of computer simulators are available, however only: SimGrid, OptorSim, ChicSim, Bricks, MicroGrid, GridSim, and Monarc, support grid features, generally job scheduling. OptorSim is not shown in the table because it is used for replication and replica selection.

Table 2.2: Features of the Simulators

Design	SimOS	SimJava	NS-2	Parsec
Simulated Systems	Parallel systems	Distributed systems, networks	Wired and wireless networks	Wireless networks, parallel
Simulation	Static, discrete, deterministic	Static, discrete, deterministic	Static, discrete, deterministic	Static, discrete, deterministic
Simulation engine	Parallel, event-driven DES	Serial, event-driven DES	Serial, event-driven DES	Serial & parallel, event-driven DES
Programming framework	Structured	Object-oriented	Object-oriented	Structured
Design environment	Language	Library	Language	Language, library
User interface	Non-visual	Animation, graph	Animation	Drag-drop, form
Design	Bricks	GridSim	MicroGrid	SimGrid
Simulated Systems	Grid, resource scheduling systems	Grid, resource scheduling systems	Grid, resource scheduling systems	Grid, resource scheduling systems
Simulation	Static, discrete, deterministic	Static, discrete, deterministic	Dynamic, continuous, deterministic	Static, discrete, deterministic
Simulation engine	Serial, event-driven DES	Multithreaded, event-driven DES	Parallel, event-driven DES	Serial, trace-driven DES
Programming framework	Object-oriented	Object-oriented	Structured	Structured
Design environment	Language	Library	Language	Library
User interface	Non-visual	Form	Non-visual	Non-visual

Category	Tool	Organization	Key similarities and differences, simulated systems, and Web site
Parallel systems	SimOS	Stanford University, U.S.A.	<ul style="list-style-type: none"> • Models complete computer systems through fast simulation of hardware and levels of abstraction. • Simulates a complete multiprocessor system and studies all various aspects including hardware architecture, operating system and application programs. • http://simos.stanford.edu/
Distributed systems	SimJava	University of Edinburgh, U.K.	<ul style="list-style-type: none"> • Provides a core set of foundation classes for simulating discrete events. • Simulates distributed hardware systems, communication protocols and computer architectures. • http://www.dcs.ed.ac.uk/home/simjava/
Networks	NS-2	University of California at Berkeley, U.S.A.	<ul style="list-style-type: none"> • Supports several levels of abstraction to simulate a wide range of network protocols via numerous simulation interfaces, such as using scripting language and/or system language. • Simulates network protocols over wired and wireless networks. • http://www.isi.edu/nsnam/ns/
	Parsec	University of California at Los Angeles, U.S.A.	<ul style="list-style-type: none"> • Uses a portable runtime kernel that executes simulations on either sequential or parallel architectures enhanced by ready support of numerous parallel simulation protocols. • Simulates very large scale integrated (VLSI) parallel architectures, parallel databases and wireless networks using parallel simulation. • http://pcl.cs.ucla.edu/projects/parsec/
Mobile systems	GloMoSim	University of California at Los Angeles, U.S.A.	<ul style="list-style-type: none"> • Provides an extensible and modular library that supports implementation of alternative protocols for each layer of the wireless communication protocol stack. • Simulates large-scale wireless mobile networks. • http://pcl.cs.ucla.edu/projects/gloimosim/
Grid scheduling systems	Bricks	Tokyo Institute of Technology, Japan	<ul style="list-style-type: none"> • Provides simulation for resource allocation strategies and policies for multiple clients and servers as in global computing systems in a Grid environment. • Simulates resource scheduling algorithms in Grids. • http://matsu-www.is.titech.ac.jp/~takefusa/bricks/
	GridSim	University of Melbourne, Australia	<ul style="list-style-type: none"> • Supports simulation of space-based and time-based, large-scale resources in the Grid environment. • Simulates economy-based resource scheduling systems in Grids. • http://www.gridbus.org/gridsim/
	MicroGrid	University of California at San Diego, U.S.A.	<ul style="list-style-type: none"> • Runs emulations by executing actual application code on the virtual Globus Grid and thus requires more time to complete the application. • Emulates the Globus Grid environment for resource management. • http://www.csag.ucsd.edu/projects/grid/
	SimGrid	University of California at San Diego, U.S.A.	<ul style="list-style-type: none"> • Simulates a single or multiple scheduling entities and time-shared systems operating in a Grid computing environment. • Simulates distributed Grid applications for resource scheduling. • http://grail.sdsc.edu/projects/simgrid/
Embedded systems	Ptolemy II	University of California at Berkeley, U.S.A.	<ul style="list-style-type: none"> • Builds upon a component-based design methodology that hierarchically integrates multiple models of computation to capture different design perspectives. • Simulates systems that comprise heterogeneous components and sub-components. • http://ptolemy.eecs.berkeley.edu/ptolemyII/

Figure 2.4: A Wide List of Simulators from [116]

The simulators in the above mentioned table have been studied and the one that supports grid features are summarized in Table 2.3, based on the current research requirements as explained above.

Table 2.3: Listing of Functionalities and Features for Grid Simulators

Features	GridSim	OptorSim	Monarc	ChicSim	SimGrid	MicroGrid
Tool stability	yes	yes	yes	yes	yes	yes
Data replication	good	best	good	good	N/A	N/A
Different replication strategies	good	Best	good	good	N/A	N/A
Scheduling user jobs	Yes	yes	yes	Yes	yes	yes
Network features	good	good	good	good	good	best
Based on real project	No	Yes	no	No	no	no
User-friendly	Yes	yes	no	No	yes	yes

The simulators: SimGrid and MicroGrid are general purpose simulators that focus on job scheduling and they do not support data grid environment such as replication management.

Monarc and ChicSim support replication management, but they are designed for assessing replications performance of systems in unusual circumstances, rather than in general grid circumstances. They are not considered extensible and user-friendly, and thus only very few researchers have used Monarc or ChicSim.

Nevertheless, the simulators: GridSim, and OptorSim fit to this research requirements. They are stable, with the support from the grid features and replication management. In addition to this, they are applied using JAVA and support GUI, which is user-friendly. In addition, quiet a number of earlier studies have employed GridSim and OptorSim, for evaluating purposes. Hence, GridSim and OptorSim have been thoroughly investigated, as along with their current approaches and algorithms. GridSim focuses more on job scheduling algorithms and does not focus on the data replication issues [48, 117], meanwhile OptorSim focuses on both job scheduling and data replication strategies, with a lot of optimization algorithms [118]. Additionally,

OptorSim is designed constructed based on the European Data Grid project (EU datagrid) [33, 119] and consequently, the model very much realistic. As a result, this study has used OptorSim, to evaluate the performance of the proposed system.

OptorSim [117, 119] is a grid simulator implemented in JAVA, and developed to assess the functionality of various job scheduling and replica optimization strategies. OptorSim is a well-known simulator, which is widely employed in a lot of studies [37, 118, 120]. A number of elements exist in OptorSim, based on EU data grid [119, 121]. These elements include: CEs to which the job is sent, SEs where data is kept, the network elements which are used for connecting grid sites. The Resource Broker (RB) which is responsible for submitting jobs to grid sites, according to some scheduling algorithms. The Replication Manager (RM) plays a role in replication optimization strategies.

Much works have been done for building the basic grid infrastructure; Globus [49, 97, 122], Condor [123], and recently the EU data grid [33, 119, 124]. These works have facilitated the core grid middleware services, existing for further development of the grid applications. Therefore, OptorSim is much closer to reality, since it is based on the EU data grid architecture [33, 119, 124] as shown in Figure 2.5. The EU data grid facilitates the construction of data grids not only for High Energy Physics (HEP), but also for Observation science and Biological science. HEP conduct their experiments at CERN, and the 5000 scientists around the world have collaborated in the four experiments: ATLAS, LHCb, CMS and ALICE. More details about OptorSim are presented in chapter 3.

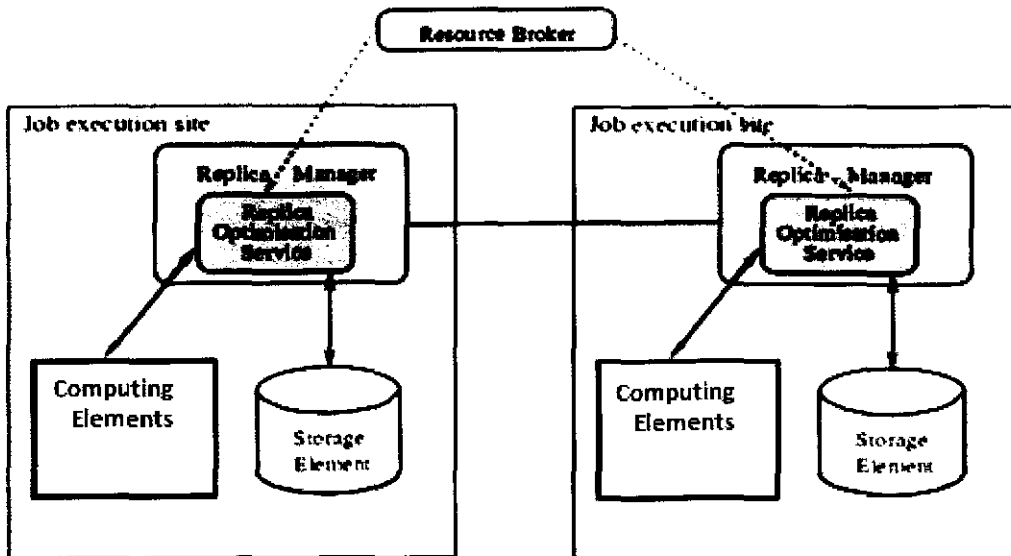


Figure 2.5: The EU Data Grid Architecture

K-means algorithm is one of the popular clustering tools used in scientific and industrial applications. The name instigates from representing K clusters S_j by mean (i.e. weighted average) S_i of its points, called the centroid (i.e. the center of the cluster). The sum of Euclidean distances $d(x,y)$ between a point (X_i) and its centroid (S_j) is used as an objective function as shown in equation 2.1.

$$E = \sum_{i=1}^k \sum_{x_j \in S_i} (x_j - s_i)^2 \quad 2.1$$

Generally the K-means Clustering Algorithm selects a random preliminary partition or centers and continually recalculates the centers depending on the partition and then re-computes the partition, based on the centers. The popularity of the K-means algorithm lies in its simplicity and user friendliness. Additionally, the K-means algorithm works with any standard norms and it is insensitive to data ordering. However, there are some shortcomings in the K-means algorithm. For example, the result strongly depends on the initial selection of the centroids. In addition, the accurate number of clusters is not apparent and it produces unstable clusters. The basic K-means algorithm by [125] is outlined below:

- Preliminary centroids are selected based on K-data points.
- Reallocate all points to their adjacent centroids.

- The centroid of each newly formed cluster is recalculated.
- Step 2 and 3 are repeated until the centroids are not altered.

When representing data with few clusters, the fine details of the data are lost, however, the representation will be simpler.

2.7 Genetic Algorithms

Darwin's theory of evolution has motivated John Holland [126] to formulate Genetic Algorithms (GA). GA is a very popular search algorithm, which is based on the technicalities of natural selection and natural genetics [127], or as software and procedures modeled after genetics and evolution [128]. Generally, most of the genetic algorithms have few common features like: (a) populations of chromosomes as shown in Figure 2.6, (b) selecting based on fitness, (c) crossover to generate offspring, and (d) random mutation of a new offspring [129].

Chromosome 1:	1 0 1 1 0 0 1 0 1 1 0 0 1 0 1 0 1 1 1 0 0 1 0 1
Chromosome 2:	1 1 1 1 1 1 1 0 0 0 0 0 1 1 0 0 0 1 0 1 1 1 1 1
Chromosome 3:	1 0 0 1 0 0 1 0 0 1 0 0 1 0 1 0 1 1 1 0 0 1 0 1
Chromosome 4:	0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 1 0 0 0 1 1 1 1 1
⋮	
Chromosome n:	1 0 0 0 0 0 0 0 1 1 0 0 1 0 1 0 1 1 1 0 0 1 0 1

Figure 2.6: Populations of Chromosomes in GA.

Additionally, the algorithm begins with a population of "individuals", which signify a potential resolution, for a particular issue. Each and every probable solution in the population of a natural entity is coded in a professed chromosome (sequences of genes). All the chromosomes are allotted "fitness" depending on the suitability of a solution, based on the issue. The solutions (individuals) are chosen for the process based on their robustness, particularly, those that follow the survival of the fittest theory by Charles Darwin. These solutions (individuals) are used for reproducing by "cross breeding", with other individuals in the population, and employed to build new

individuals as offspring, with a hope that the offspring are best fits than the old individuals, and a generation is complete [126]. This process is iterated until specific criteria are met. The Figure 2.7 illustrates the basic steps for GAs [127] in which, t represents the generation number, and P stands for population. The first population is initialized by coding it into a specific type of representation (i.e. binary, decimal, float, etc), then it is assigned to the pool. Fitness is computed in the evaluation step. While, the termination condition is not met, which might be number of generations or a specific fitness threshold, the processes of selection, recombination, mutations and fitness calculations are done.

```
t = 0;  
Initialize P(t);  
Evaluate P(t);  
While not (termination condition or convergence)  
Begin  
T = t+1;  
Select P(t) from P(t-1);  
Recombination pairs in P(t);  
Mutate P(t);  
Evaluate P(t);  
End
```

Figure 2.7 Pseudo-code of Genetic Algorithm

Individuals from population for the process of crossover are chosen by the selection process. Recombination (or crossover) is executed by swapping a part (or some parts) among the selected individuals, which depends on the type of crossover (Single point, Two points, Uniform, etc), see Figure 2.8.

Parent 1	1 1 0 1 1 0 0 1 0 0 1 1 0 1 1 0
Parent 2	1 1 0 1 1 1 1 0 0 0 0 1 1 1 1 0
Offspring 1	1 1 0 1 1 1 1 0 0 0 0 1 1 1 1 0
Offspring 2	1 1 0 1 1 0 0 1 0 0 1 1 0 1 1 0

Figure 2.8: Single Point Crossover

After that, mutation is performed by substituting few points among arbitrarily selected individuals as shown in Figure 2.9. Later the fitness has to be recomputed to be the base for the subsequent cycle. Prior to a GA execution, a suitable encoding (or representation) for the problem must be devised. The coding is a population of strings, each of which represents a solution to the problem.

Original offspring 1	1 1 0 1 1 0 0 1 0 0 1 1 0 1 1 0
Original offspring 2	1 1 0 1 1 1 1 0 0 0 0 1 1 1 1 0
Mutated offspring 1	1 1 0 1 1 0 0 1 0 0 1 1 0 1 1 0
Mutated offspring 2	1 1 0 1 1 0 1 1 0 0 1 1 0 1 0 0

Figure 2.9: Mutation in GA

GAs, facilitate a variety impending solutions, known as a population, which comprises encoding of the concurrently initiated parameter. In GA, the preliminary step is coding in which the real problem is translated into biological terms, and appropriately depicts it for GAs. Encoding is the format of a chromosome. The size of population is indicated by the number of chromosomes in the population, where the best population size will be based on both, the application and the length of the chromosome. Longer chromosomes enable larger population sizes and increase diversity for the initial population. This would result in better exploration of the search space at the expense of requiring more fitness evaluations.

GAs, will slow down in case if there are a lot of chromosomes. Nevertheless, GAs, have less probability to perform the crossover operation if there are less number of chromosomes, and only a small part of search space is explored. If the population

loses variety, it is said to have a premature convergence and little exploration is being carried out [130].

Next step is crossover; recombination or crossover is done independently, irrespective of the problem of encoding or the fitness scores. It takes two individuals and cuts their chromosome strings at some specific position, to produce two “head” and “tail” segments apiece. The tail segments are then swapped over to produce two new full length chromosomes. Each of the two offspring inherits some genes from each parent. Crossover cogitates that new chromosomes will carry the high-quality constituents of the old chromosomes. Consequently, the novel chromosomes are believed to be superior. If crossover is performed, the genes between the parents are swapped, and the offspring is made from chromosomes parts of both parents. However, if there is no crossover, the offspring is an accurate duplicate of its parents. The most widespread recombination is the uniform crossover method. In this method, a crossover point is selected along the chromosome, and the genes up to that point are swapped between the two parents.

Mutation is independently implemented to each child, after the crossover that modifies each gene with a low prospect, characteristically in the range 0.001 and 0.01, and alters elements in the chromosomes [131]. Frequently the mutation assures that the prospect of seeking any given string will never be zero. It acts as a safety net to recuperate the high-quality genetic material which might be lost in the selection process and crossover. Mutation avoids the GA from falling into local extremes and offers a small amount of random search that makes sure that no point in the search space has a zero possibility at investigation. While performing the mutation, one or more parts of a chromosome are altered, and if there is no mutation, the offspring is instantly produced after the crossover (or directly copied) without any change [131]. It is necessary to judge the quality of that solution in relation to other solutions in the search population. This is referred to as measuring the fitness of the solution.

The most crucial aspect of GAs is the fitness function, which returns a single numerical “fitness” that is believed to be the relative capability of the individual, which is represented by the chromosome. Preferably, fitness function is required to be soft and accepted so that chromosomes with a rational fitness can be close to the

chromosomes with somewhat enhanced fitness [132]. The general rule in building a fitness function is that it should pragmatically replicate the value of the chromosome. If the fitness function is greatly slow or multifarious to be evaluated, then sometimes an approximate function evaluation can be employed. If a much faster function can be devised (which, approximately gives the value of the true fitness function), the GA may find a better chromosome in a given amount of CPU time, than when using the true fitness function. At the beginning of a run, the values of each gene for different members of the population are randomly distributed. Consequently, there is a wide spread of individual fitness. As the run progresses, particular values for each gene begin to predominate. As the population converges, the range of fitness in the population is reduced. This variation in fitness range throughout a run, often leads to the problems of premature convergence and slow finishing. Premature convergence is a classical problem with GAs, where, the genes from a few comparatively highly fit (but not optimal) individual may rapidly come to dominate the population, causing its convergence on a local maximum, and thus the ability of the GA to continue to search for better solutions is effectively eliminated.

Mutation may be a factor that helps to explore new offspring, but it is less effective as it makes the process of exploration slower. Another problem is slow finishing, when termination is a matter of fitness and not the number of generations, and in a certain point of time, where fitness for all individuals are almost the same, the average fitness moves slowly to the maxima [132]. A common practice is to terminate the GA, after a pre-specified number of generations, and then the quality of the best members is tested from the population against the problem definition. If no acceptable solutions are found, the GA may be restarted, or a fresh search will be initiated [126]. The general scheme of GA is shown in Figure 2.10.

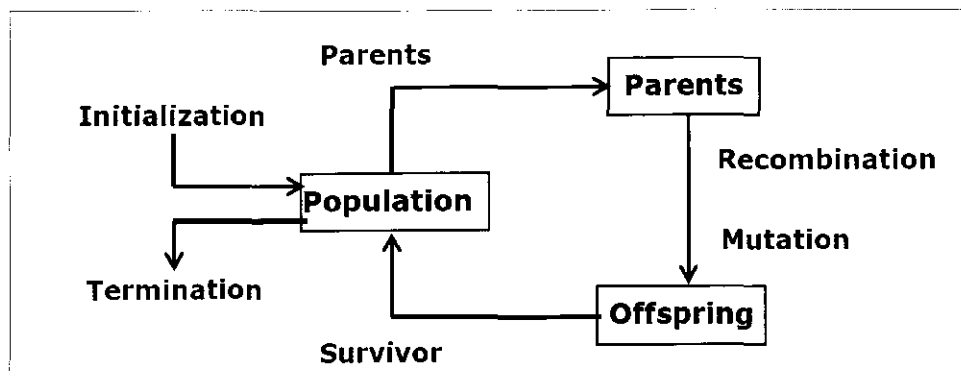


Figure 2.10: The General Scheme of GA

2.8 Multi-objective Optimization Model

Lots of real world designs or decision makings involve in simultaneous optimization of multiple objectives. In principle, multi-objective optimization is more different than the single objective optimization. In single objective, one attempts to obtain the best design or decision. That is usually the global minimum or maximum. Whether it is a minimization or maximization depends on the nature of the optimization problem. In the case of multiple objectives, there might not be a single solution that can be considered as the best solution with respect to all objectives. In typical multi-objective optimization problem, there exists a set of solutions which are superior to the rest of solutions in the search space, when all objectives are considered, but are inferior to others solutions in the space in one or more objectives. These solutions are known as Pareto-optimal solutions or non-dominated solutions [133, 134]. The rest of solutions are considered as dominated solutions. Any one of these solutions is an acceptable as long as none of the solutions in the non-dominated set is absolutely better than any other. The choice of one solution over the others requires problem knowledge and a number of problem related factors. Thus, one solution chosen by a designer may not be acceptable to another designer in a changed environment.

In fact, the concept of Pareto optimal solutions was proposed by authors of [135]. Later authors of [136] have formulated optimality conditions for multi-objective optimization. Since then, many methods in multi-objective optimization have been

proposed (see e.g. [137-141]). Although a multi-objective optimization problem usually has many Pareto optimal solutions, typically only one solution is desirable for implementation. A human Decision Maker (DM), an expert in the domain of a multi-objective optimization problem provides the necessary information to select the most preferred solution based on her/his preferences. The methodologies in multi-objective optimization revolve around the type of support provided to the DM, to choose the most preferred solution. Multi-objective optimization has been studied for over a century. In 1881, researchers of [142] have proposed a scalarization technique, called an utility function for multi-objective optimization.

One way to solve multi-objectives problems is to scalarize the vector of objectives into one objective by averaging the objectives with a weight vector as a process to allow simpler optimization. It is always advisable to use a scalarizing function that is proven to generate Pareto optimal solutions.

Additionally, in Multiple Criteria Decision Making (MCDM), decision support is typically seen as the main goal. The preferences of the DM are often included when formulating a scalarized problem, which is subsequently solved using any suitable mathematical programming technique, to find a single Pareto optimal solution (at a time) satisfying DM's preferences and this process may be iterated. Different method classes can be identified in MCDM, depending on the role of the DM in the solution process [143].

The procedure of formulating a multi-objective optimization problem is as follows:

$$\text{minimize or maximize } \{ f_1(x), f_2(x), \dots, \dots, f_k(x) \} \quad (2.2)$$

$$\text{Subject to } x \in S \subset \mathbf{R}^n,$$

With $k \geq 2$ conflicting objective functions $f_i: S \rightarrow \mathbf{R}$. If the objective function f_i is to be maximized then we minimize the function $-f_i$, which is equivalent to maximizing f_i . We denote the vector of objective function values by $(f_1(x), f_2(x), \dots, \dots, f_k(x))^T$ to be called an objective vector. The decision vectors $x = (x_1, x_2, \dots, \dots, x_n)^T$ belong to the decision space S . For example, we may have $S = \{ x \in \mathbf{R}^n: g_i(x) \leq 0, h_i(x), \mathbf{x}^{LO} \leq X \leq \mathbf{x}^{UP} \}$, where $g_i: \mathbf{R}^n \rightarrow \mathbf{R}$, $i = 1, \dots, \dots, n$, are the

functions of inequality constraints, $h_j: \mathbb{R}^n \rightarrow \mathbb{R}$, $j = 1, \dots, m$ are the functions of equality constraints and $x^{lo}, x^{up} \in \mathbb{R}^n$ are the lower and upper bounds of the decision variables, respectively. The objective function values of all decision vectors belonging to S belong to a k -dimensional space called objective space ($f(S)$). In general, multi-objective optimization problem have many optimal solutions with different trade-offs. These optimal solutions are called Pareto optimal solutions.

2.8.1 Scalarization in Multi-objective Optimization

Further to what has been discussed in the above subsection, multiple criteria decision making or multi-objective optimization problem, is one of the research fields where several methods exist, to aid the DM to find a Pareto optimal solution that satisfies the user's preferences. Usually in MCDM, multiple objectives are converted into a single objective problem. Several scalarization techniques can be found in the literature, but the most commonly used are:

Weighted sum method: In the weighted sum method, a weighted sum of each of the objective functions is used as a scalarizing function, and a scalarized problem is formulated and solved to obtain a Pareto optimal solution.

The scalarized problem is defined as:

$$\text{Minimize } \sum_{i=1}^k W_i f_i(x) \quad (2.3)$$

Subject to $x \in S$,

Where, $W_i > 0$, for all $i = 1, \dots, k$ and $\sum_{i=1}^k W_i = 1$. The weighted sum method is simple to use [141] is one its main advantages.

2.9 Proposed Solution

This study has proposed a new replica selection system termed as "Replica Selection in Data Grid (RsDGrid)", which consists of three components (systems) namely: A-system, D-system, and M-system. Each of these systems has its own scope

and specifications. RsDGrid is designed to switch between these systems, based on decision maker requirements. The first two systems, A-system and D-system represent the continuous improvement of the existing single user replica selection systems, by incorporating new features, while M-system goes in a new direction by focusing on simultaneous multiple users' selection.

The main feature of A-system is considering the site availability QoS parameter during the selection process. Site availability is a very important parameter to address the replica selection process because avoiding it could lead to faults or at least excessive delays in the turnaround time of jobs. Therefore, it is a continuation of the previous efforts in the literature, which had only one objective: decreasing jobs' turnaround time. Moreover, one of the main objectives of the A-system is to avoid faults.

D-system is also a single user approach that considers many different QoS parameters. The main distinction between A-system and D-system is that the later handles heterogeneous QoS parameters and uses performance metrics other than time. Therefore, the focus in D-system is not only on time, but rather on a mixture of QoS parameters, one of which is time. D-system always tries to locate the replica location that has the best ratings for all QoS parameters, and those ratings are almost equal to one another. On the other hand, D-system has an option that allows users to choose their preferences with respect to the QoS parameters. Each of the grid users could have his/her own preferences. For example, one user may prefer to select the replica location in a minimum turnaround time, regardless of the other criteria. Other users may prefer the security criterion more than the turnaround time. While, others may only be concerned with cost or targeting a balanced solution. Therefore, the second approach of D-system considers the user's preferences or priorities (if any), and integrates the preferences into the selection process. Moreover, the users' preferences can be exchanged among the users in a way that is similar to the stock market. Since the replica selection decision has conflicting criteria and is measured by heterogeneous values, the K-Means model was deployed to solve the heterogeneity of the criteria set in the replica selection system.

M-system is an upgraded version of D-system and other previously proposed systems by facilitating multiple users' selection, in order to achieve equal satisfactions amongst the grid users. In all previous replica selection systems, users' requests are fulfilled in a FIFO manner, one by one, regardless of the preferences of the remaining requests in the queue. This could satisfy the first users, at the expense of those who come after. Considering all the users' requests to gain fair satisfaction can obtain better results. However, it is a difficult task because it produces a huge search. Therefore, the use of the Genetic Algorithm (GA) is proposed in M-system, to overcome the huge searches involved in replica selection.

RsDGrid is an enhanced replica selection system that can be deployed into any grid middleware such as, Globus, Condor, and GridWay. The final objectives of RsDGrid are: reducing the turnaround time, increasing the QoS perceived by the users, increasing the fair satisfactions among grid users, and providing a flexible and dynamic system for both the grid users and the system administrators.

2.10 Summary

This chapter has presented and discussed the topics that are related to the proposed research work. The literature review has shown that there is a lack of studies that consider the site availability, the security, and the cost issues as important factors of the QoS, along with the turnaround time in the replica selection process. Previous study has yet to tackle the factor of site availability in the replica selection process. The K-Means algorithm is applied in the proposed solution to tackle the challenging points of the problem and to cater the heterogeneous values that represent site parameters. Moreover, this research will consider multiple users' replica selection in data grid environment to equally satisfy all users. The Multi-objective decision making approach is used in the proposed system to overcome the complexity of problem and the GA is employed to handle the wide search space. Chapter 3 will present the research methodology.

CHAPTER 3

METHODOLOGY

3.1 Overview

This chapter presents an overview of the methodology applied in this research. The methodology consists of three systems namely: A-system, D-system and M-system as shown Figure 3.1. Each system has its own specifications such as objectives, QoS parameters, performance metrics and so on. For each system, the objectives are identified then the QoS parameters are selected and modeled. The system is modeled as a multi-objective problem. M-system is also modeled as an optimization problem. After that, the algorithm of the replica selection system is designed. Next to that, the system is implemented in the simulator and compared with other systems. Finally, several experiments have been performed and the results are evaluated and analyzed in comparison with other systems. The details of A-system, D-system and M-system are explained in chapters 4, 5 and 6 respectively.

3.2 The Proposed Systems

Replica selection provides the mechanism to decide the best replica places for the grid users based on some criteria. To this end, a family of three efficient replica selection systems has been proposed and so-called (RsDGrid).

The problem presented in this thesis is how to select the best replicas within grid sites that achieve less jobs' times, higher QoS, and more and almost equal users' satisfactions (fair users' satisfactions). RsDGrid consists of three systems,

specifically, A, D, and M systems. Each of them has its own scope and specifications. RsDGrid is designed to switch amongst these systems according to the decision maker. The first two systems, A-system and D-system, represent the continuous improvement of the existing single user replica selection systems by incorporating new features, while M-system goes in a new direction by focusing on simultaneous multiple users' selection.

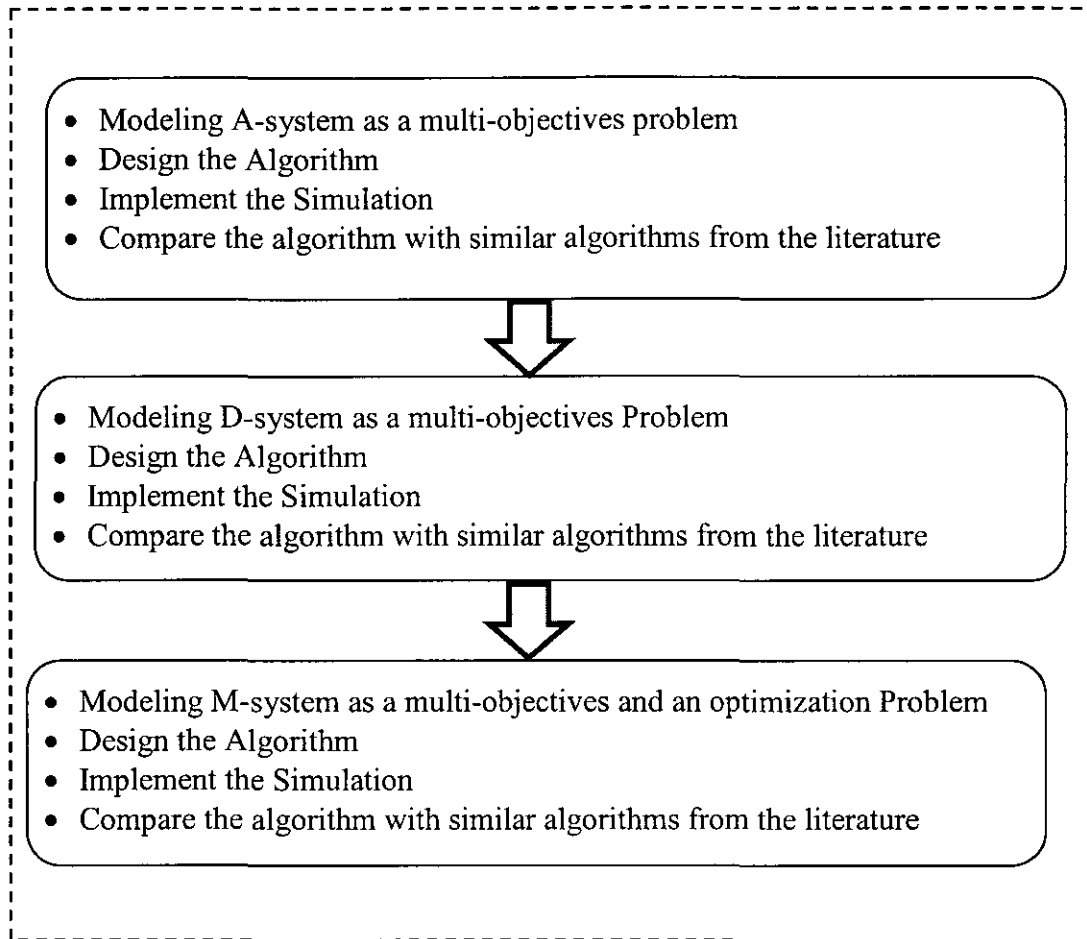


Figure 3.1: A Flowchart of the Research Methodology

3.2.1 A-system

A-system's is a single user based selection. This means that this system makes the decision for the users one by one according to the requests arrive from the scheduling queue. A-system's main feature is taking into account the site availability QoS parameter in the selection process. Site availability falls under sites' local policies. This parameter is a very important to be considered in the replica selection process

because the absence of this parameter could lead to faults or at least excessive delays in the jobs' turnaround time.

Based on its local policy, each site has its operating hours to serve other sites or users. However, accessing sites which are available for a time that is less than required will lead to timeout. This obliges to complete or to restart the task in another site if such mechanism is available. Sometimes, such problems occur and it is very difficult to know or to trace the causes. Availability in this research is defined as the capability of a given resource to fulfill a given task until it is completed.

In a continuous effort to what has been referred to in the literature, site availability parameter has one objective and that is decreasing jobs' turnaround time. In A-system each site's turnaround time A_i is rated out of 100 and each site's availability is rated out of 100 as will and presented in the two dimension space as shown in Figure 3.2.

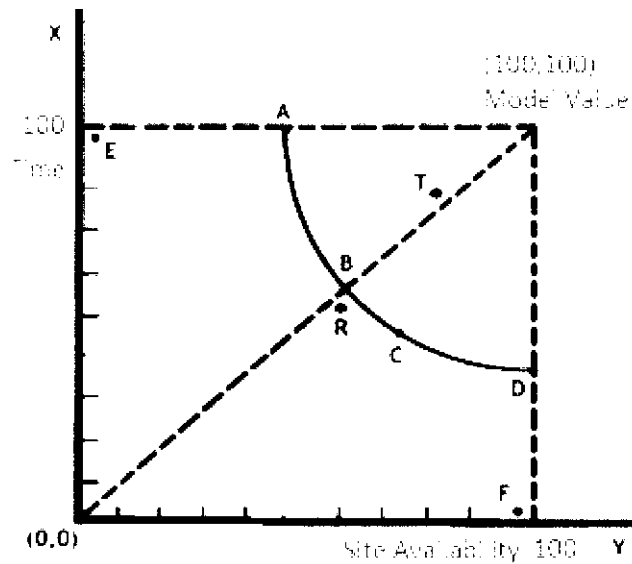


Figure 3.2: Visual Representations for Sites and their Model Values

The model value represents an imaginary site with a value of 100 for its parameters. The best site is the one that is the closest to the model value (T in Figure 3.2) and at the same time the closest to the diagonal because these two conditions ensure achieving the highest and almost equal rates for time and availability. For example if site T does not exist in Figure 3.2, site B will be selected (due to being the closest to the diagonal) even if sites A, B, C and D have the same distance from the

model value. If site B does not exist, site R is the best despite being the furthest in comparison to A, C and D because it presents more meet halfway (compromised) solution.

In general, the problem is a tradeoff between time and site availability therefore, A-system has been modeled as a multi-objective problem with two objectives: the first objective is to decrease the distance between the site and the model value. The second objective is to decrease the distance between the site and the diagonal.

The performance metric that has been used to evaluate the efficiency of this system is jobs average time. Several experiments have been carried out to check the scalability of A-system and to compare its performance in comparison with the OporSim Built-in algorithm (OsBi) and under different replication strategies (simulation is explained in the next section). Then the system has been analyzed using statistical methods. The full details of A-system are presented in Chapter 4.

3.2.2 D-system

D-system is also a single user approach but considers more QoS parameters comparing to A-system. D-system satisfies the same objectives of A-system and in addition to other objectives like: 1- to increase the number of QoS parameters utilized in the replica selection, 2- to increase the QoS received by the users, 3- to the make the QoS received by the users consistent with their preferences.

D-system is a two approach system. In general both approaches of D-system handle heterogeneous QoS parameters and use performance metrics other than time. Therefore, the focus in D-system is not only on time but rather on a mixture of QoS parameters. The first approach of D-system (D-SystemAp1) focuses on three QoS parameters, specifically time, availability and security as shown in Figure 3.3.

D-SystemAp1 always tries to locate the replica location that has the best ratings for all QoS parameters, and those ratings are almost equal to one another. The best site is the one that is the closest to the model value (B in Figure 3.3) and at the same

time the closest to the diagonal because these two conditions ensure achieving the highest and almost equal rates for time and availability.

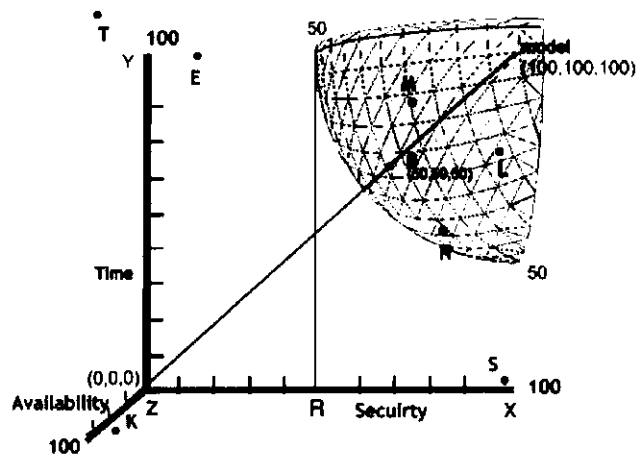


Figure 3.3: D-SystemAp1 Utilizing 3 Parameters

In general, the problem is a tradeoff between time, security and site availability and therefore, A-system has been modeled as a multi-objective problem with another two objectives: the first is decreasing the distance between the site and the model value and the second is decreasing the distance between the site and the diagonal. The metric utilized to measure the efficiency performance is TAS (an acronym from time, availability and security). The lowest value of TAS is the best value because it is the closest to the model value.

The second approach of D-system (D-SystemAp2) considers the user's preferences, if there is any, and integrates the preferences into the selection process. This approach allows increasing the number of QoS parameters. For example, it is possible to add cost parameter to them to be four parameters instead of three. D-SystemAp2 allows also decreasing the number of QoS parameters. For example, to be two parameters if the decision maker is interested to use only 2 parameters like time and security. Therefore, for TAS performance, metric has been replaced with UPQ which stands to user's preferred QoS parameter. Similar to TAS, the smallest value of UPQ is the best performance because it reflects the smallest difference between what the user prefers and what he/she gets. This in return means that D-SystemAp2 is a single objective problem.

Moreover, to achieve fairness amongst the users, each user is given a number of different points. Each number of points presents a QoS parameter in the data grid. Equal number of points is given to each user for the same QoS parameter. The number of points depends on the grid situation. Since the users have different preferences, they are allowed to exchange the points among themselves in a way that is similar to the stock market.

D-system has been implemented in OptorSim and several experiments have been conducted to check its scalability and to compare its performance in comparison with the OsBi and the random algorithm. Then the system has been analyzed using statistical methods. The full details of D-system are presented in Chapter 5.

3.2.3 M-system

M-system is an upgrade of D-system and other previously proposed replica selection systems by facilitating simultaneous multiple users' selection in order to achieve almost equal satisfactions amongst the grid users. M-system has the same objectives of D-system in addition to another objective and that is achieving fair users' satisfactions. In all previous replica selection systems, users' requests are fulfilled from the scheduling queue in a FIFO manner, one by one, regardless to the preferences of the remaining requests in the queue. This could satisfy the first users in the queue in comparison to those who come after. Considering the available resources and all the users' requests in the queue simultaneously could lead to a comprehensive view of their need and could lead to better resources allocation which almost fairly satisfying the users.

Generally, the M-system aims to achieve high QoS to the users and at the same time to achieve fair users' satisfaction. Fulfilling this aim will lead to production of a huge search space. Hence, it becomes a multi-objective optimization problem. Therefore, a hybrid approach between M-system and the Genetic Algorithm (GA) has been proposed to overcome the complexity of the problem. The performance metrics that have been used to evaluate the efficiency of this system are Fair Users Satisfaction (FUS) and Total UPQ.

Several experiments have been conducted to check the M-system scalability and to make comparisons among the M-system, AHP [113] and the D-system. Afterward, these systems have been analyzed using statistical methods. The full details of M-system are presented in Chapter 6.

3.3 Simulation

Simulation is a valuable tool in Data Grid researches and this research is no exclusion to this well-known rule. The advantages of simulation are the simplicity with which a variation of different strategies and algorithms can be tested and evaluated prior to implementing the best ones in real environments. Simulation also helps researchers to vary experiments' parameters such as the number of sites, their locations on the network, bandwidths, number of jobs, offered workload and the system scale. It provides a better understanding of the results of performance under a wider variety of conditions than what is in the real Grid environment with real implementations.

3.3.1 OptorSim Grid Simulator

OptorSim was developed to imitate the structure of a real Data Grid and study the effectiveness of replica optimization algorithms within such an environment. One of the main design considerations for OptorSim is to model the interactions of the individual Grid components of a running Data Grid as realistically as possible. Therefore, the simulation is based on the architecture of the EU Data Grid project [8] as presented in chapter 2. The grid topology as an input to OptorSim consists of 20 sites in the USA and Europe that were utilized during a data production form (CMS test bed) for major LHC experiments [119] as shown in Figure 3.6 and the other input simulates grid jobs and data file configurations. The European Organization for Nuclear Research (CERN) and Fermi National Accelerator Laboratory (FNAL) are producing the original files and storing them locally with a storage capacity of 100 GB each and other sites which have at least one CE and a storage capacity of 50 GB each. The order in which a job requests files is determined by the *Access Pattern*

used. Some different access patterns have been selected for the simulation. Such as sequential (all files are requested in a predetermined order), Gaussian random walk [119] (successive files are selected from a Gaussian distribution centered on the previous file) and Zipf. A Zipf-like distribution can be regarded as a special kind of exponential distribution allowing the simulation of several types of grid job. The developers of OptorSim have created the simulator to experiment with their own replica optimization algorithms [144].

The simulation has been created due to the base that the Grid contains many distributed sites in which some of them may provide data-repository and computing resources for jobs under processing. Computing resources consist of Computing Elements (CEs) that execute jobs which utilize data replicas kept in the Storage Elements (SEs). In the Grid, there is also a one Resource Broker which manages scheduling jobs to CEs. Any site does not have SEs or CEs acts as network nodes or routers. Replica Manager is responsible for the data movement among sites. The data movements are usually associated with jobs. Within the Replica Manager, the decision to create or delete replicas is controlled by the replica optimization algorithm built into the Replica Optimizer. In the simulation, each CE is represented by a thread. Job submission to the CEs is managed by another thread. The resource broker and the execution flow of these threads are shown in Figure 2. The Resource Broker ensures every CE is continuously running a job by frequently attempting to distribute jobs to all the Computing Elements.

When a file is requested by a job, the file name is used to locate the best replica via the Replica Optimizer function *getBestFile*. The function *getBestFile* checks the Replica Catalogue for copies of the file. The Replica Catalogue is a Grid middleware service currently implemented within the simulation as a table of file names and all corresponding physical file copies by examining the network available bandwidth between the destination storage element and all sites on which a replica of the file is stored, *getBestFile* chooses the physical file copy that will be accessed fastest and hence decrease the job running time.

The simulated version of *getBestFile* may cause replication to a Storage Element located in the site where the job is running. After the completion of any replication the physical copy of the best available replica is returned to the job. If replication does not

occur, the best replica is located on a remote site and is accessed by the job using remote I/O. Replica Optimization algorithms implement *getBestFile* so that it may copy the requested file from the remote site to a Storage Element on the same site as the requesting Computing Element. If all Storage Elements on this site are full, then a file must be deleted to ensure the success of the replication after gaining replication optimization algorithm approval.

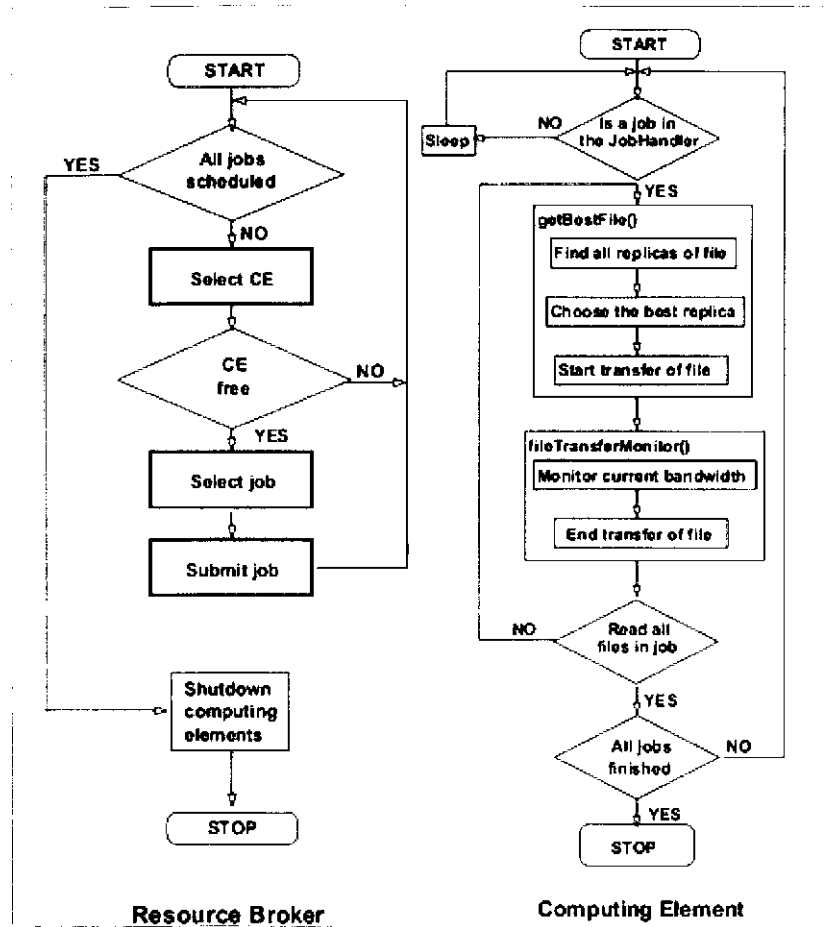


Figure 3.4: Execution flows of the RB and Computing Element Threads [119]

3.3.2 Simulator Modifications

The strategy used to decide whether to replicate or not and which file should be deleted if replication is conduct can differentiate optimization algorithms. These algorithms are the main focus of the original Optorsim team. A particular strategy could be selected in the parameters configuration file, as shown in Figure 3.5:

```

users = 1
#
# The choice of optimisers is:
# (1) SimpleOptimiser - no replication.
# (2) LruOptimiser - always replicates, deleting least recently
#     created file.
# (3) LfuOptimiser - always replicates, deleting least frequently
#     accessed file.
# (4) EcoModelOptimiser - replicates when eco-model says yes, deleting
#     least valuable file.
# (5) EcoModelOptimizer Zipf-like distribution
optimiser = 2

```

Figure 3.5: Choices of Optimizer in OptorSim Configuration File

Whatever the choices are, the algorithm implements *getBestFile* method in the *ReplicatingOptimiser* Class. This class provides an implementation of *getBestFile* method, which will attempt to perform replication of files to the close Storage Element of the Computing Element that require these files (if the files are not already available there). If there is a space on the close SE, the success of replication is inevitable. If there is no space, the *chooseFileToDelete* method of the subclass is called to determine which files should be removed to create a space.

OptorSim provides a standard implementation of *getBestFile*. It simply looks at the replica catalogue and the current network bandwidth that will take the shortest time to be accessed. It calculates the expected time cost using the *getCost* method that is defined in the *NetworkCost* object. The later object is created via *getNetworkCost* method (in the *NetworkClient* class), given the source and destination sites, in addition to the file transfer size.

In this thesis, the modifications that have been carried out are in the implementation of the *getBestFile* method where different versions of the *getBestFile* have been created. The modified copy of OptorSim allows calling one of the modified versions of *getBestFile* prior to a simulation run. One of the versions is the standard implementation given in the original Optorsim copy, which exclusively depends on the network latency cost. Network latency cost in turn depends on the file size and the available network bandwidth between the sender and the receiver (a new version [50] depends on the file size, machine speed, available network bandwidth and waiting

time for requests the in queue). This original version is kept to act as a benchmark against which the proposed system is compared.

OptorSim does not consider the site availability; therefore, a new model that considers site availability has been integrated into the simulator. The site availability model assigns service hours (availability) to each site ranged from 1 second to 24 hours (this assignment is realized in the *GridSite* class, representing site objects, via appropriate private field, and public setter and getter methods). Thereafter, if the simulator faces a selected replica from a site with insufficient availability time, it will then increase the replica transfer time based on the expected delay. The replica transfer time is obtained via *getCost* method, as mentioned earlier, and the ratio between the availability value and the file transfer time should be at least 2, or a fault cost time is injected to compensate for possible transfer interruptions. This is done by adding the reconnection setup time (10s) and half of the time consumed to transfer the replica before disconnection, because fault tolerance techniques may require resuming or restarting from the beginning. In the simulation the average fault cost is calculated as in the below equation:

$$\text{Fault Cost} = 10 \text{ sec} + \text{resume or restart} \quad 3.1$$

The second operand in equation 3.1 has a significant impact especially if the fault tolerance technique requires restarting from scratch. Assuming uniform probability distribution over resume or restart events, both (resume or restart) are with 0.5 probability. This means half of the probabilities are resume and their cost is only more 10 seconds to the file download total time. The other half (restart) are to be penalized with extra half time of file downloading assuming the interruption occurs on average in the middle of the transfer. Manipulation of the transfer time occurs during the replication process of remote files (via a call to *replicateFile* method), assuming that the replication strategy in Optorsim would always attempt to replicate (strategy 2 & 3 in the parameters configuration file). The manipulation is located within the replicate method in *GridContainer* class called by the *replicateFile* method.

Finally, the availability values assigned to sites are assumed to be supplied by a specialized service in the real world, and simulated using a random generator. To ensure reproducible and thereby consistent simulation runs (for the purpose of

comparing the standard and modified Optorsim performance against the same set of availability parameters), the availability values have been stored in an external file, and a special class (*AvailabilityGenerator*) used to supply those values during the simulation. The modified version of OptorSim integrates the D-system equations and parameters, where the replica is chosen based on several parameters, including time, cost, security, availability and the user preferences. The later version has many variations in which α , β and the standard deviation are included, and in the same time some parameters like security parameter might be excluded. When the inclusion of the security factor or cost is desired, the values for each site's security or cost is generated randomly, and that value is used in the D-system calculations with no further impact, because the security of a given site does not affect the total job execution time. It is mainly a matter of users' preferences or QoS perceived by the users.

On the other hand, the availability factor does have an impact on the total job time, as the interruptions in site availability would lead to prolonged file transfers and longer job execution times. This effect is important to simulate too, so that the advantage of catering for the availability in replica selection algorithms could be realized. That is, standard Optorsim simulation has no provision for sites availability effects, and it is expected to perform inferior to an algorithm that is aware of the availability factor.

3.3.3 Dataset and Configurations

The default settings of OptorSim are utilized. They are copied from the EU data grid parameters. The bandwidth between the two sites is marked in Figure 3.6. In addition, the default OptorSim system workloads' values and parameters' values that have been utilized are based on the system utilized and presented in chapters 4, 5 and six (The detailed parameters' values of each site are included in the example folder within OptorSim package. These values represent the real values of the EU data grid). There are several configuration files used to control various inputs to OptorSim. The grid configuration file describes the grid topology and the content of each site. These contents are the available resources and the network connections to other sites. The

job configuration file contains information about the simulated files, jobs and the site policies for each site (the list of files each site will accept). The simulation parameters file contains various simulation parameters which the user can modify. If the user wishes to simulate background network traffic, a bandwidth configuration file is needed along with several data files to describe the simulated traffic. The simulation accomplished on an Hp desktop with 2.8 G CPU and 2 G RAM. The operating system used is Windows Vista (Tm) home basic 6.0 Edition.

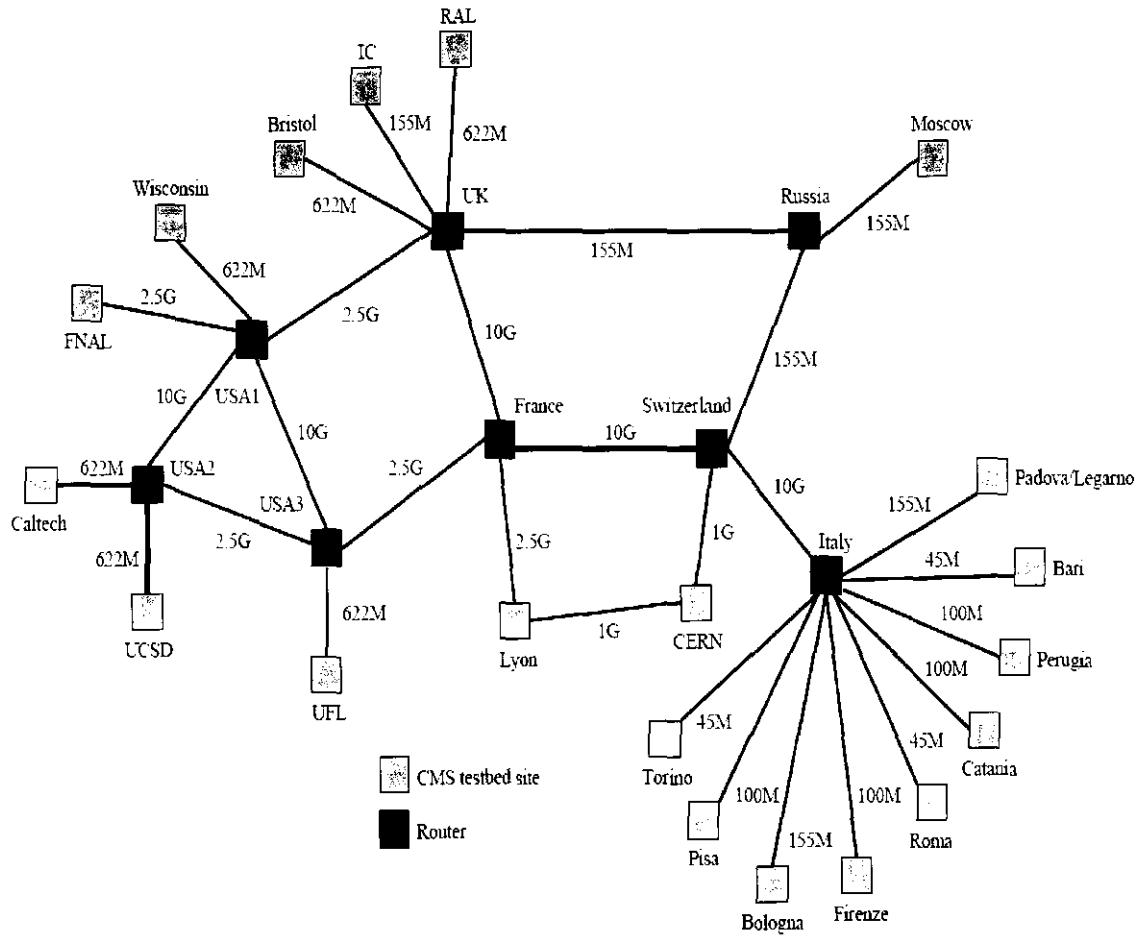


Figure 3.6: Grid Topology for CMS Test Bed

3.4 Summary

In this chapter, an overview of the proposed systems has been introduced and the methodology applied in this research has been presented. Additionally, a brief description of the OptorSim grid simulator and its functions in replica selection has been demonstrated. The extensions that have been applied to OptorSim to fit the proposed system have been clarified. The experiments setup, the configuration of OptorSim and dataset has been explained.

CHAPTER 4

THE IMPACT OF SITE AVAILABILITY IN REPLICA SELECTION

4.1 Overview

This chapter introduces the theoretical background of A-system. Then A-system's functional and non-functional requirements are presented. Next to that, the system design, new parameters and features are described. After that, rating parameters and estimating the best site are explained. Finally the experiments and results discussion are elaborated.

4.2 Theoretical Background

In chapters one and two, it has been explained that the key problems to be sorted out in this thesis can be solved by some available replica selection systems. However replica selection systems still requisite more optimization techniques such as taking into account more factors to enhance the performance of replica selection systems. Optimization techniques increase the efficiency of the replica selection systems and thus the whole grid infrastructure. As it has been presented in the previous chapters, replication and distribution of data among diverse grid sites are needed to address the requirement to increase data accessibility, reliability and availability. Replicated data lead to the requisite of replica selection, a process which selects one replica location from among many replicas based on their response times. The response time is a critical factor that influences the job turnaround time. In previous studies, data transfer time was utilized to estimate the response time. However, measuring transfer time alone is insufficient.

The continuity of service provided by the selected site plays a major role in assuring that the estimated response time will be maintained and not interrupted. This is due to the local policies of the provider that offers services to outsiders for specific times or hours only. According to the authors of [39], once a user is allowed to gain access to a resource based the access policy, the usage Service Level Agreement (SLA) determines how much of the resources the user is permitted to use. Just to recap: in the literature [6, 64-67], availability signifies the production of a number of copies for a single file (resource) in order to make it constantly available [145]. Availability in this research is defined as the capability of a given resource to fulfill a given task until it is completed. To distinguish between these two definitions, site availability or accessibility has been used to refer to the second definition. In [146] it is reported that only 65% of users' submitted jobs are executed successfully due to unknown causes of failure. The main causes of failures within grid infrastructures are grid component failures, network failures, information faults, and excessive delays. Grid component failures involve both software and hardware account for 25%-30% of the total failures. However, according to [147] the Open Science Grid (OSG) [148], encountered a 30% job submission failure rate with 90% of them due to disk filling errors, gatekeeper overloading, and network disruptions.

Though many enhancements have been done, the grid keeps growing in both size and complication. The total improvements are often not enough: for instance, the LCG grid [149] is still reporting about a 25% error rate [150]. Troubleshooting grid middleware is very challenging due to large number of interconnected components. For example, one action, like reliably transmitting a directory of files, could result in the coordination of a wide-ranging collection of loosely coupled software tools. Each of them normally generates its own log files in their own log format, semantics, and identifiers. To troubleshoot a problem as it cascades from one component into the next, this information must be combined to form a logically consistent trail of activity.

Causes of failures are mostly vague and request further investigations. Although, it is believed that excessive delays and the insufficient time of the resources to complete tasks are among of the reasons. Therefore integrating site availability in the replica selection process is necessary to avoid such faults and delays. None of the researchers has introduced site availability with the same concept that has been

specifically detailed in this research. Site availability is defined as: The relationship between the operating time declared by the service provider to serve certain VOs and the required time to transfer a file from the same provider during the replica selection decision process.

A basic alternative solution that is related to the research problem has been proposed in this chapter. This solution is encapsulated in replica selection system which is termed as: Single User Availability based Replica Selection Decision System in Data Grid (A-system).

Site availability incorporation has been highlighted in this chapter as a new intervention for a deliberated estimation of response time, enhances the data grid environment. Incorporating site availability as a selection factor in replica selection algorithm provides replication management systems with more guaranteed response time estimation. The overall system requirements, design, performance metrics, simulation results and discussion are presented in details in this chapter followed by a summary.

4.3 A-system Requirements

A number of non-functional and functional requirements have been introduced by A-system including: transparency, scalability, and performance.

Transparency: The proposed solution should provide the users' jobs with the required replicas on behalf of the users without any user intervention, while the system internal mechanism and complexity are hidden from the users.

Scalability: The proposed solution should show a high level of scalability without significant performance degrading when the related parameters, such as file sizes, number of requests, etc. are increased [151].

Performance: The proposed solution should perform better than the other similar systems. Therein, the evaluation metrics are used in order to measure the performance of the system.

However, the main functional-requirement of A-system is the replica selection. In replica selection functionality, A-system selects the best replica location from among many replicas distributed across the grid sites. As discussed before, the best replica location in the scope of A-system has two meanings. The first one refers to the site location that houses the required replica and which is capable of delivering the underlying replica in minimum turnaround time. The second meaning refers to the replica location which is available for sufficient time to complete the replica movement. A-system utilizes the Euclidean distance with the concepts from K-Means algorithm to solve the complexity of the selection problem because the problem includes compromised values to be selected. K-Means is explained in chapter 2.

4.4 A-system System Design and Features

A-system could depend on many existing successful data grid core services, such as RLS [44], that provides the physical file name locations as shown in Figure 4.1. The information about an individual resource or set of resources is collected and maintained by a Grid Resource Information Service (GRIS) daemon [152]. GRIS is designed to gather and announce system configuration metadata describing that storage system. For example each storage resource in the Globus data grid [94] incorporates a GRIS to circulate its information. Typically, GRIS informs about attributes like storage capacity, seek times, and description of site-specific policies governing storage system usage. Some attributes are dynamic varying with various frequencies such as total space, the available space, queue waiting time and mount point. Others are static such as disk Transfer Rate.

A-system, as illustrated in Figure 4.1, commenced by receiving the user request via the Grid Resource Broker (RB). Then the RB retrieves related physical file names and locations from the RLS. Subsequently, the system receives information about the sites that hold the replicas and their network status from the GRIS such as: NWS [105], Meta-computing Directory Service (MDS) [153] and Grid File Transfer Protocol (GridFTP) [52, 153]. Then, the best replica site for the concerned user's job is chosen. In this context, the replica that promises the minimum response time with the least probability of disruption is the best. Hence, the new high-level service

replica selection system is an optimization approach. The proposed system is designed to perform caching not replication. Caching [36] occurs on the user side in which the user decides which replica is the best and copies the required replica to the local site. On the other hand, replication occurs on the server side in which the server that houses the replicas decides which replicas are to be created and where to be placed.

The exact sequence of steps in the proposed system is as follows:

- Collects jobs from the Resource Broker.
- Collects replica of physical file names and locations from Replica Location Service.
- Collects sites' operating hours from their log files.
- Collects sites' current criteria values like bandwidth from the information service providers for instance GridFTP, NWS, and MDS.
- Calculates the response time and site availability of each site and rates them by percentage. The site that demonstrates the best Response Time (T) will be given the value of 100% and the rest of sites will be rated based on their performance in comparison to the site that gets 100%. On the other hand, the rank of site availability 100% will be given to the site or the sites that show sufficient time to complete the transfer even if the dynamic conditions of the network are degraded to some extent. A site is assigned 100% site availability if it shows a level of availability that is equal to the predicted download time plus the reserve time required to accommodate any decline in the network. Site availability of the remaining sites is rated based on the predicted download time and how much time is required for the reserve time.
- Selects the best location that houses the required replica for the grid user. The best location is the one that shows minimum transfer time and the least probability of failure to complete the job due to site downtime.

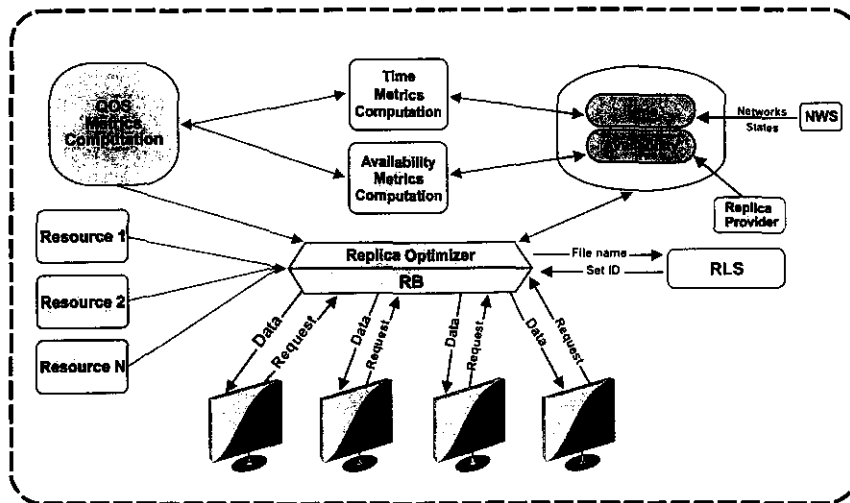


Figure 4.1: Overview of A-system.

This study focuses on incorporation of site availability as an essential element in the process of locating the best replica. Site availability in this work is defined as the relationship between the required time to download a replica and the remaining time declared by the site that offers this service. The remaining time of any site is the remaining over time to serve the user. The response time is defined as the time elapsed when moving data file from one site to another. The following subsections detail the calculation of site availability, response time, remaining time and the best site selection:

4.4.1 Rating Time

Response time is a dynamic value changing as time passes based on the load on the network or the storage devices. However it is anticipated to be steady for a while or change slightly positively or negatively. But since it is difficult to estimate the response time in a dynamic manner, the response time can be estimated at the decision time (NWS applies fast statistical models to probe histories to make performance forecasts). The response time's dynamicity is considered by integrating the new factor site availability.

The response time (T_i) is estimated by using the following equations proposed in a recently published work [47]:

$$T_i = T1_i + T2_i + T3_i \quad 4.1$$

$T1$ represents the transfer time, $T2$ represents the storage access latency and $T3$ represents the requested waiting time in the queue. $T1$ represents the data transmission via a wide area network, which depends on the network bandwidth, either a wide area network (WAN) or a local area network (LAN) and the file size which is computed by the following equation [154]:

$$T1_i = \frac{\text{File Size (MB)}}{\text{Bandwidth (MB/SEC)}_i} \quad 4.2$$

In general, the operating systems schedule the disk I/O requests in a manner that improves system performance [155]. The process of scheduling is implemented by maintaining a queue of requests for the storage device. Therefore, the storage speed and the number of requests in the queue play a major role in the average response time experienced by applications. As a result, storage access latency ($T2$) is the delayed time of the storage machines to cater the requests and the delayed time depending on the file size and the storage type. Hence, $T2$ is increased due to larger data files. Moreover, different storage machines have discrepant speeds (data transfer rates) during I/O operations. For example, a tape drive is slower than a disk pool and there are many types of tape drives with different speeds. For instance: the Hewlett-Packard (HP) Storage Works Ultrium 920 Drive speed = 120 Mega Bytes per second ($MBps$) while the HP Storage Works Ultrium 448 Drive speed = 24 $MBps$ [47]. Storage access latency ($T2_i$) is calculated using the following equation:

$$T2_i = \frac{\text{File Size (MB)}}{\text{Storage Speed (MB/SEC)}_i} \quad 4.3$$

Storage machines receive many requests at the same time, but they can only serve one request at a time. This leads to pending the requests of waiting in the queue. Input data transfer must be performed prior to an actual request. Similarly, output data transfer must be completed after an actual write process request. This buffering technique balances required time for requests waiting in the queue and the required time for storage media to serve the request in process [155]. Furthermore, the site will be busy during the period that it transfers any replica from the storage machine to the

network. Any new incoming data requests have to wait for the transaction to complete and for the requests that join the queue prior to the underlying request [154].

Consequently, the new request should wait all the earlier requests to be processed in the storage queue. The waiting time is the sum of time from the first request in queue to the last. Each of these times is the storage access latency time ($T2_i$). The request waiting time in queue ($T3_i$) is calculated using the following equation:

$$T3_i = \sum_{i=1}^n T2_i \quad 4.4$$

(n) Represents the number of requests which are waiting in the queue prior to the underlying request. To make it simple, this work assumes the queuing model is M/M/1/N Poisson arrivals and service. The queuing model represents a single server which has a waiting queue only for N customers (including the one in service). The discipline is the First Come, First Served (FCFS) [156]. Substituting Equations 4.2, 4.3 and 4.4 in Equation 4.1 produces:

$$T_i = \left[\frac{\text{File Size (MB)}}{\text{Bandwidth (MB/SEC)}_i} \right] + \left[\frac{\text{File Size (MB)}}{\text{Storage Speed (MB/SEC)}_i} \right] + \left[\sum_{i=1}^n T2_i \right] \quad 4.5$$

However, it is worth mentioning that modern storage systems with disks and flash memories allow networking and storage to occur simultaneously. Hence, the second version of Equation 4.5 can be as follows:

$$T_i = \text{Max} \left\{ \left[\frac{\text{File Size (MB)}}{\text{Bandwidth (MB/SEC)}_i} \right], \left[\frac{\text{File Size (MB)}}{\text{Storage Speed (MB/SEC)}_i} \right] \right\} + \left[\sum_{i=1}^n T2_i \right] \quad 4.5a$$

Therefore the replica selection systems should be aware of the technology utilized in each site in order to estimate its response time accurately. However, the proposed system is not limited to using the abovementioned data transfer speed models. Any other valid model could easily replace the above mentioned models as an alternative solution (the main focus in this thesis is equation 4.6).

Rating sites based on their response time (T_{0i}) is denoted by the following equation:

$$T_{0i} = \frac{\min[T_i]_1^n}{T_i} \times 100 \quad 4.6$$

where the estimated value of T_i can be done using any valid model or estimator.

For example, as shown in Table 4.1, which reflects real bandwidth, storage speeds and file sizes, the estimated download time based on Equation 4.5 from sites 1, 2, 3 and 4 are 295s, 249s, 333s and 109s respectively. Site 4 displays the minimum download time so it is rated as a 100% site, site 2 is rated based on Equation 4.6, $\frac{109}{295} \times 100 = 36\%$ while site 3 is rated $\frac{109}{249} \times 100 = 43\%$ and site 3 is rated $\frac{109}{333} \times 100 = 32\%$. As a result, all sites are rated based on estimated download time to make the selection decision in the next step feasible and easier. The content of Table 4.1 will be discussed in detail in the following subsections.

Table 4.1: 10 GB and 100 GB Replicas with Different Metric Values for: Common Storage Speed and Bandwidth, Queue Waiting Time and Remaining Time

File Size (GB)	Storage Speed (MB/s)	Bandwidth (Mbps)	Queue Waiting Time (s)	Estimated Download Time (s)	Remaining Time (s)	Number of Servers (Nodes)	Queue Size (Nodes)	Quality of Service (QoS)	Scaled Standard Deviation	Best replica T _i
10	150	45	0	295	36	500	84	46	3.39	49.39
10	300	156	150	249	43	300	60	49	1.20	50.20
10	600	622	300	333	32	70	10	79	1.56	80.56
100	600	622	1200	1233	8	2500	100	65	6.51	71.51
100	150	45	100	395	27	400	50	62	1.63	63.63
100	600	622	75	108	100	150	69	21	2.19	23.19
100	150	45	100	395	27	600	75	54	3.39	57.39
100	300	156	200	299	36	150	25	69	0.78	69.78
100	150	45	0	2958	21	2500	42	69	1.48	70.48
100	300	156	150	1147	55	2000	87	33	2.26	35.26
100	600	622	400	745	86	500	34	47	3.68	50.68
100	600	622	600	935	67	1000	53	40	0.99	40.99
100	150	45	100	3058	20	3700	60	63	2.83	65.83
100	600	622	700	1035	61	1500	72	33	0.78	33.78
100	150	45	1200	4158	15	8500	100	60	6.01	66.01
100	300	156	400	1397	45	1900	68	44	1.63	45.63

4.4.2 Rating Site Availability

Site availability is the relationship between the operating time declared by the service provider to serve certain VOs and the time required to transfer a file from the

same provider during the replica selection process. Therefore, site availability (A) is computed as follows:

1. Ascertaining the remaining operation time (or allowed time) in seconds (R_S) from the site.
2. Estimating the required time to transfer the file (T_S).
3. Site availability is calculated by:

$$A = \frac{R_S (SEC)}{T_S (SEC) \times 2\alpha} \quad 4.7$$

The value of α is measured based on the network expected performance and the expected download time as well. The replicas usually are very large in size that is why they require long time to be downloaded. During this time, the network performance is prone to change either negatively or positively. The more stable the network condition is, the smaller value of α is required. For example, if the network performance shows that the real time to transfer a file is two times more than the estimated transfer time T_S , then α should be equal to 2. The value of α can be obtained based on some factors like: place, workdays, holidays, weekends, mornings, evenings, midnights and the comparison of file transfer history and estimated time transfer history. The minimum value of α should not be less than one. This is when the replica download time estimation is 100%. The value of α is obtained from the history information by comparing the estimated transfer times with the actual transfer times. On the other hand, the maximum value of A should not exceed 100% because this value is adequate and exceeding the (100%) is considered overqualified, which adds no values as demonstrated in Equation 4.8. In the example below, the value of (1) has been assigned to α , assuming 100% accuracy in download time estimation. However, based on this study approach this number should be multiplied by 2 in order to be more sure that the transfer will be commenced and terminated from the same site and to avoid any risk of disconnection prior to download completion as shown in Equation 3.7. Hence, the minimum acceptable value for A is 50% but a higher value increases the success rate.

On the other hand, estimating α requires more attention, which is outside the scope of this study. Addressing this estimation is planned to be in a future work.

Site availability is rated as follows:

$$A_0 = \begin{cases} 100 & , R_s \geq \alpha T_s \\ \frac{R_s(SEC)}{T_s(SEC) \times 2\alpha} \times 100 & , R_s < \alpha T_s \end{cases} \quad 4.8$$

For example, using the same data shown in Table 4.1, the estimated download time based on Equation 4.5 from sites 1, 2, 3 and 4 are 295s, 249s, 333s and 109s respectively and the remaining operating time for each are 500s, 300s, 70s and 200s respectively. Assuming that the value of α is 1, the site availability for site 1 is $\frac{500}{295 \times 2 \times 1} \times 100 = 84\%$, and the site availability for site 2 is $\frac{300}{249 \times 2 \times 1} \times 100 = 60\%$. The rest of the calculations are shown in Table 4.1.

4.4.3 Estimating the Best Site

The new approach proposes an imaginary ideal or model value to be 100% Time (T) and 100% Site availability (A) as shown in Figure 3.2. The best site is the one with the closest distance (d) to the ideal value (T in Figure 3.2). It has been titled as the quality distance (qd) and is calculated using the following equation:

$$qd = \frac{\sqrt{(100-T_0)^2 + (100-A_0)^2}}{\sqrt{2}} \quad 4.9$$

The distance in Equation 4.9 is divided by $\sqrt{2}$ to normalize its value to be between 0 and 100. The smaller the qd value, the better the site.

As shown in Figure 3.2, site T is the best site because it is the closest to the model value. Not having site T, the algorithm will select site A, B, C or D randomly because they all have the same distance from the Model value. In fact, the best in this scenario is site B because it is composed of two similar or almost similar values. This signifies a balanced solution, that is not extreme for site availability or transfer speed as opposed to site F. Site F displays high site availability but low quality transfer speed,

yet, is still better than site A. Site A, displays high-quality transfer speed and low-quality site availability which could lead to a fault (disconnection).

Moreover, it is clear that site R is better than sites A, C and D. To select a balanced solution and to avoid the extreme values as experienced in sites A or D as illustrated in Figure 3.2, the Standard Deviation (sd) is conceptualized by yielding a balanced optimal composition of time and availability. For example $sd(70,70) = 0$, $sd(50,50) = 0$, $sd(30,30) = 0$ while $sd(60,40) = 14.14$ and $sd(70,30) = 28.28$, and thus, the new equation for finding $q\bar{d}$ is modified to be as follows:

$$mdq = qd + sd(T_0, A_0) = \frac{\sqrt{(100-T_0)^2 + (100-A_0)^2}}{\sqrt{2}} + sd(T_0, A_0) \quad 4.10$$

Where sd increases the value of the quality distance which means degrading qd , if the values of its parameters are distant as explained in the previous example.

Conversely, this study's experiments have proved that adopting the standard deviation has sometimes side effects that could divert from the optimal solution. For example, if site X has the combination (63,100) for time and availability, utilizing Equation 10, $mqd = 52.16$ and site Y has the combination (61, 72), $mqd = 40.78$ meaning Y is better than X, even when it is clear that X is better than Y for both parameters, site availability and time. This example proves that the standard deviation has side effects and needs to be utilized wisely. To overcome the problem of standard deviation, it has been scaled down by dividing it into a number β as in Equation 11. The result is, site X rating is corrected to be better than Y. The other sites' rates were corrected as well to reflect reality. Ultimately the A-system is formulated as multi-objective decision making problem and the last version of mqd equation is denoted by:

$$mqd = Min \left[\frac{\sqrt{(100-T_{i0})^2 + (100-A_{i0})^2}}{\sqrt{2}} + \frac{sd(T_{i0}, A_{i0})}{\beta} \right]_{i=1}^n \quad 4.11$$

where n is the number of sites that holds the required replica

Estimating the value β was carried out by using a comprehensive search for all possible paired values of availability A and time T (A, T). A table has been created

containing all the possible values of A and T . The value 50 has been assigned to availability in the first column, which is the minimum applicable value when $\alpha = 1$, the second 51 and so on until the last column has been given the value 100. The first row has been assigned the value 30 for T and the second 31 until the last row has been assigned the value 100. Table 4.2 depicts a summary of the real table. The objective is to find a value for β that satisfies the following conditions:

- 1- Decreases the value of mqd (smaller mqd , better performance) while moving in the table from top to bottom. It is logical that the pair (50, 95) is better than (50, 30). Certainly if both options are in hand, the former will be chosen.
- 2- Decreases the value of mqd while moving from left to right because it is logical that the pair (90, 30) is better than (50, 30).
- 3- Balances, to some extent, the values of A and T , for example (50, 50) is better than (90, 30) but (60, 44) is the best because 44 is faster than 30 and 60 is safer than 50.

Different values for β has been tried, from 1 onwards. Thus, a conclusion is reached and that is the value of 10 is the best. For instance, as shown in Table 4.2, beneath row 8 the value of mqd increases while T increases which is illogical and contravenes condition 1 as well. On the other hand, if the value of β is increased to be greater than 10, the (90, 30) will be better than the site rated (50, 50) resulting in an unbalanced combination. Practically, estimating β requires further researches which will be conducted by the researchers in the future. Therefore, at the moment, tuning β value is left to grid administrators and users' preferences because some users prefer speed over reliability or vice versa or a balance of the two. The research's preliminary experiments have found that the best value for β is 10 as presented in Tables 4.1, 4.2 and 4.3.

Table 4.2: Possible Paired Values of A & T & Various Values for β

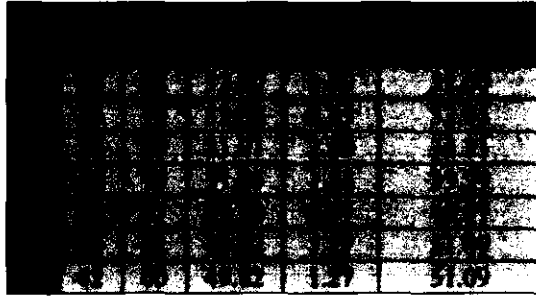
A	T	qd		A	T	qd		A	T	qd		A	T	qd		A	T	qd		
50	30	60.83	74.97	62.24	60	30	57.01	59.13	70	30	53.85	56.68	80	30	51.48	55.01	90	30	50.00	54.24
50	36	57.43	67.33	58.42	60	36	53.37	55.06	70	36	49.98	52.38	80	36	47.41	50.52	90	36		
50	37	56.87	66.07	57.79	60	37	52.77	54.39	70	37	49.34	51.67	80	37			90	37	45.11	48.85
50	38	56.32	64.81	57.17	60	38	52.17	53.73	70	38	48.70	50.97	80	38	46.07	49.04	90	38	44.41	48.08
50	39	55.77	63.55	56.55	60	39	51.58	53.06	70	39	48.07	50.26	80	39	45.39	48.29	90	39	43.71	47.32
50	40	55.23	62.30	55.93	60	40	50.99	52.40	70	40			80	40	44.72	47.55	90	40	43.01	46.55
50	44	53.08	57.33	53.51	60	44			70	44	44.92	46.76	80	44	42.05	44.59	90	44	40.22	43.48
					60	50	45.28	45.98	70	50	41.23	42.65	80	50	38.08	40.20	90	50	36.06	38.88
50	94	35.61	66.72	38.72	60	94	28.60	31.00	70	94	21.63	23.33	80	94	14.76	15.75	90	94	8.25	8.53
50	95	35.53	67.35	38.71	60	95	28.50	30.98	70	95	21.51	23.27	80	95	14.58	15.64	90	95	7.91	8.26

The modified distance mqd will be titled as TA in this study because it is composed of time and site availability and is given a new metric TA instead of meter (cm or km) because a normal distance is not measured her. TA is derived from Time and Site availability where the site with the smallest TA is the best since it is the closest to the imaginary ideal value.

Table 4.3 is a mathematical example of this research's approach where column 1 represents the value of site availability, column 2 represents the estimated download time, column 3 represents the distance from the model value, column 4 is the standard deviation of the two values for each site (estimated download time and site availability) divided by 10 and column 5 is the total of columns 3 and 4. Again, as shown in Table 4.3, qd is the lowest in row 3, with the values 56, 90 TA for site availability and time respectively. However, it is clear that a value of 56 for site availability is very unreliable and thus prone to fault. As a result, this is not the best combination even when the value of time is the highest. Therefore, standard deviation corrects the selection as can be seen in row 1, which shows the values site availability and time values of 68 each, as the best selection and row 2, as the second choice if row 1 is not available.

On the other hand, the new algorithm excludes from the selection any site with site availability less than 50. For instance, referring to Table 4.3, if sites 1 to 5 do not exist and the competition is only between sites 6 and 7, and both of them have the same TA value, the winner is site 6 because the site availability for site 7 is less Than 50% which is for sure not enough.

Table 4.3: Example of Applying the Proposed System



The Pseudo code below emphasizes the detailed system:

1. get R (list of physical file names and locations for the required replica) from RLS
2. getRs for each replica from the data grid's log file
3. estimate β
4. $i=1$
5. while R not empty
 - 5.1 calculate T_0, A_0
 - 5.2 calculate $mqd(i) = \frac{\sqrt{(100-T_{i0})^2 + (100-A_{i0})^2}}{\sqrt{2}} + \frac{sd(T_{i0}, A_{i0})}{\beta}$
 - 5.3 $i = i+1$
6. best = $mqd(1)$
7. $j=2$
8. While $j \leq i$
 - 8.1 if $mqd(j) < \text{best} \ \& \ A_0(j) > 50$
 - 8.1.1 best = $mqd(j)$
9. halt

4.5 Simulation Setup

The default settings of OptorSim were utilized. They were copied from the EU data grid parameters. The bandwidth between the two sites is marked in Figure 3.6. In addition, the default OptorSim system workloads' values and parameters' values were

utilized as shown in Table 4.4 (The detailed parameters' values of each site are included in the example folder within OptorSim package. These values represent the real values of the EU data grid).

There are several configuration files used to control various inputs to OptorSim. The grid configuration file describes the grid topology and the content of each site. These contents are the resources available and the network connections to other sites. The Job configuration file contains information on the simulated files, jobs and the site policies for each site (the list of files each site will accept). The simulation parameters file contains various simulation parameters that can be modified by the user. If the user wishes to simulate background network traffic, a bandwidth configuration file is needed along with several data files to describe the simulated traffic. The simulation accomplished on an Hp desktop with 2.8 G CPU and 2 G RAM. Since OptorSim does not consider site availability, it has been amended by assigning service hours to each site ranged from 1second to 24 hours (sites available for less than 1 second are not declared by replica catalog). Thereafter, if the simulator faces a selected replica from a site with insufficient operating time, it will then increase the replica transfer time based on the expected delay. This is done by adding the reconnection setup time (10s) and half of the time consumed to transfer the replica before disconnection because fault tolerance techniques may require resuming or restarting from the beginning. In the simulation the average fault cost is calculated as follow:

$$Fault\ Cost = Rr + Trls + Rd + Cs + Ror \quad 4.12$$

Rr: Required time to recognize that there is a fault

Trls: Time to inquire and get the response from RLS

Rd: Replica selection decision time

Cs: Connection setup time

Ror: Resume or restart from scratch, based on fault tolerance technique

In the simulation, *Rr* and *Trls* are set to 2s each, *Rd* 1s and *Cs* set to 5s each. The total is 10s, which is not that critical for usually huge replicas but in contrast, *Ror* has a significant impact especially if the fault tolerance technique requires restarting

from scratch. Fault tolerance techniques have an important impact to the replica selection process, which will be addressed in future work.

Table 4.4: Workload and System Parameter Values

Description	Value
Number of files	200
File size	1 GB
Storage available at an SE	30-100000 GB
Number of files accessed by a job	3-20
α	1-2
β	10-18

4.6 Performance Metrics & Cost

In a grid environment, users normally send their jobs to the RB, which locates the best site to carry out the jobs. The executed jobs commonly require some data files. The optimizer locates the best locations of the required files. However, each site services the users based on their local policy which allows the users to be served for a specific number of hours per day or night or even possibly only on weekends. Hence, selecting the site at an improper time could lead to disconnection. Depending on the fault tolerance approach, the job could be resumed by another site (which may also be prone to disconnection if site availability is not considered or it may be required to restart the entire process from scratch. Therefore, the job's time requirement will be increased. The job's time requirement begins from the time the RB transmits the job until the time that the job has completed its execution. This time is called the job turnaround time and includes the response time. The best replica selection according to the new system decreases the response time and consequently decreases the job turnaround time. Therefore, the *Average Job Turnaround Time (AJTT)* is suitable for a performance metric that evaluates the overall system performance and can be measured by using the following equation:

$$AJTT = \frac{(\sum_{i=1}^n T_{out} - T_{in})}{n} \quad 4.13$$

T_{in} represents the time the job is received by the system to begin execution, T_{out} represents the time the job has completed the execution, and n represents the total number of jobs processed through the system. On the other hand, the new system considers two factors to select the best replica. The first is time expenditure and the second is site availability. Therefore, a new Quality of Service (QoS) value composed of the two factors (*Time* and *Availability*) has emerged and titled TA . The lowest value of TA means the best quality. Table 4.1, illustrates scenarios for 10 GB and 100 GB replicas with different metric values for: storage speed, bandwidth, queue waiting time and time remaining. Column 6 shows that the time metrics combinations for the best sites are located in rows 4 and 7 for the 10 GB replica. Row 4 is the best due to high site availability and less disconnection risk. On the other hand, for the file size of 100 GB, the candidate site shown in row 12 reveals the best transfer time of 735s but was discarded because it is available only for 500s, which is not sufficient and a certain error will occur. The optimizer selected the site presented in row 13, which shows 28.62 TA . Even the site presented in row number 14 shows a 62s better transfer time. This decision is due to the anticipated high risk from site 14. It is difficult to estimate the cost of the new approach. In the aforementioned example, the cost was 62s, although it is worth for a reliable transfer, but different situations have different costs.

4.7 Results and Discussion

OptorSim is equipped with different built-in replication strategies (i.e., Least Recently Used (LRU), which always replicates and deletes the least recently used file, Least Frequently Used (LFU), which always replicates and deletes the least frequently used file and the Economic Model-Binomial (EB), which replicates, if it is economically advantageous, using a binomial prediction function for values). However, within these replication strategies only one OptorSim built-in replica selection system (OsBi) is applied. It selects the best replica locations that show the least transfer time [47]. The simulations have been performed to calculate $AJTT$ as the average of the total time required for all jobs, measured in seconds. The simulation commenced by investigating the best value for β . Several values have been tested for

β starting from 1 until 18. On the other hand, the abovementioned tests have been performed utilizing different values for α under LFU replication strategy. Table 4.5 depicts the results of these experiments wherein the best value of β is 10 when $\alpha = 1$ or 1.5, and the best value of β is 9 when $\alpha = 2$.

Table 4.5: Average Jobs' Time for 500 Jobs with Different Values of α & β

β	<i>AJTT</i> when $\alpha = 1$	<i>AJTT</i> when $\alpha = 1.5$	<i>AJTT</i> when $\alpha = 2$
1	698314.10	1313303.40	1525233.60
2	678414.25	1046605.20	1370006.00
3	650086.10	1023575.94	1335122.10
4	716841.20	1159565.00	1324494.40
5	732999.25	918903.44	1315733.10
6	635794.00	1093129.10	1276658.80
7	680829.75	1418769.50	1376628.80
8	698921.40	978713.56	1353125.50
9	685447.56	1220957.00	1225239.60
10	594141.75	893544.75	1176878.20
11	979156.90	836304.30	1323419.00
12	753310.75	993881.50	1473392.90
13	743662.50	1106562.00	1240304.00
14	634496.50	937375.10	1514095.50
15	734516.50	1029952.30	1438059.80
16	665705.60	1133184.00	1618809.80
17	643339.60	1061195.90	1245205.20
18	801867.10	1104780.40	1318509.40

To verify that the site availability is the only difference between the A-system and the OsBi, both systems were run with site availability always set to 100%. The expectation was that similar performance would be achieved from both because response time is the only selection factor in the OsBi and should be in the A-system when site availability is 100%. However, the simulation results in Table 4.6 below were surprising. They show that A-system is less efficient than OsBi under all the three replication strategies. The justification for that is the number of jobs in this experiment is 100. Each of them is accompanied by 10 to 100 replicas, which means on average around 5500 replicas (decisions). Therefore, there will certainly be some overhead.

Table 4.6: Average Jobs' Time in Seconds for 100 Jobs when Availability is Always 100%.

Test #	LUR		LFU		Economic	
	A-system	OsBi	A-system	OsBi	A-system	OsBi
1	286111.66	231099.27	278278.66	230458.10	1060176.60	906395.90
2	275035.66	242358.60	271168.10	220449.84	1085247.60	1035999.10
3	284864.25	222627.05	255691.20	223970.70	913550.25	904781.75
4	238518.45	232944.86	288959.16	216565.40	1005183.75	954523.25
5	256388.00	224360.67	242888.72	224768.03	977966.00	1062441.60
AJTT	268183.60	230678.09	267397.17	223242.41	1008424.84	972828.32
Efficiency	13.99 %		16.51 %		3.53 %	

Based on the abovementioned experiments, the remaining simulation experiments have been carried out by setting the value of β to 10. In view of the fact that the number of jobs influenced data transfer time, the system's performance has been evaluated in three different scenarios by varying the number of jobs each time. In the first, second and third scenarios, the number of jobs were 100, 500 and 1000 respectively. Simulation has been executed 5 times for each scenario along with a predetermined site operating time scenario, utilizing both A-system and the OsBi. Experiments have been conducted using three different OptorSim built-in replication strategies, namely, LRU, LFU and EB. The new replica selection system has been tested by performing several executions on the same replicas with a different number of jobs. The results of the simulation has demonstrated that the AJTT based on A-system is less than the AJTT based on OsBi for all scenarios and under different replication strategies as shown in Tables 4.7 (a, b, c), which signifies that the proposed system outperformed the previous systems.

Table 4.7 (a): Average Jobs' Time in Seconds for 100 Jobs

Test #	LUR		LFU		Economic	
	A-system	OsBi	A-system	OsBi	A-system	OsBi
1	704189	1278467	628694	1544662	933333	933571
2	582321	1090155	747886	1013950	909764	955317
3	582266	1280359	579806	1243939	836163	946263
4	720064	1105045	706270	1248695	855435	956849
5	650280	1041018	648674	1161950	934881	964598
AJTT	647824	1159009	662266	1242639	893915	931120
Efficiency	44.11 %		46.70 %		4.00 %	

Table 4.7 (b): Average Jobs' Time in Seconds for 500 Jobs

Test #	LUR		LFU		Economic	
	A-system	OsBi	A-system	OsBi	A-system	OsBi
1	12108976	17167802	9644758	19077332	9065388	8516753
2	11204400	15578618	11485171	12449995	9751129	8698534
3	8571915	17896652	9990595	17365892	8654000	9336498
4	12741477	14618266	11045485	16848332	8335365	10864292
5	9886645	14733334	10801072	13096076	9451450	9190288
AJTT	10902682	15998934	10593416	15767525	9051466	9321273
Efficiency	31.85 %		32.81 %		2.89 %	

Table 4.7 (c): Average Jobs' time in Seconds for 1000 Jobs

Test #	LUR		LFU		Economic	
	A-system	OsBi	A-system	OsBi	A-system	OsBi
1	44425416	59635972	41978100	51684568	30145046	291046946
2	41558504	72720032	44334344	73405680	24623898	24063774
3	41331136	64606356	41013108	67973424	26333746	27421194
4	45374036	60879928	42693512	58488172	28583260	26592294
5	45420436	83777552	42533128	75843184	28765640	26274540
AJTT	43621905	68323968	42510438	65479005	27690318	79079749
Efficiency		36.15 %	35.08 %		64.98 %	

The results presented in the above tables also show clearly the A-system efficiency in comparison with the OsBi. But to draw a clear demarcating line between the two systems, a paired sample t-test has been used under different replication strategies. The results are included in Table 4.8.

Table 4.8: Paired Samples Statistics between A-system & OsBi

Replication Strategy	System	Mean	N	Std. Deviation	t	df	Sig. (2-tailed)
LRU	A-system	647824.00	5	65173.148	-8.680-	4	.001
	OsBi	1159008.80	5	112439.872			
LFU	A-system	662266.00	5	65913.977	-5.479-	4	.005
	OsBi	1242639.20	5	193691.527			
Economic	A-system	893915.20	5	45549.936	-3.081-	4	.027
	OsBi	951319.60	5	11868.565			

The paired-samples t-test has been conducted to evaluate the impact of the site availability represented by A-system on AJTT in comparison with OsBi under different replication strategies namely LRU, LFU and Economic. According to all

scenarios, there were statistically significances in FOST scores from A-system to OsBi. Under LRU A-system (M=647824.00, SD=65173.148) to OsBi [M=1159008.80, SD=112439.872, $t(4)= 8.680$, $p<.001$], under LRU A-system (M=662266.00, SD=65913.977) to OsBi [M=1242639.20, SD=193691.527, $t(4)= 5.479$, $p<.001$] and under economic A-system (M=893915.20, SD=45549.936) to OsBi [M=951319.60, SD=11868.565, $t(4)= 3.081$, $p<.001$]. These results reveal that A-system overcomes OsBi under all scenarios and shows how site availability QoS parameter is important.

On the other hand, Figures 4.2 (a, b, c) depict the average jobs' total time for A-system and the OsBi under the replication strategies LRU, LFU and EB where the number of jobs was 100, 500 and 1000 respectively. It is clear that when the number of jobs is increased, AJTT is increased, regardless of the algorithm or the strategy utilized. However, the increment will be more if site availability is not implemented in the algorithm. This is because of the increasing of the probability of selecting unavailable sites or sites available for an insufficient amount of time.

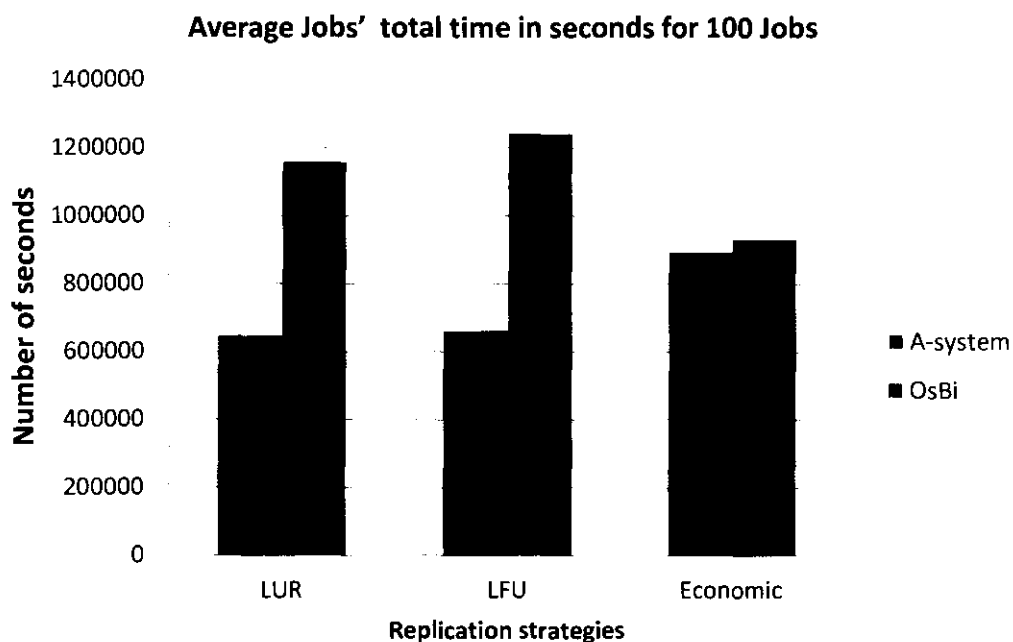


Figure 4.2 (a) : Average Jobs' Total Time in Seconds for 100 Jobs

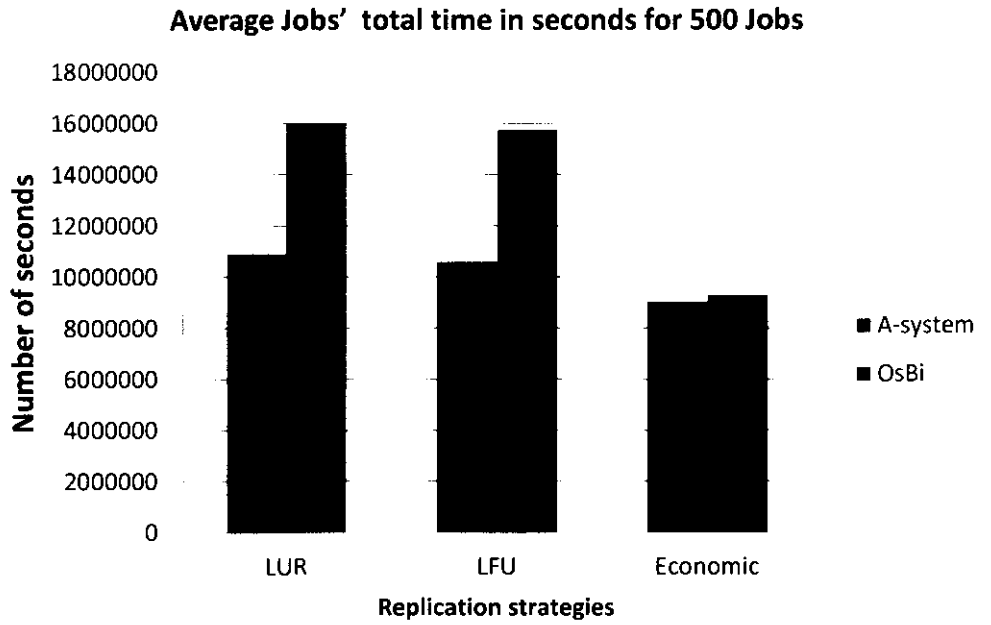


Figure 4.2 (b) : Average Jobs' Total Time in Seconds for 500 Jobs

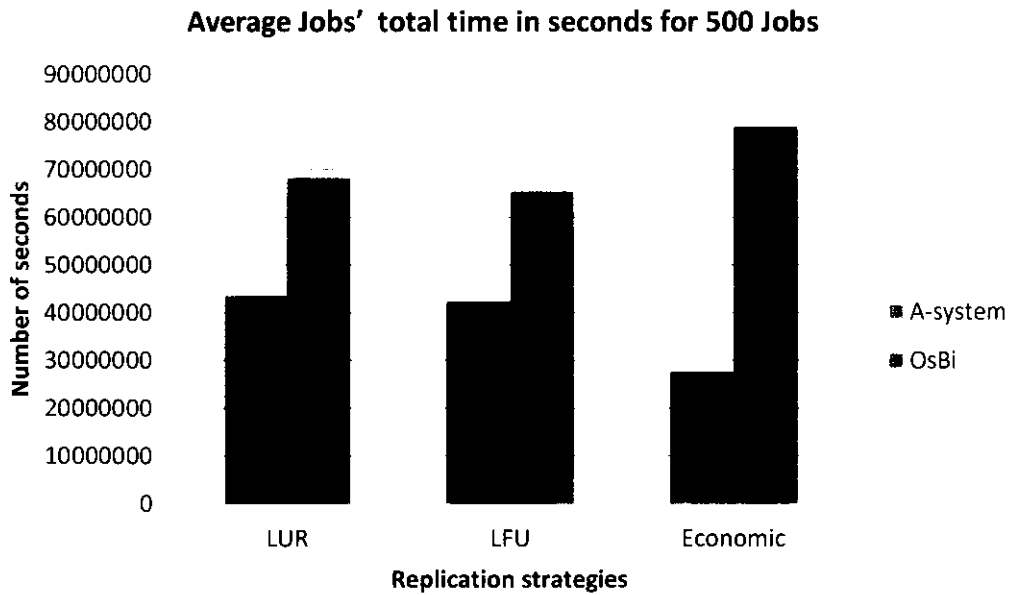


Figure 4.2 (c) : Average Jobs' Total Time in Seconds for 1000 Jobs

In real life scenarios, replica selection based only on response time could perform better than the proposed system if the selected sites display insufficient availability but still succeed to deliver the replicas without any disconnection, or if all the sites are available 24 hours per day.

4.8 Summary

This chapter introduces a new replica selection system in the data grid environment. The system engages a new QoS criterion specifically site availability in the replica selection process. The definition of the novel QoS criterion and its importance is provided in this chapter. OptorSim has been utilized to integrate the new parameter and to evaluate the system. The simulation experiments have been setup by expanding some modules in OptorSim. The strengths of the system have been investigated and the results of the experiments are presented. The simulation results have demonstrated that A-system enhances the performance of the grid environment and thus, decreases the job's average total time. A new network performance parameter α has been proposed and the impact of fault tolerance techniques against the download time is highlighted. Even though, integrating site availability QoS parameter improves the replica selection process but integrating more QoS parameters could result in more improvement. In the next chapter more QoS parameters will be added for more improvement of replica selection.

CHAPTER 5

MULTI QOS PARAMETERS REPLICA SELECTION

5.1 Overview

This chapter presents the background of D-system. Then D-system's functional and non-functional requirements are introduced. Next to that, the system design and the security parameter are explained. After that, the D-systemAp1 functionalities are described and tested, and the results are discussed. Similarly after that, the D-systemAp2 functionalities are described and tested, and the results are discussed.

5.2 Background

In the previous chapter, the focus has been on the importance of site availability as a QoS parameter that ensures the safety of jobs (tasks), prevents faults and reduces jobs times. Simulation results have proved the importance of site availability and the necessity that makes it highly required in the selection process. This chapter aims to further improving the selection process by considering more significant QoS parameters specifically security and users' preferences. Sometimes it is better to consider selecting data from secure sites rather than sites with short response times, especially if the required replicas are critical or confidential. The point is that the wide area networks are prone to violated and made unsecure by hackers, viruses, and many unauthorized users. On the other hand users may have their specific QoS requirements or preferences and they are the best to be consulted about that. Allowing the users to provide their preference could guide the replica selection systems to better solutions that provide the users with much more satisfaction.

The main objective of this chapter is achieved by performing more optimization techniques and by considering more QoS parameters in line with users' preferences in order to improve the replica selection system performance. The main purpose of optimization techniques is to increase the efficiency of systems. Improving the replica selection systems in data grids reflects positively the whole grid infrastructure. Therefore, this chapter offers some alternative solutions to the research problem that is encapsulated in replica selection system. This problem is termed as Single User QoS-based Replica Selection Decision System in Grid (D-system).

D-system has two approaches. The first one behaves as a replica selection decision maker independently from the users' preferences (D-systemAp1) while the other one is a user driven replica selection decision maker (D-systemAp2). So the only difference between the two approaches is that the second allows the users to make a choice among the available QoS parameters or in other words allows the users to impose their preferences in this respect. In this chapter, the complete D-system requirements, the design, the functionalities, performance metrics and the simulation results are discussed in details. Then after, a summary is presented.

5.3 D-System Requirements

A number of non-functional and functional requirements have been introduced by D-system including transparency, scalability, performance, and QoS.

Transparency: The proposed solution should provide the users' jobs with the required replicas without the user's intervention if the user has no specific preferences in terms of QoS. In this case, the user is given a balanced QoS. In case the user prioritizes certain QoS parameter over others, the user should show a minimal level of intervention. The system internal mechanism and complexity are not seen by the users.

Scalability: The proposed solution should scale well without significant degrading of performance when the related parameters, such as size of files, number of requests, etc. are increased [151].

Performance: The proposed solution should perform better than the other similar systems. Therein, the evaluation metrics are used in order to measure the performance of the system.

However, the main functional-requirement of D-system is the replica selection. In replica selection functionality, D-system selects the best replica location from among many replicas distributed across the grid sites. As discussed before, the best replica location has two meanings. The first one refers to the site location that houses the required replica and which is capable of delivering the underlying replica in minimum turnaround time and high level of QoS because the grid sites vary in their capabilities.

The second meaning refers to the replica location that is more with accordance to the user's preferences as different users have different preferences and even a single user could change his preferences based on the replica content or based on his commitments. D-system deploys the concepts from K-Means to solve the complexity of the selection problem because the problem includes heterogeneous values in the selection parameters, and the parameters usually are in conflict with each other.

5.4 D-system Design

D-system could depend on many existing successful data grid core services such as RLS [44] that are provided with the physical file name locations. Also D-system could get the required information from Information Service Providers (ISP) such as the NWS [105], which provides information on the network status. Next to that, D-system can use other services such as GridFTP [95] to transports replicas to grid users securely and to make logs of the end-to-end transfer data. Figure 5.1, depicts an overview of D-system and other related entities.

D-system evaluates each grid site that houses the required replica based on a set of parameters in order to achieve the best replica. The best replica is the one that shows a high rate and almost equal portions of all QoS parameters or the parameter that is identified by the user (satisfy user's preferences). As a result, D-system performs the following functionalities:-

- Receives the requests from the users or typically from the Resource Broker (RB).
- Receives the users' preferences if there is any or any priorities.
- Gathers the replica location information from RLS.
- Gathers the current network bandwidths, storage speed, storage requests queues and security information utilizing grid services from the ISPs and grid sites.
- Rates each QoS parameter out of 100%.
- Deploys the users' preferences into the system if there is any. Evaluate each alternative (Site) based on concepts of K-Means, site QoS parameters and the user's preferences.
- Select the best replica location for the users' jobs (requests).

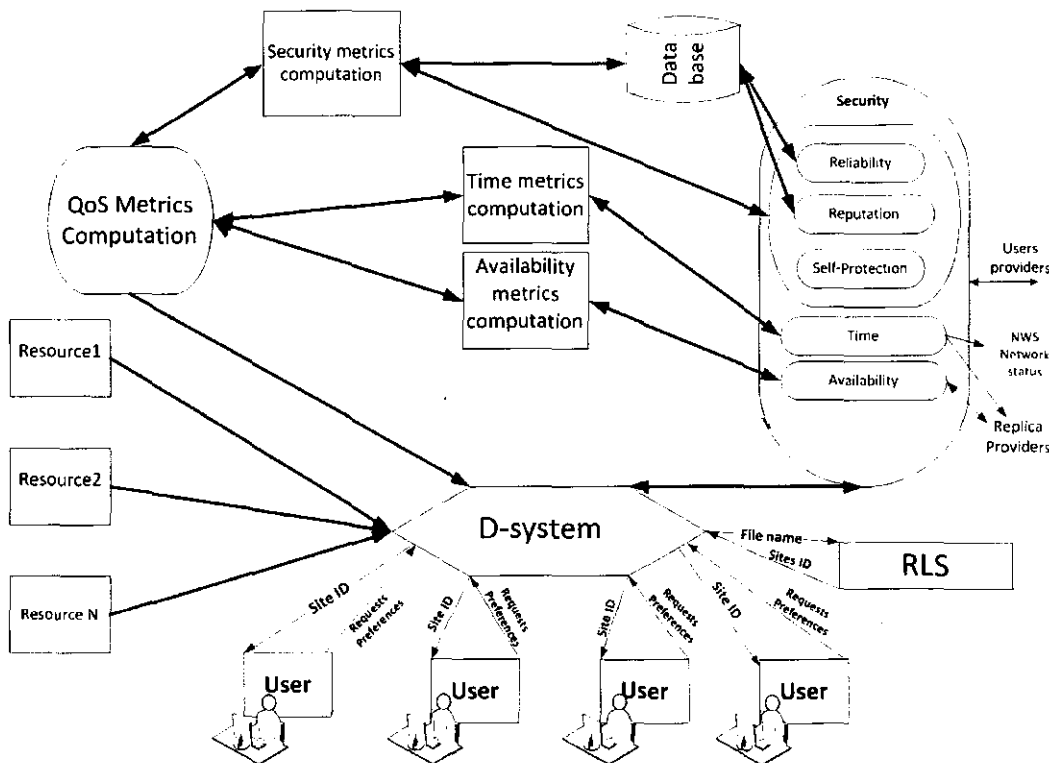


Figure 5.1: Overview of D-system and other Related Entities

5.5 D-system Detailed Design

The *D-system* performs the selection process to choose the best replica location for the users' jobs in almost minimum turnaround time and high level of QoS, in order to enhance the users' sense of satisfaction. Therefore, turnaround time, security, and site availability are called replica selection QoS parameters or simply QoS parameters throughout this thesis. In chapter 4, a detailed description has been provided for the time or turnaround time and site availability. The next section contains a full description for the security parameter.

5.6 Security Parameter

Each site that houses the required replica in the grid has a level of security to protect the data files from unauthorized users. The levels of the security [157, 158] are varied from one system to another. Many categories are found in the literature. The levels of security considered in this thesis explained below. In general, the purpose of a security mechanism is to provide protection against malicious parties. Traditional security mechanisms typically protect the resources from malicious users by restricting access to only authorized users. However, in a multitude of situations within distributed applications one has to protect oneself from those who offer resources. For instance, a resource that provides information can act craftily by providing untrue or misleading information. The traditional mechanisms are unable to protect the users against these types of threats. Trust can be used to overcome such threats in a distributed grid system. Therefore, trust can be helpful to provide entry level security such as authentication and access control. Trust is specified in terms of the relationship between a trust or trustee and the context in which the target entity is trusted.

The security model (S) employed in this research uses the computing Trust Factor (TF) proposed by [159]. According to them, TF consists of Self-Protection Capability (SPC) Reputation Weigh (RpW). They defined the self-protection capability of a site to include its ability to detect intrusions, viruses and unauthorized access as well as having secured file storage and job completing abilities. Conversely, they defined

reputation as a mechanism that offers a way to build trust through social control by using community based feedback about the past experiences of entities. Moreover, reliability [160] is also an important element to build trust because as reliable sites are should be more trusted than less reliable sites. The proposed security rating model extends beyond the model [159] by integrating reliability proposed by [160] with some modifications. The details of the calculation are presented in the succeeding subsections.

5.6.1 Self-Protection Capability Calculation

The Grid Organization Manager (GOM) maintains the self-protection capability of all entities in a grid organization. Frequently, each entity reports its self-protection capability trustfully and honestly to the GOM. The self-protection capability of an entity is calculated by aggregating the values of the security factors shown in Table 5.1. The values of these factors range between 0 and 1. Based on their contribution to security, a weight age is given to all the security factors and as a final point, the weight ages are aggregated to compute the self-protection capability. The self-protection capability is calculated using the following formula:

$$SPC = \sum_{i=1}^n W(i) * A(i) \quad (5.1)$$

Where (n) is the total number of factors, W_i is the weight age and A_i is the value of the factor.

5.6.2 Reputation Calculation

Since reputation is a multi-faceted concept [161], with many aspects (i.e. truthfulness, honesty, etc.). Reputation weight is calculated via feedback analysis concerning a multitude of security characteristics derived from the previous experiences of the user community. After usage, the users will provide feedback on the attributes to the Reputation Manager (ReMg) based on their experience. The feedback, which is obtained from all the users, is a value in the range between 0 and 1. The ReMg in the grid organization maintains the reputation weight of all entities. The security attributes considered for reputation are shown in Table 5.2.

Table 5.1: Security Factors Considered for Self-Protection Capability

IDS Capabilities	The capabilities of a site to shield the system against host and network based intrusions.
Anti-virus Capabilities	The capabilities of a site to protect against viruses and malicious codes.
Firewall Capabilities	The capabilities to protect the site from other network accesses.
Authentication Mechanism	The capabilities of the mechanism to verify an identity required by or for system security.
Secured File Storage Capabilities	The capabilities of a site to securely store the files needed for job execution.
Interoperability	The capabilities of a site to restrict interfacing of concurrent jobs.
Secured Job Execution	The capabilities of a site to securely execute the job

Table 5.2: Security Attributes Considered for Reputation

Consistency	The capabilities of a site to perform its required functions under stated conditions for a specified period of time
Confidentiality	The capabilities to prevent the disclosure of information to unauthorized users
Truthfulness	The capabilities of the site to protect against unauthorized data modifications
Security	The capabilities of the system to offer job execution and file storage protection
Privacy	The capabilities to keep some information isolated solely to oneself
Non-repudiation	The inabilities of something that has performed a particular action to later deny responsibility for the event
Authentication	The process of verifying an identity demanded by or for a system object. An authentication process comprises two steps: Identification and Verification
Authorization	Refers to the process of granting privileges to processes and ultimately, to users

Authorization differs from authentication. Authentication is the process used to identify a user. Once the user is identified (reliably), the privileges, rights, property,

and permissible user's actions are determined by authorization. Based on the aggregated feedbacks of all the security attributes of an Entity (E_a), reputation weight $RpW(E_a)$ is calculated using the following equation:

$$RpW(E_a) = \sum_{i=1}^n \frac{E_i}{n} \quad 5.2$$

E_i = The value of all the feedback factors (between 0 to 1)

n = The total number of factors

5.6.3 Reliability Calculation

Reliability in general is defined as the capability of a system or component to execute its requisite functions under stated conditions (in this research case operating hours) for a particular period. In this research, reliability is the extent of confidence where selected replica functions properly with no failures or crashes [162]. Reliability is calculated using the following formula:

$$Reliability(Rb) = \frac{Nt}{Nc} \times 10 \quad 5.3$$

Where Nt = Number of times the resource is available to the grid resource provider (Rb) and Nc = Number of resource access attempts performed with the condition that both numbers were attempted during the operating hours declared by the resource owner.

5.6.4 Security Rating & Trust Factor Calculation

The Trust Factor (TF) of each Entity (E) or a grid node is calculated by utilizing the SPC, Reputation weight age (Rpw) and Reliability (Rbw) weight age calculated as in the following equation:

$$TF(E_i) = SPC(E_i) + RpW(E_i) + Rbw(E_i). \quad 5.4$$

From this equation, each site is rated by assigning values from 0 to 100.

$$S_i = \frac{TF(E_i)}{\text{Max}_{i=1}^n \{TF(E_i)\}} \times 100. \quad 5.5$$

5.7 Replica Decision Maker (D-System)

According to the previous literature presented in chapter two, all the previous works that have addressed the replica selection problem are concerned with (or focused on) response time to choose the best replica. The best replica in this context is the one that shows the shortest time. Moreover, the authors of [163] have summarized the QoS by referring only to the time factors as shown in Figure 5.2. On the other hand, a recent work [113] has added the security parameter in the selection process as an important factor especially if the required replicas are breach sensitive. The point is that in wide area networks hackers, viruses, and many unauthorized users attempt to violate such networks making them at times unsecure. However, this approach did not focus on the best replica simultaneously in relation to both parameters response time and security. Instead and according to their work, the best replica is the one that achieve fairness based on history. On the other hand, site availability is a very critical parameter as presented in chapter 4 where most grid sites are available daily for a limited number of hours. Selecting the most available replica improves the grid system because there is more time available to resolve any problems that may arise (resume the download). However, it is confirmed from the literature that none of the previous works have addressed site availability at all. In addition to that, none of the previous works have considered the best replica based on a compromised solution that aims to increase the quality of the all QoS parameters simultaneously in a greedy manner.

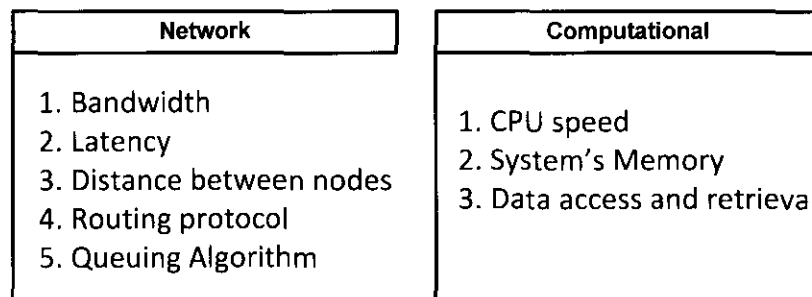


Figure 5.2 QoS Parameters in a Grid Distributed Between the Network and Computational Aspects.

The solution proposed in this research improves the grid environment by adding the parameter site availability to response time parameter as a critical factor that ensures completing the download from that node even with the existence of some

difficulties like malfunctioning or degradation in bandwidth or response time. In addition to that, the solution considers the security factor because it is vital for securing the data and the whole process. The selection criteria (response time, availability, and security) are heterogeneous; therefore, their rates values cannot be added to each other to get one value in order to make the selection decision. Furthermore, the criteria may contradict one another.

Therefore, the K-means concept model has been used in the selection process to solve the complexity and discrepancy problems. Moreover, the solution uses an efficient and simple clustering algorithm (K-means) as a technique to do the complex job of selecting the best replica. The best replica is the one that shows good response time, enough availability and an acceptable level of security at the same time. Hence, the proposed selection model should consider a comprehensive view of the criteria set to make the best selection.

Data can be divided into several non-overlapping homogenous groups that are called clusters. Each cluster contains objects that are similar amongst themselves and dissimilar to other groups' objects. This process is called clustering, and it is used in various applications in engineering statistics and numerical analysis. One of the most well-known clustering tools used in scientific and industrial applications is the K-means algorithm. The name originates from representing K clusters S_j by mean (i.e. weighted average) S_i of its points, called the centroid (i.e. the center of the cluster). The sum of Euclidean distance $d(x,y)$ between a point (X_i) and its centroid (S_j) is used as an objective function as shown in equation 2.1.

$$E = \sum_{i=1}^k \sum_{x_j \in S_i} (x_j - s_i)^2 \quad 2.1$$

The K-means Clustering Algorithm normally entails selecting a random preliminary partition or centers and repeatedly recalculating the centers based upon the partition and then re-computing the partition based on the centers. In large part, the attractiveness of the K-means Algorithm is due to the fact that it is simple and easy to execute. Furthermore, the K-means Algorithm works with any standard norms and it is insensitive to data ordering. However, the K-means Algorithm has some drawbacks among which the result's strong depending on the initial seeds of

centroids. In addition, the correct number of clusters is not obvious, and the resulting clusters can be unbalanced. The basic K-means Algorithm [125] is outlined as below:

- Choose K-data points as the preliminary centroids.
- Re-dispense all points to their nearest centroids.
- Recalculate the centroid of each newly formed cluster.
- Replicate steps 2 and 3 until the centroids do not alter.

When representing data with few clusters, the fine details of the data are lost, but the representation is simpler.

The proposed replica selection decision maker D-system has two different faces to select the best replica. The first one is conducted without the user's intervention while the second face is integrated and guided by the users' preferences.

5.8 D-System First Approach (D-SystemAp1)

In this model the assumption is that the users do not prefer some QoS parameters over others. The final QoS criterion or parameter in this model is a combination of time, site availability and security. This research treats time as the single value obtained based on Equation 4.6. D-SystemAp1 selects the best site location which houses the required replica. In this context, the best site is the one that provides the highest combined security and availability and at the same time the lowest response time possible between the local site and the remote site that houses the required replica. Also, these QoS parameters are almost equal to each other. Henceforth, the term "best replica" has been used to express the highest level of QoS for both the replica and the site that houses this replica.

The QoS parameters set are heterogeneous and conflicting with each other, making the problem quite complex to solve. Therefore, the concepts of K-means Clustering Algorithm as selection factors are used to select the best replica by assuming a model site with a performance of 100% in response time (T), 100% in site

availability (A) and 100% in security (S). This site is so-called centroid. The system tries to find the closest replica to the centroid using the Euclidean distance. The following steps show the procedure of the D-SystemAp1:

Step 1: Receives the requests from the users or typically from the RB.

Step 2: Gathers the replica location information such as physical locations from grid services like RLS.

Step 3: Collects storages speeds, replica size and security information from grid services like RLS and/or sites' log files, replica providers, Grid Manager (GM).

Step 4: Gathers the current criteria values such as network bandwidth and request waiting time in the queue from the NWS, MDS, GridFTP and replica providers.

Step 5: Rates each QoS parameter out of 100%

Step 6: Sends a model replica $MR(T,A,S) = (100,100,100)$ with the collected replicas to K-Means Clustering Algorithm, where k is the number of replicas, to allow the model replica $MR(T,A,S) = (100,100,100)$ to find the replica closest to it to form a cluster. But this approach could lead to joining two collected replicas together and preventing the model replica from forming a cluster with any replica, especially if any two replicas are closer to each other than the model replica and its nearest replica. Hence, this step could be modified by following two steps:

a. Find the distances between the model replica $MR(T, A, S) = (100,100,100)$ and the available number of replicas (n) using Equation 5.6:

$$d_i = \sqrt{(100 - T_{i0})^2 + (100 - A_{i0})^2 + (100 - S_{i0})^2}, i = 1, n \quad 5.6$$

b. Find the shortest distance between the model replica and the available replicas. Since the distance measures heterogeneous data (t,a,s) a new unit of measurements, TAS, is proposed where the lowest value of TAS equals the best available site.

However, up to this point this D-systemAp1 still has some limitations that can be best explained through the illustrations in Figure 3.2, which uses only two parameters

instead of three (for simplicity) while Figure 3.3, illustrates the use of three parameters.

In Figure 3.2, B represents a site that holds the values 60% for time and 60% for security. The above solution gives the same rate for sites A, B, C and D because they have the same distance from the model value. Therefore, any of these rates will be selected randomly. Consequently, the important drawback in such situation lies in that any sites' representative which falls in the diameter of the quarter circle will be chosen randomly for the smallest circle near the model value. The equation of the quarter circle is:

$$(100-X)^2 + (100-Y)^2 = R^2. \quad \text{Where } R \leq X, Y \leq 100 \quad 5.7$$

This can be resolved by rewriting Equation 5.8:

$$(100-X)^2 + (100-Y)^2 = R^2. \quad \text{Where } R \leq X \leq 100, \text{ Min } |X-Y| \quad 5.8$$

However, the fact that site B in Figures 3.2 and 3.3 should be selected because the rates of all QoS parameters are equal, which means it is a balanced and best selection from all QoS parameters points of view. Therefore, a modified distance is required as expressed by the following Equation 5.9:

$$md = \frac{\sqrt{(100-T_0)^2 + (100-A_0)^2 + (100-S_0)^2}}{\sqrt{3}} + \frac{Sd(T_0, A_0, S_0)}{\beta} \quad 5.9$$

The normal distance $d = \frac{\sqrt{(100-T_0)^2 + (100-A_0)^2 + (100-S_0)^2}}{\sqrt{3}}$ did not utilize scaled down standard deviation while md did to increase the distance if the parameters are divers which reduces their chance of being selected.

Step 7: Select the site with smallest md .

Step 8: If availability is less than 50 % discard this site and go to step 5.

Step 9: Utilize other services such as Grid FTP to transfer the replica.

Ultimately the D-SystemAp1 is formulated as multi- objective decision making problem and the last version of mqd equation is denoted by:

$$mqd = \text{Min} \left[\frac{\sqrt{(100-T_{i0})^2 + (100-A_{i0})^2 + (100-S_{i0})^2}}{\sqrt{3}} + \frac{Sd(T_{i0}, A_{i0}, S_{i0})}{\beta} \right]_{i=1}^n \quad 5.9a$$

5.8.1 Case Study

To clarify the research's approach, a scenario of 9 sites has been created as shown in Table 5.3, columns 1 to 4 are the parameters of Equation 4.5. The estimated download time based on Equation 4.5 from sites 1, 2, 3 and 4 are 295s, 249s, 333s and 109s respectively. Site 4 displays the smallest download time so it is rated as a 100% site and site 2 is rated based on Equation 4.6, $\frac{109}{295} \times 100 = 36\%$ while site 3 is rated $\frac{109}{249} \times 100 = 43\%$ and site 3 is rated $\frac{109}{333} \times 100 = 32\%$. As a result, all sites are rated based on the estimated download time to make the selection decision feasible and easier in the next step. On the other hand the remaining operating times for the same sites respectively are 500s, 300s, 70s and 200s respectively. Assuming the value of α is 2, the site availability for site 1, is $\frac{500}{295 \times 2} \times 100 = 84\%$, and the site availability for site 2 is $\frac{300}{249 \times 2} \times 100 = 60\%$. The rest of the calculations are illustrated in Tables 5.3 and 5.4.

However from Table 5.4, column 4, it has been noticed that the D-SystemAp1 has selected the combination in row 4, (100, 100, 44). The problem here lies in that the first two parameters are excellent but the third is below the average and that shows unbalanced solution extremes to some parameters over others. Other example in this regards can be noticed in rows 1 and 2 where D-SystemAp1 overweight the combination (36, 84, 60) over the combination (43, 60, 60) and that is not logical. To overcome this problem the standard deviation is utilized Equation 5.10 and as shown in Table 5.4, the aforementioned drawback is resolved and the best site now is in row 7, (81, 56, 68) the parameters as block are better than row 4. There are no extremely high values or low values for the QoS parameters but somehow the discrepancy is acceptable.

Tables 5.3 & 5.4: 10 GB Replicas with Different Parameters & Sub Parameters'

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	10	150	45	0	295	36	500	84	5	3	7	60	44
2	10	300	156	150	249	43	300	60	3	4	8	60	46
3	10	600	622	300	333	32	70	10	8	8	9		65
4	10	300	156	10	109		300		3	7	1	44	
5	10	600	622	1200	1233	8	2500		9	8	8		53
6	10	150	8	100	1448	7	2000	69	7	6	7	80	57
7	10	600	622	100	133	81	150	56	7	8	2	68	33
8	10	150	45	100	395	27	600	75	2	3	7	48	53
9	10	300	156	200	299	36	150	25	8	1	7	64	60

	1	2	3	4	5	6	7
1	36	84	60	44	24.00	68	48.8
2	43	60	60	46	9.81	55.81	48
3	32	10		65	46.92	111.9	74.4
4			44		32.33	64.33	38.5
5	8			53	53.12	106.1	63.6
6	7	69	80	57	39.36	96.36	64.9
7	81	56	68	33	12.50		35.5
8	27	75	48	53	24.06	77.06	57.8
9	36	25	64	60	20.11	80.11	64

Also the problem of rows 1 and 2 is resolved. Even though utilizing standard deviation has resolved the previous problems, yet, other side effects have appeared which can be noticed in both rows 5 and 6. Row 5 is better than 6 in all parameters but the standard deviation has made it worse therefore the influence of the standard deviation should be reduced or tuned to reasonable values. The study's preliminary experiments have concluded that it should be divided by 5.

These steps are also represented as follows:

1. get R (file requested from the sits)
2. i=1
3. while R found in site
 - 3.1 get t, a, s
 - 3.2
$$d(i) = \frac{\sqrt{(100-t_{i0})^2+(100-a_{i0})^2+(100-s_{i0})^2}}{\sqrt{3}} + \frac{Sd(t_{i0},a_{i0},s_{i0})}{\beta}$$
 - 3.3 i=i+1
4. best= d(1)
6. j=2
5. While i<= j
 - 5.1 if d(j) < best
 - 5.1.1 best = d(j)
6. halt

5.8.2 Simulation Setup

In chapter 2, the advantages of using OptorSim have been introduced, and explained in chapter 3. In this chapter, OptorSim has been used with its default settings. These default settings were copied from the EU data grid parameters. The bandwidth between the two sites is marked in Figure 3.6. In addition, the default OptorSim system workloads' values and parameters' values have been utilized as shown in Table 5.5 (The detailed parameters' values of each site are included in the example folder within OptorSim package. These values represent the real values of the EU data grid). The detailed explanations of the simulation are introduced in chapter 3.

Table 5.5: Workload & System Parameter Values

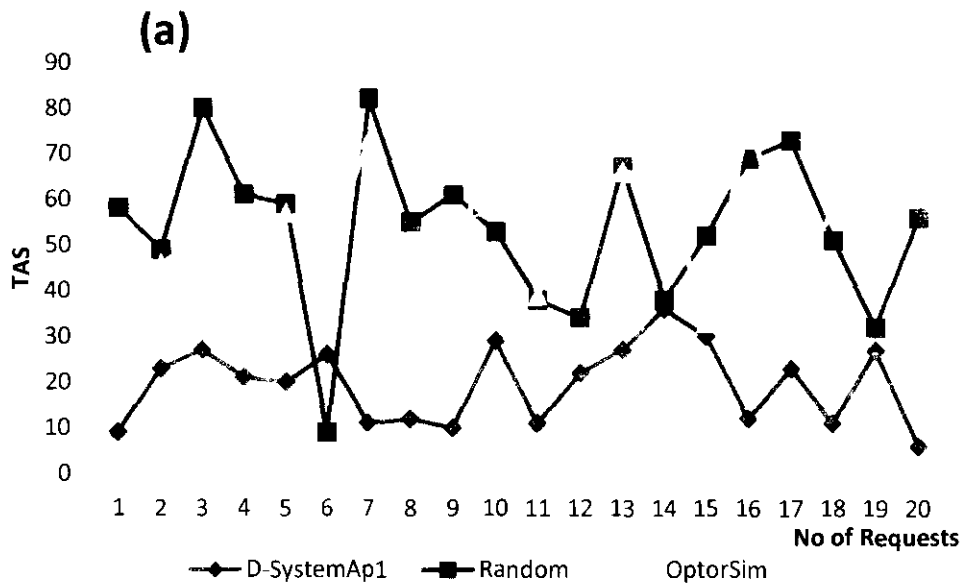
Description	Value
Number of files	200
File size	1 GB
Storage available at an SE	30-100000 GB
Number of files accessed by a job	3-20
α	1
β	5

5.8.3 Performance Metrics

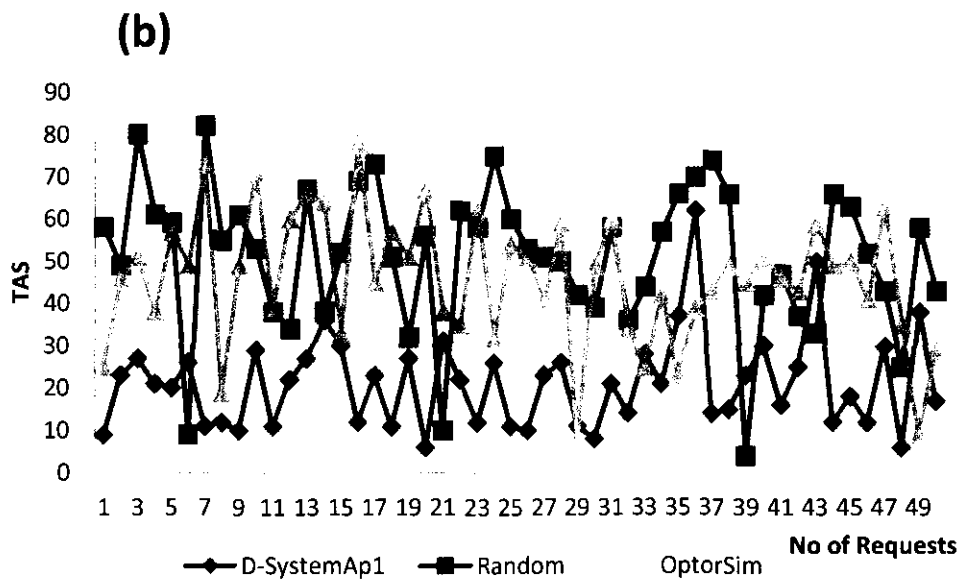
It has been explained in the previous chapter that in a grid environment, users normally send their jobs to the RB, which locates the best site to carry out the jobs. The executed jobs commonly require some data files; the optimizer locates the best locations of the required files. However, the best site here is the one the shows best combination of time, availability and security so, the metric will be called TAS which represents the distance between the model site (100,100,100) and the prospected site (t, a, s). The smallest value of TAS means the shortest distance and the best site.

5.8.4 Results and Discussion

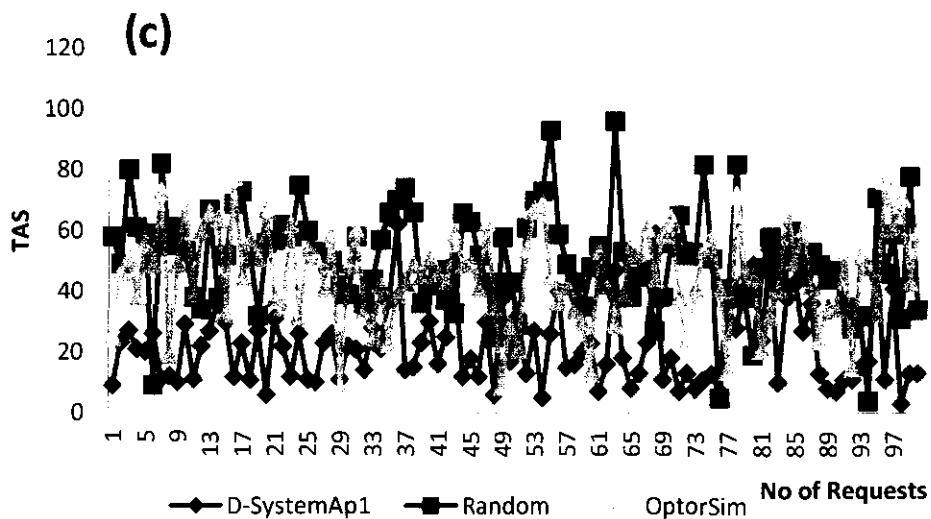
The research's simulations have been carried out for 10, 50 and 100 jobs of various types to examine and evaluate D-SystemAp1. In this regard, availability and security values have been assigned randomly to the locations. The sites have been simulated to evaluate and contrast the outcomes. The same experiments used on D-SystemAp1 have been performed also with the same data using the random system and the OsBi under the LRU replication strategy. The performance of the D-SystemAp1 has showed the best results and the closest to the ideal model node amongst the three systems. Thus, these findings are an added value to previous research findings, by increasing the availability and increasing the security level.



Figures 5.3 (a): Show the Performance of the 3 Systems for 20 Requests



Figures 5.3 (b): Show the Performance of the 3 Systems for 50 Requests



Figures 5.3 (c): Show the Performance of the 3 Systems for 100 Requests

Figure 5.3a shows the performance of the three systems with 20 requests. The results collected from the D-SystemAp1 consistently have been found to be closer to the ideal modal value and displayed a trend to reach it. The ideal modal value is the zero line (the x-axis). The zero line indicates where the time is minimized, the availability value is maximized and the security level is maximized. The D-SystemAp1 selects the best node at any given time as shown in the Figures 5.3 (a,b,c),

where the line represented by the D-SystemAp1 is almost the closest to the x-axis. Any selected site will lose some of its power and the D-SystemAp1 determines the best available one for the next request. Consequently, it allows the other sites to recover or finish processing of some existing jobs and therefore release more system resources.

This ability of the D-SystemAp1 is an added value, in contrast with the other two systems where selection is performed based on the speed only or randomly. If the site selected by the other two systems is overloaded or is not the best site at that particular point in time, then this will lead to a high value of TAS indicating bad performance, while allowing the best site or other better sites to release more system resources and therefore become even better. As a result, the value of TAS becomes very low at some points (even lower than what has been experienced from the D-SystemAp1) and drastically higher at other points. This performance is clearly represented by Figures 5.3 (a, b). The very low values of TAS presented by random and OsBi unfortunately occurred randomly or periodically only in an inefficient or unbalanced manner. In contrast, the stable and managed behavior of the D-SystemAp1 has ensured a higher combined level of security, availability and response time.

Additionally, Figure 5.3b demonstrates 50 requests during which the D-SystemAp1 is still functioning properly, even after increasing the number of requests, in contrast to the random behavior of the other two systems. Moreover, Figure 5.3c shows 100 requests indicating that the D-SystemAp1 successfully has scaled up to a hundred requests. The D-SystemAp1 even has performed better as the number of requests increased. It has been noticed also that the behavior of the OsBi has showed better performance than the random system after increasing the number of requests. This observation is most likely due to its nature of requesting from each site in a predetermined order. However, the D-SystemAp1 still has performed better than both other systems. Figures 5.3 (a, b, c) also show that the D-SystemAp1 is more stable than other systems and trends to move closer to the modal ideal value (the zero line).

5.8.4.1 D-SystemAp1/Statistical Testing

Even though, it is very clear from Figures 5.3 (a, b, c) that D-SystemAp1 overcomes both systems the random and the OsBi. However, a statistical testing is a useful method to achieve a better confirmation concerning the results significance. Therefore, a one-way repeated measure ANOVA has been conducted to compare the three systems D-SystemAp1, the random and OsBi based on TAS measuring metric. The means and the standard deviations are presented in Table 5.6.

Table 5.6: TAS Descriptive Statistics for M-system, Random and OsBi

Systems	Mean	Std. Deviation	N
D-SystemAp1	21.70	12.865	100
Random	51.04	17.003	100
OsBi	45.69	15.637	100

The mean of TAS is 21.70 when utilizing D-SystemAp1; 51.04 when utilizing the random system; and 45.69 when utilizing OsBi. To test whether the difference between the three conditions' mean are significant or not, a multivariate test has been conducted as shown in Table 5.7.

Table 5.7: The Results of Multivariate Tests based on TAS

Effect	Value	F	Hypothesis df	Error df	Sig.	Sig.
Wilks' Lambda	.300	114.394	2.000	98.000	.000	0.7

The results in Table 5.7 show that there is a significant effect on TAS [Wilks' Lambda=.3, $F(2, 98) = 114.394$, $p < .0005$]. Partial eta squared=.7. Although, there is a significant effect on TAS values based on the utilized system especially when using D-SystemAp1. More analyses have been carried out to set the directions of these differences in TAS values. Therefore, tests to shed light on systems effects have been done. The results of the multivariate tests are presented in Table 5.8.

Table 5.8: The Results of Tests Between-Systems Effects based on TAS

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Intercept	467522.163	1	467522.163	1898.301	.000	.950
Error	24382.170	99	246.285			

The one-way repeated-measures ANOVA shows that these TASs are significantly different. $F(1, 99) = 1898.3$, $p < .001$, partial eta squared=.95. Repeated-measures using a Bonferroni adjustment ($\alpha = .05/3 = .017$). Moreover, for pairwise comparisons

as presented in Table 5.9 prove that the TAS values (Euclidean distances) are significantly shorter when utilizing D-SystemAp1. Furthermore, there is a significant reduction in the Euclidean distances in comparing OsBi with the random system. It appears that the random performs better than OsBi while the D-SystemAp1 is the best among all.

Table 5.9: TAS Pairwise Comparisons

(I) factor1		(J) factor1		Mean Difference (I-J)	Sig.a	95% Confidence Interval for Differencea	
						Lower Bound	Upper Bound
dimension1	1	dimension2	2	-29.340-	.000	-33.425-	-25.255-
			3	-23.990-	.000	-28.231-	-19.749-
	2	dimension2	1	-29.340-	.000	25.255	-33.425-
			3	-5.350-	.016	1.012	-9.688-
	3	dimension2	1	-23.990-	.000	19.749	-28.231-
			2	-5.350-	.016	-9.688-	-1.012-

5.8.5 Summary of D-SystemAp1

In summary, the proposed D-SystemAp1, has performed better than both the Random and the OsBi. New factors, namely security and availability, have been considered in the D-SystemAp1. A new performance metric is proposed and so-called TAS. Therefore, the distance in the D-SystemAp1 in replica selection is very close to zero, as shown in the results, indicating enhanced system performance. The D-SystemAp1 can be of much benefit to other grid services that require data selection in data grid environments, utilizing less time and demonstrating high quality performance.

5.9 D-System Second Approach (D-SystemAp2)

In section 1, it has been introduced that incorporating users' preferences is the second approach or the other face of D-system. Utilizing users' preferences to guide the replica selection algorithm has been addressed in [44] but it is limited to disk available space and bandwidth only as shown in Figure 5.4. However, there are other

QoS parameters that can be guided by the users' preferences to increase users' satisfactions such as security, availability and cost.

```
hostname = "comet.xyz.com";
reqdSpace = 5G;
reqdRDBandwidth = 50K/Sec;
rank = other.availableSpace;
requirement = other.availableSpace >
5G && other.MaxRDBandwidth >
50K/Sec;
```

Figure 5.4: Integrating Users' Preferences in Replica Selection System [44]

In fact, each grid user could have his own preferences. Which means that one user may prefer to select the replica location in minimum turnaround time regardless of the other criteria. Other users may prefer the security criterion more than the turnaround time especially if the data is very confidential. While some users' main focus is completing the job safely without interruption even if the job takes more time or prone to security breaches especially if the data is not confidential. In other words, users have different requirements based on their need. The QoS parameters are mostly conflicting and the percentage of imposing each of them is infeasible by the replica selection without user's guidance. For example, a certain user would like his/her replica to be 100% secure and 0% time and other user would like the value of 100% time and 0% secure.

The second approach of D-system allows the users to provide their preferences or priorities, if there is any, and then integrates these preferences into the selection process. In general, all users like to have the maximum values and certainly all of them will ask for 100% for the all parameters. This could hinder managing the replica selection process because the sites varies in their capabilities in which maybe one is (90,40,70) and other is (60,70,40). The solution for such complication is putting some restriction like giving the users equal QoS credit points to use in the selection process. For example 70, 55, 80 points are given to each user. The numbers of QoS credit points are based on sites' average rate of each parameter. This process can achieve a level of equality among users but it will not satisfy them. To achieve both equality and satisfaction the users are allowed to exchange their quotas similar to what

happens in the stock-market place. For example, a user who tends more to security issue can exchange his time credits with a user who tends more to time issue.

What has been presented about D-system first approach is applicable in the second with some minor modifications in the design. Even though the modifications in the design are minors but if they applied properly they would result in significant positive impacts on both the users' satisfactions and the whole grid environment as well.

5.9.1 Stock Market Model

The research's scheme utilizes three different QoS attributes that should be given equally to every user in order to carry out and control the selection. However, the users have different needs and want to control the selection accordingly. Therefore, exchanging these QoS attributes could enhance the selection for the user's satisfaction. The appearance of this process is similar to that which transpires in the stock market. Thus, the stock market model has been adopted with some assumptions.

The stock market is a place for trading company stocks or shares at an agreed price. Usually stakeholders come to the market either to exchange shares of different companies or to purchase shares. In a single transaction, both parties have to give and take the same value, but maybe a different number of shares or money. Both transaction participants are satisfied, believing that this exchange meets their needs therefore, it is the best for them. The model described in this section imitates this process with the following assumptions:

1. Three types of points (shares) are available, namely time, security and availability.
2. All the shares' values are equal which means that one security share can be exchanged with one availability share or one time share.
3. The number of buyers at a single time is equal to the number of sellers having the required share (sites holding the required replica).

4. The number of shares to be given to each buyer prior to the exchange process is equal to the average of the total shares available from the seller of each kind of shares.

In this model, the users have categorized three main single interest groups and many overlapping groups with two or three interests:

1. The first group consists of users who are only concerned with time. They want their replicas in the shortest time without security constraints and they do not want to invest any price to protect unfailing transfers.
2. The second group refers to users who are concerned only with security. They want to receive their data securely at any price since the cost is time to ensure unfailing transfers.
3. The third group refers to users who are concerned only with unfailing transfers. Once the transfer has started, they want it to continue without any disruption whatsoever. They target the most available replica whatever the price may be, either in security or time.
4. The rest are different groups who want to maintain a combination of the above two or three parameters in different portions according to their needs or preferences.

5.9.2 D-SystemAp2 Detailed Procedure

The following steps present the D-SystemAp2 second approach detailed procedure:

Step 1: Collects the requests and preferences from the users or typically from the Resource Broker (RB). The preference values are named the **Ideal Model** values.

Step 2: Gathers the replica location information such as physical locations from grid services like RLS.

Step 3: Collects storages speeds, replica size and security information from grid services like RLS and/or sites' log files, replica providers, GM.

Step 4: Gathers the current criteria values such as network bandwidth and request waiting time in the queue from the NWS, MDS, GridFTP and replica providers.

Step 5: Rates each QoS parameter out of 100%

Step 6: Deploy the users' preferences into the system.

Step 7: Computes and equally divides the site's rank point values among users where the selected number of users is the same as the number of sites that hold the required replica. For example, each user will be given 60 security points, 55 availability points and 72 time points. These QoS points are called **QoS Equal Values (QEV)**. The example supposes that there are only three sites holding the required replica x, y, z each with the following rate x (80, 60, 70), y (60,100,100), z (70, 80, 10) then

$$\text{QEV} = \left(\frac{80+60+70}{3} + \frac{60+100+80}{3} + \frac{70+100+10}{3} \right) = (70,80,60).$$

Step 8: Users start to exchange points among themselves targeting their **Ideal Model** values. Each user attempts to make his QoS equal values as similar as possible to his **Ideal Model** values. The exchange process is performed in a manner that provides all participants the same opportunity of fair exchange. Considering that the number of users involved in the exchange process should be the same as the number of sites that host the same replica. The output is called a **Semi-Ideal Model**. The exchange process is as follows:

- a) Consolidating the users that require the same replica into one group.
- b) Allow the first user to exchange one preference with the second user if possible and continue exchanging consecutively with users until reaching the last user. This step will be repeated starting from the second user with the other users, including the first user. Subsequently, the third will do the same onwards until all users have the same number of chances, then, the exchange process will be repeated from the beginning until no further changes occur.
- c) Computing *Exchange Points* (ε) for each criterion for each user:

$$\varepsilon = \alpha - \beta \tag{5.10}$$

In this equation α represents current points and β represents desired points. From equation (1), the following sub equations are derived:

$$\varepsilon_{\text{Security}} = \alpha_{\text{security}} - \beta_{\text{security}} \quad 5.11$$

$$\varepsilon_{\text{Time}} = \alpha_{\text{Time}} - \beta_{\text{Time}} \quad 5.12$$

$$\varepsilon_{\text{Availability}} = \alpha_{\text{Availability}} - \beta_{\text{Availability}} \quad 5.13$$

If ε is negative, then the user needs points from the underlying criterion, and if ε is positive, this signifies that the user offers these points for exchange.

Step 9: Since the criteria sets are heterogeneous and conflicting with each other, solving the problem becomes quite challenging. Therefore, Euclidean distance is used to choose the best site by assuming an **Ideal Model** site with a performance of X% in response Time (T), Y% in Availability (A) and Z% in Security (S), where X, Y and Z are provided by the user. This location node will be named **Model** to maintain equality even though this node will not be used but the **Semi-Ideal Model** node will be used instead. The system uses Euclidean distance and attempts to find a contiguous replica to the **Semi-Ideal Model** as:

- a) Determine the distance between the **Semi-Ideal Model**

SM (T, A, S) = (X, Y, Z) and the collected replicas using equation 5.14 below:

$$d = \sqrt{(T_1 - T_0)^2 + (A_1 - A_0)^2 + (S_1 - S_0)^2} \quad 5.14$$

The above equation is normalized as shown below to be comparable with others of more or less quality parameters.

$$d = \frac{\sqrt{(T_1 - T_0)^2 + (A_1 - A_0)^2 + (S_1 - S_0)^2}}{\sqrt{30000}} \times 100 \quad 5.15$$

The above formula is generalized as follow:

$$d = \frac{\sqrt{\sum_{k=1}^n (q_{k1} - q_{k0})^2}}{\sqrt{k \times 10000}} \times 100, \text{ and } \sum_{k=1}^n q_{k0} > \sum_{k=1}^n q_{k1} \quad 5.16$$

The equation can be simplified as: $d = \frac{\sqrt{\sum_{k=1}^n (q_{k1} - q_{k0})^2}}{\sqrt{k}}$ 5.17

where q_{k1} is the required quality, q_{k0} is the collected quality and k is the number of quality criteria considered in order to ensure that D-SystemAp2 is targeting the higher quality and not the lower quality and also to ensure that it is measuring in an upward direction and not downward. The following equation should be applied: $\sum_{k=1}^n q_{k0} > \sum_{k=1}^n q_{k1}$, to determine the shortest distance between the Semi-Model and the available replicas. Since the distance measures heterogeneous data (T, A, S), a new unit of measurement, TAS, is recommended where the lowest value of TAS equals the best available replica. TAS will be used specifically for Time, Availability and Security.

b) Utilizes other services such as Grid FTP to transfer the replica with the shortest quality distance from the Semi-Model replica to the grid users securely and tracks history logs for end-to-end transfer data.

5.9.3 Performance Metrics

In a grid environment, users usually furnish their jobs to the RB, which locates the best site to execute them. The jobs under execution usually require some data files. The optimizer searches for the best location of the required files for the jobs.

However, the best in this context consists of three parameters (i.e. Time, Availability and Security) therefore; a new QoS metric is required to measure the performance of this approach. Thus, mathematical calculations are performed to address the three parameters as depicted in Table 5.10 and to be a case study as well. The parameters values are generated randomly.

Table 5.10: Time Metrics Calculation

	1	2	3	4	5	6
1	Time Rated out of 100 %	Availability Rated out of 100 %	Security Rank 100%	Best Replica TAs	Best for preferences (100,100,0)	Best for preferences (0,100,100)
1	36	84	60	44	51	32
2	43	60	60	46	53	41
3	32	10	100	65	87	55
4	100	100	44	25	66	
5	8	100	100	53	78	
6	7	69	80	57	73	21
7	81	56	68	33	48	56
8	85	75	20	49		68
9	36	25	64	60	67	52

TAS is the performance metric used to measure the best site, where the smallest value of TAS signifies the best site, which can be seen in column 4 and row 4. QoS is represented with the distance between the model site and the site under consideration. The integration of user preferences can be observed in the fifth column, where in this case study, the user preferences are (100, 100, 0) as the user does not value security and his main priorities are time and availability. The best value is 20 *tas*, which is better than 32 when the user preferences are not engaged. This proves that the model employed in this research does not only improve quality but also is compliant with user preferences. The same situation is experienced in column 6, as the user preferences are (0,100,100), when the user does not value download speed and is concerned with security and reliability.

Columns five and six prove the importance of integrating user preferences and their impacts on the selection process. What would the result be if the user's real need is (100, 100, 0) and behaved in a greedy way by requesting (100, 100, 100)? This could lead to negative results like 32 *tas* in this case study. However the real value is 25 *tas* which is next to 32 *tas* column 5 and in line with user's preferences but 20 is still better. The stock market model precludes these kinds of acts by limiting the number of points in the user's hand.

D-SystemAp2 imposes user preferences, which biases some quality-parameter over others. In another scenario, if one of the proposed quality-parameters is eliminated, *tas* will become *ta*, *ts* or *as*. Another scenario would be adding a new criterion like read/write data consistency for success rate [164]. Success rate is defined as the count of successfully executed jobs by a grid resource against the entire number of jobs furnished to the resource. In this case *TAS* will be *TASF*, *TASR* or *TASFR*. Since D-SystemAp2 will be applied generally to the Users' Preferred Quality parameters and there are many quality parameters, the unit can be generalized as *UPQ*.

5.9.4 Results and Discussion

The simulations have been carried out with the same setup presented in subsection 5.7.2 after eliminating α because α has no role D-SystemAp2. The number of jobs is 1000 various types to examine and evaluate the proposed User Preferences System (D-SystemAp2) for data replica selection. Users' preferences are generated randomly and used directly. Points exchange has not implemented in the simulation. In this regard, availability and security values have been assigned randomly to the locations. D-SystemAp1 was tested using various data requests. These trials have been first performed with the same data using D-SystemAp1. As displayed in Figure 5.5, a plot of TAS (Time, Availability, and Security) versus the number of requests is illustrated. It is clear that the results gathered from D-SystemAp2 most of the time are better than those gathered from D-SystemAp1 and are closer to the zero line (the X-axis). Hence, the response time is minimized while the availability and security values are maximized in a way consistent with the users' preferences. This indicates that the D-SystemAp2 outperforms D-SystemAp1 and offers the best replica for the majority of data requests.

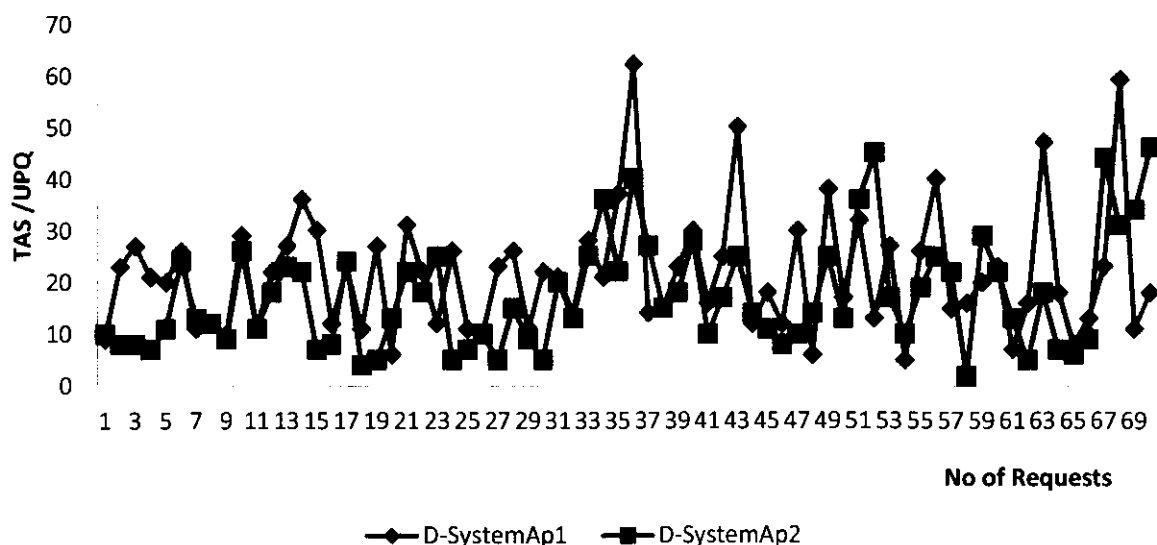


Figure 5.5 D-SystemAp1 Versus D-SystemAp2

If a site has been selected, it will be preoccupied with processing the new job, which leads to an increase in its TAS value due to an increase in response time. In case a site is not selected, the site will be free to complete the job that it is currently processing and this leads to a TAS value decrease.

In order to compare the performance of D-SystemAp2 with the OsBi and the Random algorithm in reliance to the number of replica requests, the same experiments are repeated for different number of jobs. While varying the number of jobs, the number of requests will be changed accordingly. As illustrated in Figure 5.6 (a), it is clear that when the number of requests increases, the TAS for OsBi and the random algorithms become very low in certain cases, even lower than what has been experienced from D-SystemAp2. The lower values for TAS that are achieved by the Random algorithm and OsBi occurred randomly in an inefficient manner. OsBi demonstrates an almost random behaviour in terms of TAS as shown in Figure 5.6 (b) but it is able to balance the user-induced requests across the available replicas better than the Random algorithm because it considers time, which is one component of the QoS parameters. However, since the OsBi chooses replicas without assessing their ability to meet the user's QoS specification, it is difficult to say conclusively whether the replicas chosen by the OsBi will be able to meet the user's requested response time, availability and security requirements. For example, if a user has requested a

60% response time, 40% availability, and 90% security, which corresponds to a TAS (60, 40, 90) = 42, the OsBi may choose a replica location that corresponds to a TAS (80, 50, 40) = 47 which satisfies the TAS requirement but fails to meet the user's QoS specification.

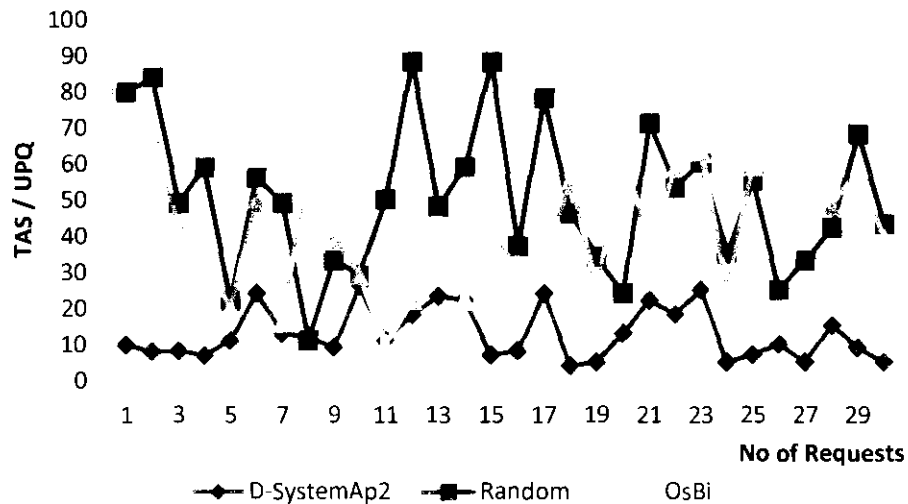


Figure 5.6(a) : Random & OsBi Versus D-SystemAp2

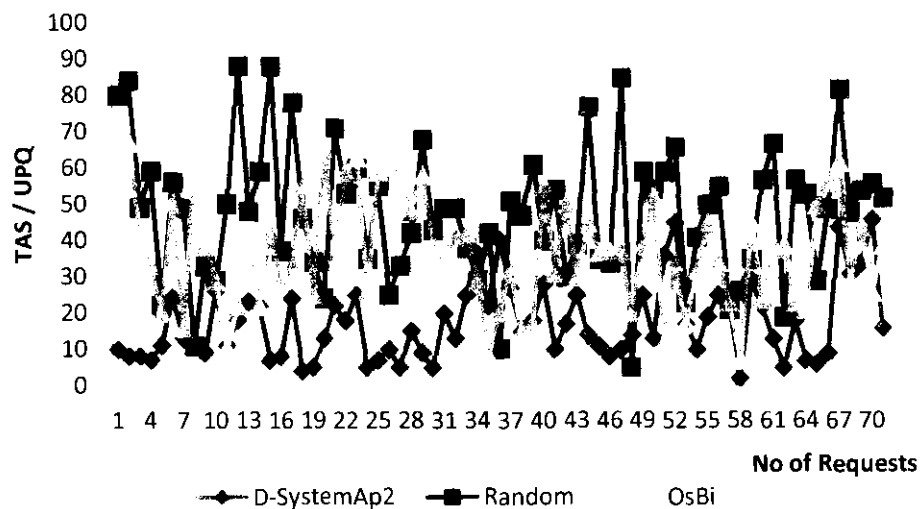


Figure 5.6 (b) : Random & OsBi & D-SystemAp2 with a larger Requests

Up to this point, D-SystemAp2 has been examined based on 3 QoS parameters only. Furthermore, to investigate the performance of D-SystemAp2 based on different

number of QoS parameters (k), the same experiments are repeated as above while varying the values of k . The average UPQ values for 30, 100, 700 requests under D-SystemAp2, OsBi and the random are shown in Figures 5.7 (a, b, c) while varying k values from 2 to 5. As apparent from Figures 5.7 (a, b, c), the D-SystemAp2 outperforms the OsBi and the random in all cases. Hence, adding or dropping QoS parameters from D-SystemAp2 provides good results and only minor changes to TAS (UPQ) occur. This proves that D-SystemAp2 is applicable regardless of whether the number of QoS parameters to be considered is increased or decreased.

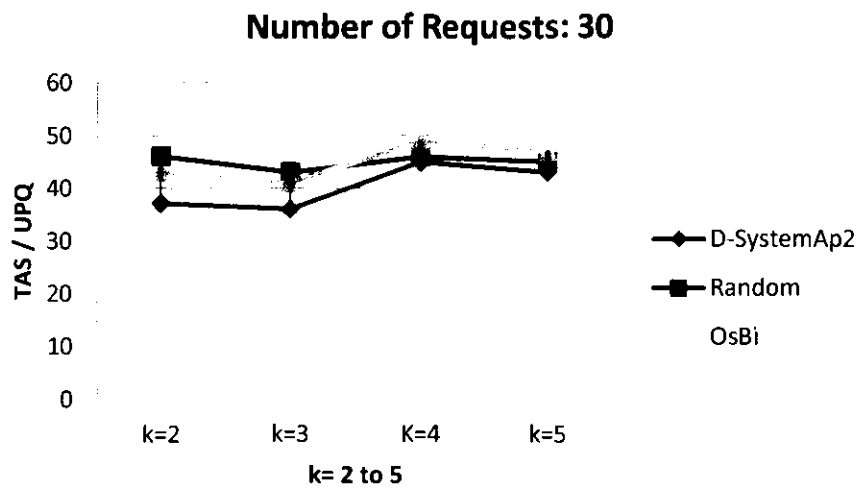


Figure 5.7 (a) D-SystemAp2, OsBi & the Random for 30 Requests, $k=2-5$

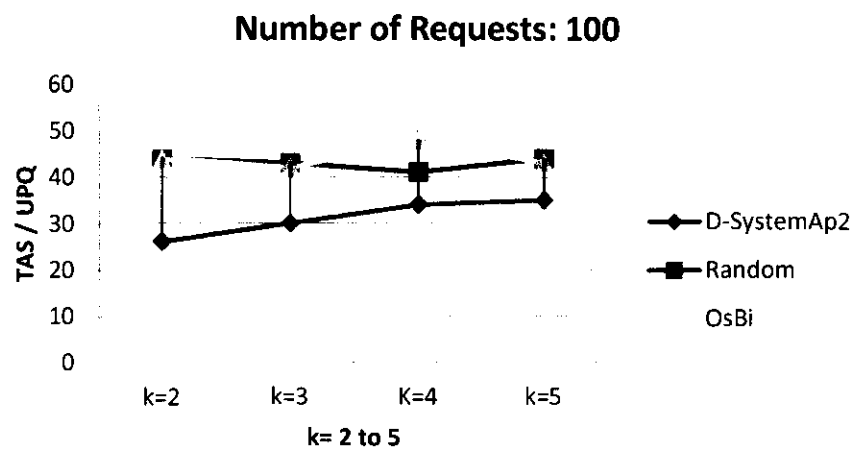


Figure 5.7 (b): D-SystemAp2, OsBi & the Random for 100 Requests, $k=2-5$

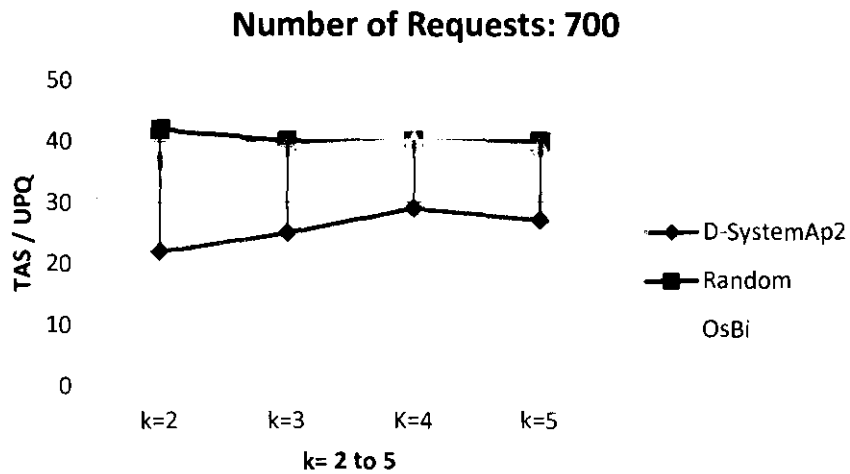


Figure 5.7 (c): D-SystemAp2, OsBi & the Random for 700 Requests, k=2-5

Table 5.11 shows the average value of UPQ after 150 requests have been carried out by both algorithms. For D-SystemAp2, it has been noticed that while increasing k, UPQ slightly increased until k=5 is reached whereupon UPQ decreased. This demonstrates that altering k's value does not affect D-SystemAp2 performance, and the UPQ collected values are more correlated to the resources and user preferences as well. On the other hand, OsBi behaves the same despite k. The minor differences in UPQ values that are shown in Table 4.7 are due to different grid conditions and different user preferences.

Table 5.11: Average Values of UPQ after 150 Requests for Values of K from 2-5

Number of QoS parameters	K=2	K=3	K=4	K=5
D-SystemAp2	24	26	26	29
OsBi	38	39	39	40

5.9.5 Summary of D-SystemAp2

In this section, a data grid model has been presented that considers user preferences to select the best replica from a number of sites that hold replicas. In order to select the best replica, a user preferences system (D-SystemAp2) has been developed and designed to meet the user's QoS specifications (i.e. response time, availability and security). Simulation results prove that D-SystemAp2 outperforms D-SystemAp1, Random algorithm and OsBi. Furthermore, by varying the number of QoS parameters k , for similar requests and using D-SystemAp2 random and the OsBi, it was found that D-SystemAp2 outperforms the other systems in all scenarios. Even though, D-System shows a very high performance replica selection process but it handles users' requests one by one in a greedy way without considering the other users in the queue. This could lead to satisfy the users to the front of the scheduling queue at the cost of those further to the back, resulting in an unfair distribution of users' satisfaction. In the upcoming chapter, the limitation experienced in D-System will be addressed by proposing a technique archives fair users' satisfaction.

CHAPTER 6

MULTI USERS SELECTION & FAIR USERS' SATISFACTION

6.1 Overview

This chapter introduces the theoretical background of M-system. Then, the problem is formulated. Next to that, the chapter clarifies the role of GA in permutation problems. After that, M-system is mathematically modeled. Then, M-system's functional and non-functional requirements are presented. Finally, the performance metrics and the simulation results are discussed.

6.2 Theoretical Background

In chapter 4, the importance of site availability QoS parameter has been discussed, validated and new replica selection system has been introduced. This model, entitled as A-system, has been described as basic QoS replica selection system. Simulation results have been proved as significant in terms of decreasing jobs' turnaround time. In chapter 5, the importance of both security QoS parameter and users' preference in the selection process have been presented. To integrate security and users' preferences in the replica selection process A-system has been extended to be D-system in which a new direction in the replica selection has been introduced by giving attention to QoS parameters that have different influences like time or/ and other than time. Even though A-system and D-system show a significant advancement in the replica selection optimization techniques, yet, they still require more enhancements due to two reasons. The first reason is related to cost as QoS parameter is still not addressed yet. The second reason is that the D-system is still weak in achieving almost equal

satisfactions amongst grid users. It is reported in [40] that in grid the executed jobs are subject to failures because of the infected hardware, software vulnerability, network failure, overloaded resources and non-availability of required resource. Furthermore, the same data set may have location-dependent access costs, just like tangible goods in actual economies [41]. According to [42], some users attempt to optimize the mapping to the required replica based on the billing costs of their network operators or hosting services, especially if a 95th-percentile billing mechanisms are imposed for the services. Cost minimization can be achieved by decreasing the frequency of peak consumption, or by not exceeding their budgeted rates.

In previous studies the replica selection systems tried to satisfy the user request in a greedy [42, 45-48, 165, 166] way, or in a fair way [113] based on the history information. These approaches are carried out from the scheduling queue one by one in a FIFO manner. This could more satisfy the users at the front of the scheduling queue at the expense of those queued at the back, resulting in an unfair users' satisfaction. A fair satisfaction in the context of this research implies an almost equal level of QoS fulfillment for all users in the scheduling queue. As an example, if a user who pays \$10 gets back a service worth only \$9 (90% satisfaction), then a user who pays \$1000 should be getting a service worth \$900. In other words, the equality of the percentage of all users' satisfaction is highly demanded.

In this chapter, the replica selection has been presented as a multi-criteria problem. The global optimization system is proposed to attain optimal efficiency and fair satisfaction for all the grid users. The main contributions of this chapter are to give a detailed formulation of the problem, the design and the application of a hybrid GA-based solution. The main objective of this chapter is achieved by performing more optimization techniques and by considering more QoS parameters in line with users' preferences in order to improve the replica selection system performance. The main purpose of optimization techniques is to increase the efficiency of the systems and equality in terms of users' satisfaction (fair users' satisfaction). To achieve fair users' satisfactions, the users' preferences and the available solutions should be carefully analyzed to produce the most suitable pair matches. In turn, this brings up a huge search space which requires a solid technique to address. In this research GA is

adopted as a concrete technique to reduce the complexity of the search space. Ultimately improving the replica selection systems in data grids reflects positively the whole grid infrastructure. This chapter offers some alternative solutions to the research problem, encapsulated in replica selection system termed as: Multi-users Replica Selection System for fair users' satisfaction and is abbreviated to M-system. The proposed system hybrids the GA and the D-system.

The remaining part of this chapter is organized as follows: Section 2 exhibits the problem formulation for a better understanding with examples. Section 3 focuses on the importance of Genetic Algorithm as a proposed solution. Section 4 is the mathematical model of the problem. Section 5 is the system requirements and design. Section 6 presents the performance metrics and evaluation. Section 7 includes the results and discussion. Lastly, Section 8 presents the summary of the chapter.

6.3 Problem Formulation

The proposed model involves pairing users with replica locations in which each user should be assigned to only one location. Each location has its own specification and each user has specific preferences as well. The assignment should be as fairly as possible in satisfying all users in a single stage (scheduling cycle). For example, suppose that there are four sites with different QoS rates and four users with different QoS requirements. To make it simple, one QoS parameter, namely replica transfer speed, is utilized. As shown in Table 6.1, the first user requested a transfer speed rated at 80% meanwhile the best site at that time is rated at 80%. Such matching results in 100% satisfaction. However, this pairing degrades Site1 rate due to consuming amount of the bandwidth and slices of storage system. Subsequently, the second user was only granted 77% satisfaction due to the fact that the second best site rate is 70% while the requester is targeting 90% rate. The same scenarios were also experienced by requests 3 and 4.

Table 6.1: Pairing Requests to Sites in an Arbitrary Order

Site No.	Site 1	Site 2	Site 3	Site 4
Sites Rate	80	70	65	50
Request No.	1	2	3	4
Requested rate	80	90	60	70
Requester Satisfaction	100%	77%	100%	71%

The example above shows unfair satisfying resource allocations despite the relatively high level of satisfaction. The satisfaction average is $(100 + 77 + 100 + 71) / 4 = 87\%$ whereas the standard deviation of such allocation was 0.13 which is relatively an unfair satisfaction as it means very high divers of users' satisfactions. However, reordering the requests in the queue to select the best replica can increase both the level of satisfaction and the QoS fairness resources allocation as well.

In Table 6.2, the requests are reordered before the matching process. If the scheduler starts with 90% requester and the best available site is rated 80%, the result of satisfaction will be 88.87%. Consequently, Request 2, which is targeting 80% rate, will be granted the 70% rated site resulting in 87.5% satisfaction. The results of the third and the fourth requests are illustrated in Table 6.2.

Table 6.2: Pairing Requests to Sites in a Managed Order

Site No.	Site 1	Site 2	Site 3	Site 4
Sites Rate	80	70	65	50
Request No.	1	2	3	4
Requested rate	90	80	70	60
Requester Satisfaction	88.87%	87.5%	92.85%	83.33%

Reordering selection approach shows slightly better average of satisfaction $(88.87 + 87.5 + 92.85 + 83.33) / 4 = 88.14\%$ and exhibits an enormously better standard deviation of 0.028 satisfaction. This indicates an improvement of the fair satisfactions because it refers to very low diverse of users' satisfactions. The above mentioned example justifies that reordering the requests to make a decision that gratifies, to a reasonable extent, all the requesters, would bring a better global fair satisfaction that caters all the users in a well-adjusted manner. The technique of using one parameter is not difficult, but considering 4 parameters becomes very challenging. For example if there are three sites and three requests that means that there are six different solutions.

The problem mathematically becomes permutational without repetition, i.e. $[n!/(n-r)!]$. However, each solution yields in different total values of UPQs and different standard deviation of the UPQs of all users. Table 6.3 illustrates the result of each permutation.

Table 6.3: Different Pairing Requests to Sites

Request s No.	Site No.	Site 1	Site 2	Site 3	Total UPQ	Standard Deviation	Total UPQ+SD
	Sites Rate	70,60,40,60	65,70,60,55	55,65,60,40			
1-	Firs order	User 1 70,70,60,60	User 2 75,75,80,60	User 3 80,60,60,70	67	14.57	81.57
	UPQs	7	36	24			
2-	Second order	User 1	User 3	User 2	67	14.57	81.57
	UPQs	7	24	36			
3-	Third order	User 2	User 1	User 3	84	9.54	93.54
	UPQs	23	22	39			
4-	Forth order	User 2	User 3	User 1	72	1	73
	UPQs	23	24	25			
5-	Fifth order	User 3	User 1	User 2	81	7.81	88.81
	UPQs	23	36	25			
6-	Sixth order	User 3	User 2	User 1	81	7.81	88.81
	UPQs	23	25	36			

In real life situation, the numbers of sites are thousands among which only tens are selected each time to introduce a very large search space. For example, to select 10 out of 1000 sites, the total permutations are 8.26×10^{59} .

6.4 Permutation & Genetic Algorithm

Evolutionary algorithms are usually utilized in optimization problems. For example authors of [167] have used the genetic algorithm to sort out the QoS-based selection. However, the genetic algorithm is a mean of stochastic optimization specifically very useful for the permutation approaches as reported in [168] due to the following advantages:

- As the well-known advantage of stochastic optimization, genetic algorithm is not prone to stuck into local optima if its attributes are prepared properly. This is particularly very useful for permutation correction as the all criteria have strong local minima.
- Genetic algorithms demonstrate faster convergence in comparison to other stochastic optimization algorithms, specifically for the problems with a wide-dimensionally space [169].
- Genetic algorithms are significantly very suitable for discrete optimization because they naturally utilize binary series to present the solution.
- It is an efficient algorithm for multi-objective genetic optimization that enables the whole multi-objective optimum solutions set to be evolved in parallel [170].

6.5 Mathematical Modeling

The parameters of the model, objective functions and decision variables are defined as follow:

6.5.1 Parameters

Let $Z = \{g_1, g_2, g_3, \dots, g_n\}$, be a set of data grid sites and $U = \{u_1, u_2, u_3, \dots, u_m\}$, is a set of users Such that $m \leq n$. $u_i, i = 1, 2, 3, \dots, m$ and $g_j, j = 1, 2, 3, \dots, n$ each site and user has four parameters (T, A, S , and C) the values of each parameter lies between 0 and 100.

Note: Each parameter value for grid site represents its rate based on its performance and for a user, it represents its preferences.

6.5.2 Decision variables

Each user u_i is assigned to one grid site g_j , the assignment is denoted by

$$R_{ij} \in \{0, 1\}, i = 1, 2, \dots, m, j = 1, 2, \dots, n \text{ or } R_{ij} = \begin{cases} 1 & \text{if the } i\text{th user assigned to the } j\text{th grid site,} \\ 0, & \text{otherwise.} \end{cases} \quad 6.1$$

To guarantee that each user u_i is assigned to only one grid site g_j , the following summation should be satisfied:

$$\sum_{j=1}^n R_{ij} = 1, \quad i = 1, 2, 3, \dots, m \quad 6.2$$

6.5.3 Objective functions

Satisfaction in the model employed in this study is measured by the Euclidean distance between user i and grid site j and denoted by:

$$d(u_i, g_j) = \sqrt{(T_{u_i} - T_{g_j})^2 + (A_{u_i} - A_{g_j})^2 + (S_{u_i} - S_{g_j})^2 + (C_{u_i} - C_{g_j})^2}, \text{ The shortest distance is the best} \quad 6.3$$

Hence, the objective functions are created by maximizing the levels of preference for each user in the required grid site. Preferences maximization is conveyed by minimizing the Euclidean distance between each user i and its assigned grid site j . But minimizing one user's distance (increasing satisfaction) may lead to increase other(s)' distances (decreasing their satisfactions). Therefore, the solution should be a tradeoff between users' satisfactions.

$$\text{Let } \sum_{i=1}^m d(u_i, g_j), j = 1, 2, \dots, n \text{ be a general vector of all decision variables} \quad 6.4$$

The following objectives are needed:

- Minimize the total Euclidean distances among users and the grid sites:

$$\text{Min} \left(\sum_{i=1}^m \sum_{j=1}^n R_{ij} \cdot d(u_i, g_j) \right) \quad 6.5$$

- Minimize the standard deviation among the users' distances. This means that the users will get almost the same distances which grantee equal users' satisfactions.

$$\text{Min} \left(SD \left(\sum_{i=1}^m \sum_{j=1}^n R_{ij} \cdot d(u_i, g_j) \right) \right) \quad 6.6$$

Subject to

$$R_{ij} \in \{0, 1\} \quad , i = 1, 2, \dots, m, j = 1, 2, \dots, n$$

$$\sum_{j=1}^n R_{ij} = 1, \quad j = 1, 2, 3, \dots, m$$

Therefore, the study's objective is to minimize both the total distances among the users' preferences and the grid sites' rates and at the same time, the standard deviation among users' distances. This means that the research problem is a multi-objective decision making with two parameters.

6.5.4 Scalarization

Scalarization: is consolidating various objectives into a one objective in a manner that repetitively sorting out the single objective optimization problem with different parameters. This can allow the researchers to obtain all effective solutions for the preliminary multi-objective problem. Many scalarization methods have been established; see [133, 171-173]. Therefore, scalarization will be adopted to be the best solution for the research problem which is denoted by:

$$\text{Min} \left(\alpha \cdot SD \left(\sum_{i=1}^m \sum_{j=1}^n R_{ij} \cdot d(u_i, g_j) \right) + \beta \cdot \sum_{i=1}^m \sum_{j=1}^n R_{ij} \cdot d(u_i, g_j) \right) \quad 6.7$$

α and β are used by the data grid administrator to scale up (or down) the standard deviation or the total distances value based on the experienced observations and their affects. Due to the multi-objective nature of this research decision making design, the solution procedure has been implemented in two steps:

1. Scalarizing the problem as done above, and
2. Sorting out the scalarized problem.

6.6 M-system Requirements and Design

The main functional and the none-functional requirements of M-system are similar to D-system which have been presented in section 5.2. In replica selection functionality, the M-system selects the best replica locations from many replicas distributed across the grid sites. In this chapter, the best replica location embodies three meanings. The first one refers to the site location that houses the required replica which is capable of delivering the underlying replica in minimum turnaround time and high level of QoS because the grid sites vary in their capabilities. The second meaning refers to the replica location that is more with accordance to the user's preferences as different users have different preferences and even a single user could change his preferences based on the replica content or based on his commitments. The third meaning refers to the replica locations that achieve nearly equal fair users' satisfactions. M-system hybridizes D-system with GA to solve the complexity of the selection problem. The complexity of the problem is represented by the existence of heterogeneous values in the selection parameters. These parameters usually are in conflict with each other. On the other hand, each user has his own preferences, and the search space is very wide.

The main focus of the proposed work is the replica selection decision and the formation of fair users' satisfaction. The main resources entitled in this research are data files required for job execution. Meanwhile, each dataset is duplicated to a decided number of copies titled replicas. Several replicas of each data file are distributed to different grid sites. Rating a grid site is based on its own QoS parameters which are, in this research, represented by time response, security, availability and cost. These parameters play the role of allocating resources to users and are the factors of the selection process because each user has his/her own preferences in this regard. Thus, the selection system decides which site is the best to satisfy the user's preferences. Rating time, availability and security QoS parameters have been presented in the previous chapters while cost parameter is described below:

6.6.1 Rating Cost

Replicas of the same data set are stored on different data centers. The access costs are different on different data centers, just like the economic phenomenon in real world [41]. Rating a replica based on its price is denoted by the following equation:

$$c_i = \frac{\min\{c_i\}_{i=1,n}}{c_i} \times 100 \quad 6.8$$

The criteria set are not homogenous. Collecting them in one cluster to get one value is not feasible. In addition to that, each user prefers a ration of each criterion. The search space is very wide with the objective of achieving fair users' satisfaction. These reasons show the difficulty of the problem and how a complex decision is required. Therefore, a solid technique for the decision process GA [126] has been chosen to be integrated in D-system as the most suitable technique for this problem.

6.6.2 Genetic Algorithm

Genetic Algorithms are classes of evolutionary, random, adaptive, algorithms that include optimization and search. Genetic Algorithm was inspired by [126]. The elementary notion is to imitate the manner of natural evolution, to generate artificial procedures for a "clever" algorithm that is able to obtain the solution of complicated problems such as scheduling jobs in grid. GA preserves a pool of feasible solutions to a problem called population. It is presented as chromosomes depending on the representation structure [174]. During iterations, the fitness value of each individual in the population is evaluated. After that, reproduction operators and selections are implemented to produce a new population, which is utilized again in the following iteration of the GA. The performance of GAs is susceptible to population size. In general, a small population size increases the likelihood of premature convergence due to the loss of niche, while a large population size leads to a high computation cost. Reproduction consists of crossover and mutation operators. The entire procedure is reiterated a number of times and it is called generations. Typical GA main's structure is described in Algorithm 6.1.

Algorithm 6.1

```

Create Initial Population:

Evaluation ( );

while (stopping condition not met){

Selection ( );

Crossover ( );

Mutation ( );

Evaluation ( ); }

Return best solution:

```

Genetic Algorithms constitute two main components, i.e., the encoding schema and the evaluation function, known as fitness function. In this research, the chromosome is presented as a matrix of integer numbers. The places of elements show the users who are requesting replicas while the value of every element specifies the site id that holds the replica assigned to the user. For example, the tenth element (entitled gene) of the chromosome presented in Figure 6.1 denotes the tenth user in the queue assigned to site 6. In order to measure the value or the quality of a solution the fitness function is implemented. Every chromosome is associated with its fitness value. In this research, the fitness function is a multi-criterion, as denoted in Equation 6.7. Euclidean distance between QoS specifications desired site by the user and the QoS specifications site assigned to him is the first criterion, which should be minimized. The second one is the standard deviation of users' Euclidean distances. It has been assumed that both criteria are equally weighted ($\alpha = \beta = 1$).

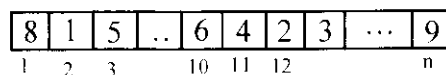


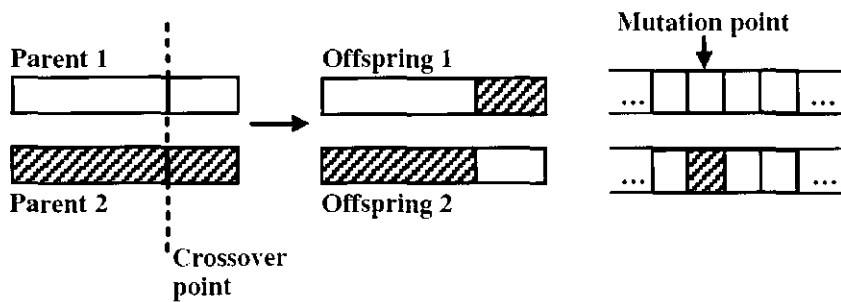
Figure 6.1 Chromosome Encoding

6.6.3 Selection of Operator

The selections of operators that are explained in this section have a significant role to exploit the benefits of the GA. A selected operator decides which individuals will be reproduced for the next generation aiming at disregarding or replacing the poor solutions with a predefined probability (the standard probability value is 70% [175, 176]) to produce the new offspring, as depicted in Figure 6.2 (a) At this stage of this research the simplest form of crossover is adopted in which the crossover point is in the middle of the chromosome. In future work different forms crossover points will be experimented.

6.6.4 Mutation Operator

The mutation operator randomly changes the integer (site number) of the chromosome. This process is conducted with a very small probability (e.g. 0.05% [175, 176]), as shown in Figure 6.2 (b) to keep the diversity in the chromosomes' population and to overcome the local optima in the search space. The Genetic Algorithm is terminated by either producing a predefined number of generations, or converging the fitness of the population individuals. The result of GA is the fittest chromosome of the produced populations.



Figures 6.2 (a) One Point Crossover, (b) Mutation

6.6.5 The Repair Operator

In highly constrained optimization problems, the crossover and mutation operators generally produce illegal or infeasible solutions, therefore, there is a production of a

waste-time search. For example the constraint in the research's problem is that each site is to be selected by one user in each scheduling cycle, as illustrated in Figure 6.3 which represents 30 users paired with different site. Figure 6.4 depicts the content of Figure 6.3 after mutation in which duplication experienced by users between 1 and 11.

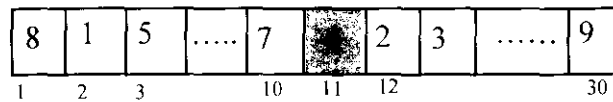


Figure 6.3 Pairing Users with Sites before Mutation

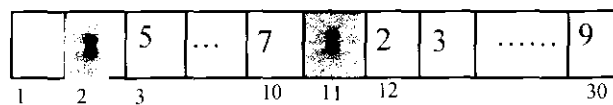


Figure 6.4 Pairing Users with Sites after Mutation

This problem can be solved by incorporating problem-specific knowledge. Problem-specific knowledge can be used either to prevent the genetic operators from producing infeasible solutions or to repair them [67]. A combinatorial problem like replica selection place constrains the produced solutions to guide the GA during the search process. In replica selection problem, solutions are only valid when all M requests in the problem are assigned in the solution. Furthermore, a solution is only considered valid if all sites are selected at most once in the solution, and no site is selected more than once. Thus, a fixed-length chromosome has been used to represent the assignment process. These constraints act as a trigger for the application of the repair operator. The GA with repair operator can be summarized as follows:

- Generate the initial population $P(0)$ at random and set $i = 0$;
- Evaluate the fitness of each individual in $P(i)$;
- Select parents from $P(i)$ based on their fitness.
- Apply standard crossover
- Apply standard mutation.

- Apply Repair Operator.
- Repeat until convergence or reaching the number of generations.

In the repair mechanism, each duplicated site number in each chromosome is identified. After that, the repair mechanism replaced each of them with a new randomly selected site from the available site pool with the condition that the selected site is not already presented in the chromosome.

6.6.6 Hybridizing D-system with GA

Though GA is reputed to be slow, it has actually been utilized in real time applications like scheduling in grid computing [177]. The key is to merge greedy algorithm with GA. The role of the greedy algorithm is to fill up the initial population in order to decrease convergence time. Similarly, in this research several runs of D-system are carried out to fill up the initial population and the results in terms of time convergence are promising in comparison to the results gained from solely GA.

6.6.7 System Detailed Design

The architecture of the data grid services is divided into two levels as shown in Figure 2.3 the upper level includes the high-level services that utilize the low-level or core services. Replica selection optimization technique is of high-level services so it invokes a number of core services. Information about an individual resource or a set of resources is collected and maintained by a grid Resource Information Service (GRIS) daemon [178]. GRIS is designed to gather and announce system configuration metadata describing that storage system. For example, each storage resource in the Globus data grid [44] incorporates a GRIS to circulate its information. Typically, GRIS informs about attributes like storage capacity, seek times, and description of site-specific policies governing storage system usage. Some attributes are dynamic and vary with several frequencies such as total space, the available space, queue waiting time and mount point. Others are static such as disk Transfer Rate.

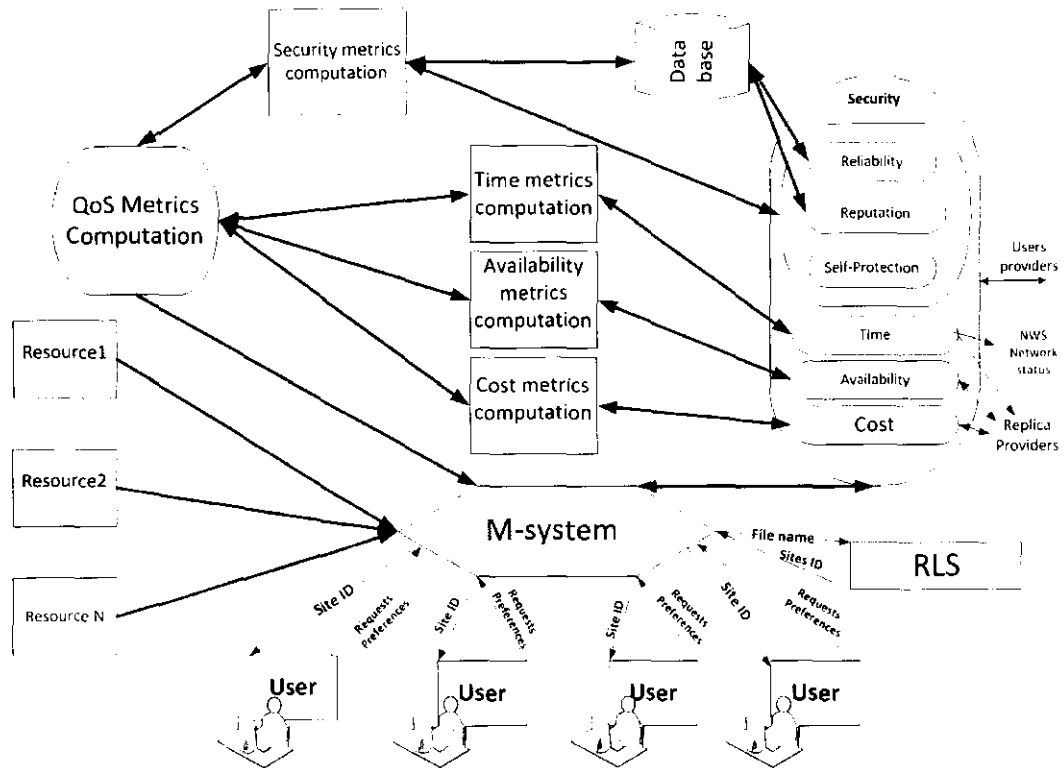


Figure 6.5 Major Components and Structure of M-system

M-system, as illustrated in Figure 6.5, is functioned by receiving the users' request via the grid Resource Broker (RB). RB then retrieves related physical file's names and locations from the RLS. Subsequently, M-system receives information about the sites which hold the replicas and their network status from the GRIS such as: NWS [105], Meta-computing Directory Service (MDS) [179] and Grid File Transfer Protocol (GridFTP) [179]. Also, M-system receives security rates for each replica location from the GM and receives availability and cost information from the log files of each replica. Afterward, rating each replica location is performed. Then, M-system pairs the users' requests with the best replica location in a way that maintains fair users' satisfaction. So the replica for the concerned user's job is chosen. Hence, the new high-level service replica selection system is an optimization approach.

The exact sequence of steps in M-system is as follows:

- Collects the requested replica and the users' preferred QoS from the resource broker.
- Collects replica of physical file names and locations from Replica Location Service
- Collects sites' operating hours and access cost for each location from their log files.
- Collects sites' security rating for each location from the GM.
- Collects sites' current criteria values like bandwidth from the information service providers for instance GridFTP, NWS, and MDS.
- Rate each location in terms of time, availability, security and cost using the relevant equations.
- Utilizes D-system through several runs to generate the initial population.
- Utilizes GA to Pair each user request with preferred replica location in a way that achieves fair users' satisfaction.

6.7 Performance Metrics and Evaluation

M-system performance is evaluated by means of calculating, anatomizing and comparing its outputs with other systems. Thus, two new metrics are proposed to evaluate and reflect the performance of M-system and these are: Average User's Satisfaction and Fair User's Satisfaction.

6.7.1 Total User's Satisfaction

User's satisfaction criterion is very important particularly because one of M-system main objectives is to increase users' satisfaction as much as possible. In the proposed model, user's satisfaction is measured by the distance between the QoS

preferred by the user and the actual QoS already assigned to that user. As a result, the metric used to measure M-system is user's preferred QoS (UPQ). The smallest value of UPQ means the best M-system performance. UPQ level for any userⁱ is calculated as follow:

$$UPQ = \sqrt{(T_u - T_g)^2 + (A_u - A_g)^2 + (S_u - S_g)^2 + (C_u - C_g)^2} \quad 6.8$$

While the Total UPQ (TUPQ) for all users is denoted by:

$$TUPQ = \sum_{i=1}^n UPQ \quad 6.9$$

Also, the average of UPQ for the users is another metric which is obtained by the following equation:

$$\overline{UPQ} = \frac{\sum_{i=1}^n \sqrt{(T_u - T_g)^2 + (A_u - A_g)^2 + (S_u - S_g)^2 + (C_u - C_g)^2}}{n} \quad 6.10$$

6.7.2 Fair Users' Satisfaction

Fair Users' Satisfaction (FUS), as metric, measures the diverse level of QoS distributed to users. The preferred QoS (UPQ) gained by the users should be fair as much as possible. The discrepancy of the UPQ values the user gains should be reduced in an attempt to attain fairness among users. Since UPQ is the proposed criterion, therefore, the standard deviation (SD) metric is the favorable one to calculate fairness level. Fair users' level of satisfaction is calculated as follow:

$$FUS = SD(UPQ_1, UPQ_2, \dots, UPQ_n) \quad 6.11$$

The smaller value of FUS means the better performance of M-system in terms of fair users' satisfaction.

6.7.3 Evaluation

The literature review confirmed that there is no previous work in the context of replica selection in data grids considers simultaneous multi-users replica selection

decision making. It is also confirmed from the literature that the D-system (chapter 5) and the AHP [113] are the most similar systems to M-system, yet there are two main differences between M-system and D-system which can be summarized as follow: First, the D-system does not integrate cost. Second, it pairs users to their preferred sites one by one (no global consideration). Similarly the main differences between M-system and AHP are as follow: First, the AHP does not integrate cost as well. Second, it pairs users with the sites one by one. Third AHP does not consider users' preferences instead it rely on the history information.

In this chapter, D-system has been extended to integrate cost. AHP also has been extended to integrate cost and users' preferences and both of D-system and AHP are compared with M-system. All AHP, M-system and the D-system have been utilized to measure values brought by simulation. The assumption of the simulation can be clarified as:

- The chosen proxy server should always be up.
- The network is always reliable among all sites.
- The parameters' values are rated between (0 to 100). This is in line with the reality (any other range can be easily integrated in the proposed system).

6.8 Results and Discussion

In this section the experiments will be based on two cases. The first case will conclude the performance of M-system in comparison with D-system and AHP. The second case examines the scalability of M-system. The simulation setup is presented in Table 6.4.

Table 6.4: Experiment Setup

No of Users	10-70
No of sites that holds the replica	20-200
Population size	No of Users
Offspring Producing Probability	70%
Mutation Probability	0.05 %
Crossover	Uniform
No of Generations	No of Users × No of sites

6.8.1 Case (1) Fair Users' Satisfactions & the Total UPQ

The first step is called "Prior to Running Simulations of M-system, AHP and D-system". 20 grid sites have been assumed and each site has 4 QoS parameters with a value rated between (0 to 100) for each. These values are generated randomly as shown in Table 6.5. On the other hand, it is assumed that 10 users independently will request one replica according to their preferred QoS level for each parameter. These values are created randomly as shown in Table 6.6.

Table 6.5: Sites with their QoS Parameters' Values

Site no	T	A	S	C	Site no	T	A	S	C
1	55	87	95	86	11	54	71	56	96
2	68	95	65	83	12	54	76	76	69
3	88	93	92	90	13	50	51	81	100
4	57	61	77	62	14	70	55	54	99
5	63	88	61	72	15	90	95	96	93
6	81	76	69	66	16	50	92	65	52
7	85	72	90	83	17	51	84	77	50
8	90	53	82	53	18	69	66	60	61
9	55	76	62	59	19	63	51	82	72
10	81	82	72	53	20	59	82	61	95

Table 6.6: Users with their Preferred QoS Parameters' Values

user no	T	A	S	C
1	75	79	68	88
2	71	77	62	95
3	77	89	78	87
4	65	85	93	71
5	73	91	75	90
6	88	84	99	92
7	89	73	77	75
8	89	91	92	100
9	86	73	97	86
10	98	86	95	89

The second step is to run the process of simulations by using all the systems. AHP and D-system implement the selections for the users depending on their positions in the scheduling queue beginning from the first up to the last. FUSs and TUPQs have been computed on the systems M-system, AHP and D-system as shown in Table 6.7 (a) and 6.7 (b). The same experiment has been repeated 10 times with the same data in Table 6.5, but with different orders in Table 6.6. The 10 different users' orders of Table 6.6 were generated randomly. The results of the aforementioned experiments are illustrated in Tables 6.7 (a) and 6.7 (b). In order to demonstrate the efficiency of M-system over AHP and D-system, the values resulting from the

simulations were computed as shown in Tables 6.7 (a) and 6.7 (b), and the efficiency is expressed by the following equation:

$$\text{Efficiency} = \frac{O_{av} - U_{av}}{O_{av}} \times 100 \quad 6.12$$

Where O_{av} is other system value and U_{av} is the underlying system value

As it is illustrated in Tables 6.7 (a) and 6.7 (b), all experiments show that M-system always performs better than both D-system and AHP in terms of both TUPQ which means more users' satisfaction, and FUS which signals that higher quality and more fairness are achieved. The efficiency in terms of user's satisfaction reaches 24.88% and in terms of fairness reaches 45.31% in comparison with D-system. While with AHP, the efficiency reaches 46.73% and the fairness reaches 46.15% which point pins the significance of M-system. Based on the above experiments and with respect to D-system the average TUPQ enhancement is 16.22 % and the standard

deviation is 6.98. While with respect to AHP, the average enhancement is 38.05% and the standard deviation is 5.55. On the other hand the average FUS enhancement based on D-system is 30.95% and the standard deviation is 11.38 while based on AHP the average FUS enhancement is 33.51% and the standard deviation is 10.75.

Table 6.7(a): FUSs & TUPQs of the 10 Experiments Using M & D systems

Run no	M-system		D-system		Efficiency Based on TUPQ	Efficiency Based on FUS
	TUPQ	FUS	TUPQ	FUS		
1	178.86	7.69	218.92	11.59	18.30%	33.65%
2	178.86	7.69	230.09	11.55	22.27%	33.42%
3	178.86	7.69	186.71	7.85	4.20%	2.04%
4	180.87	7.41	230.64	12.01	21.58%	38.30%
5	178.86	7.69	238.10	10.79	24.88%	28.73%
6	180.87	7.41	218.92	11.59	17.38%	36.07%
7	180.87	7.41	235.47	10.45	23.19%	29.09%
8	178.86	7.69	197.62	11.77	9.49%	34.66%
9	178.86	7.69	190.69	10.72	6.20%	28.26%
10	180.87	7.41	212.03	13.55	14.70%	45.31%

Table 6.7(b): FUSs & TUPQs of the 10 Experiments Using M-system & AHP

Run no	M-system		AHP		Efficiency Based on TUPQ	Efficiency Based on FUS
	TUPQ	FUS	TUPQ	FUS		
1	178.86	7.69	269.08	12.86	33.53%	40.20%
2	178.86	7.69	292.98	11.6	38.95%	33.71%
3	178.86	7.69	335.76	9.19	46.73%	16.32%
4	180.87	7.41	296.55	12.61	39.01%	41.24%
5	178.86	7.69	307.39	9.05	41.81%	15.03%
6	180.87	7.41	307.03	13.76	41.09%	46.15%
7	180.87	7.41	243.01	12.61	25.57%	41.24%
8	178.86	7.69	287.25	10.59	37.73%	27.38%
9	178.86	7.69	290.55	12.81	38.44%	39.97%
10	180.87	7.41	289.87	11.20	37.60%	33.84%

The detailed combinations between users, who are ordered from 1 to 10 for simplicity, and the sites are presented in Table 6.8. The D-system and AHP make matching between the users and the sites based on the users' position in the scheduling queue. M-system provides optimal solutions that are more stable and always show almost the same combinations with little variations based on the position order of the users in the queue (initial population). This shows that the FUSs and TUPQs values in the table are the same to some extent whatever the users' order in

the queue. In contrast, the obtained values for FUSs and TUPQs from the D-system and the AHP always noticeably vary based on the users' position in the queue. The obtained pairing results from D-system and AHP are less efficient performance than that obtained from M-system. Nevertheless, there is a low percent expectation that D-system and AHP could achieve a similar performance (which occurs accidentally) to M-system.

Table 6.8: 10 Experiments Using M-system, AHP & D-system

No	The system	Users									
		1	2	3	4	5	6	7	8	9	10
1	M-system	20	11	5	12	2	1	6	15	7	3
	D-system	20	11	7	1	2	15	6	5	19	3
	AHP	6	20	5	3	2	10	1	15	8	7
2	M-system	20	11	5	12	2	1	6	15	7	3
	D-system	2	11	15	12	5	3	6	20	7	1
	AHP	7	20	6	10	15	2	8	3	1	5
3	M-system	20	11	5	12	2	1	6	15	7	3
	D-system	20	11	2	12	5	3	6	1	7	15
	AHP	20	1	6	15	3	10	5	7	8	2
4	M-system	20	11	2	12	5	1	6	15	7	3
	D-system	2	20	7	1	12	3	6	5	19	15
	AHP	6	20	5	3	2	7	15	10	8	1
5	M-system	20	11	5	12	2	1	6	15	7	3
	D-system	2	20	5	1	3	12	6	7	19	15
	AHP	15	2	20	3	5	7	10	1	6	8
6	M-system	20	11	2	12	5	1	6	15	7	3
	D-system	20	11	7	1	2	15	6	5	19	3
	AHP	8	20	6	2	1	7	3	10	5	15
7	M-system	20	11	2	12	5	1	6	15	7	3
	D-system	2	11	3	1	20	12	6	7	19	15
	AHP	5	20	10	3	2	7	6	15	1	8
8	M-system	20	11	5	12	2	1	6	15	7	3
	D-system	5	20	2	1	11	12	6	15	7	3
	AHP	3	20	10	2	6	15	8	5	1	7
9	M-system	20	11	5	12	2	1	6	15	7	3
	D-system	20	11	2	1	5	12	6	15	7	3
	AHP	7	2	10	6	1	3	8	15	20	5
10	M-system	20	11	2	12	5	1	6	15	7	3
	D-system	5	20	3	1	2	12	6	15	7	10
	AHP	7	2	10	20	1	3	8	5	6	15

6.8.1.1 Statistical Testing

1. TUPQ

Even though, it is very clear from Tables 6.7 (a) and 6.7 (b) that M-system overcomes both systems D-system and AHP. However, a statistical testing is a useful method to achieve a better confirmation concerning the results significance. Therefore, a one-way repeated measure ANOVA has been conducted to compare the three systems M-system, D-system and AHP based on TUPQ measuring metric. The means and the standard deviations are presented in Table 6.9.

Table 6.9: TUPQ Descriptive Statistics for M-system, D-system & AHP

System	Mean	Std. Deviation	N
M- system	179.6640	1.03796	10
D- system	215.9190	18.70657	10
AHP	291.9470	24.39006	10

The mean of TUPQ is 179.6 when utilizing M-system; 215.9 when utilizing D-system; and 291.9 when utilizing AHP. To test whether the difference between the three conditions' mean are significant or not, a multivariate test has been conducted as shown in Table 6.10.

Table 6.10: The Results of Multivariate Tests based on TUPQ

Effect	Value	F	Hypothesis df	Error df	Sig.	partial eta squared
Wilks' Lambda	.023	172.6	2.0	8.0	.000	.977

The results in Table 6.10 show that there is a significant effect on TUPQ [Wilks' Lambda=.023, $F(2, 8)=172.6$, $p<.0005$, multivariate partial eta squared=.977]. Although, there is a significant effect on TUPQ values based on the utilized system especially when using M-system. More analyses have been carried out to set the directions of these differences in TUPQ values. Therefore, tests to shed light on systems effects have been done. The results of the multivariate tests are presented in Table 6.11.

Table 6.11: The Results of Tests Between-Systems Effects based on TUPQ

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Intercept	1575658.336	1	1575658.336	8325.460	.000	.999
Error	1703.320	9	189.258			

The one-way repeated-measures ANOVA shows that these TUPQs are significantly different. $F(1, 9) = 8325.4$, $p < .001$, partial eta squared=.999. Repeated-measures using a Bonferroni adjustment ($\alpha = .05/3 = .017$). Moreover, for pairwise comparisons as presented in Table 6.12 prove that the TUPQs values (Euclidean distances) are significantly shorter when utilizing M-system. Furthermore, there is a significant reduction in the Euclidean distances in comparing AHP with D-system. It appears that D-system performs better than AHP while the M-system is the best among all.

Table 6.12: TUPQ Pairwise Comparisons

(I) factor1	(J) factor1	Mean Difference (I-J)	Std. Error	Sig.a	95% Confidence Interval for Differencea		
					Lower Bound	Upper Bound	
dimension1	1	2	-36.255-	5.797	.000	-53.261-	-19.249-
		3	-112.283-	7.810	.000	-135.192-	-89.374-
	2	1	-36.255-	5.797	.000	-19.249-	-53.261-
		3	-76.028-	11.506	.000	-109.778-	-42.278-
	3	1	-112.283-	7.810	.000	-89.374-	-135.192-
		2	-76.028-	11.506	.000	-42.278-	-109.778-

2. FUS

Similarly the one-way repeated measures ANOVA has been conducted to compare the three systems M- system, D-system and AHP based on FUS measuring metric. The means and the standard deviations are presented in Table 6.13.

Table 6.13: FUS Descriptive Statistics for M-system, D-system & AHP

System	Mean	Std. Deviation	N
M-system	7.58	.145	10
D-system	11.19	1.46	10
AHP	11.63	1.61	10

The mean of TUPQ is 7.58 when utilizing M-system; 11.19 when utilizing D-system; and 11.63 when utilizing AHP. To test whether the difference between the three conditions' mean are significant or not, a multivariate test has been conducted as shown in Table 6.14.

Table 6.14: The Results of Multivariate Tests based on FUS

Effect	Value	F	Hypothesis df	Error df	Sig.	partial eta squared
Wilks' Lambda	.103	34.902	2.000	8.000	.000	.897

The results in Table 6.14 show that there is a significant effect on FUS [Wilks' Lambda=.103, $F(2, 8)=34.902$, $p<.0005$, multivariate partial eta squared=.897]. Although, there is a significant effect on FUS values based on the utilized system especially when using M-system. More analyses have been carried out to set the directions of these differences in FUS values. Therefore, tests to shed light on systems effects have been done. The results of the multivariate tests are presented in Table 6.15.

Table 6.15: The results of Tests of Between- Systems Effects based on FUS

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Intercept	3079.115	1	3079.115	1491.755	.000	.994
Error	18.577	9	2.064			

The one-way repeated-measures ANOVA shows that these FUSs are significantly different. $F(1, 9) = 1491.755$, $p < .001$, partial eta squared=.994. Repeated-measures using a Bonferroni adjustment ($\alpha = .05/3 = .017$). Moreover, for pairwise comparisons as presented in Table 6.16 prove that the FUSs values are significantly smaller when utilizing M-system. It appears that D-system performs better than AHP while the M-system is the best among all.

Table 6.16: FUS Pairwise Comparisons

(I) factor1	(J) factor1		Mean Difference (I-J)	Std. Error	Sig.a	95% Confidence Interval for Differencea		
						Lower Bound	Upper Bound	
dimension1	1	dimension2	2	-3.609-	.482	.000	-5.022-	-2.196-
			3	-4.050-	.532	.000	-5.611-	-2.489-
	2	dimension2	1	-3.609-	.482	.000	-2.196-	-5.022-
			3	-.441-	.532	1.000	-2.003-	-1.121-
	3	dimension2	1	-4.050-	.532	.000	-2.489-	-5.611-
			2	-.441-	.532	1.000	-1.121-	-2.003-

6.8.2 Case (2) Scalability Test and Best Replica

Simulation has been conducted in various methods (scenarios) and compared with the D-system and the AHP to calculate the workability and scalability of the proposed system. It is expected that D-system overcomes AHP because D-system is designed specifically for replica selection with multiple parameters while AHP is a general purpose decision model. In this simulation, the total number of grid sites as a variable and the number of users as another variable are independent, therefore nine situations or scenarios have been examined as shown in Tables 6.17 (a) and 6.17 (b) (the first chromosome of M-system is the one gotten from D-system so both systems begin from the same point). The obtained results have shown that M-system is scalable and overcomes the D-system and the AHP in all scenarios. Fairness efficiency is highly significant as it could reach 72.37 % in comparison to D-system and 88.81 in comparison to AHP. The superiority of M-system is expected due to its nature as a weighted algorithm which conducts prior consideration before making any decision. In contrast, D-system and AHP are both greedy. They satisfy the current user without making any prior consideration about the remaining users exactly like the scenario that selects the closest city in the traveling salesman problem where at the end the distances become very long. Moreover, in terms of TUPQ performance metric, M-system shows better results than D-system. The average improvement value is 5.95 % with a standard deviation equal to 2.8. While in comparison with AHP, the average

TUPQ improvement value is 65.66% with a standard deviation equal to 22.27. On the other hand, the average FUS improvement compared with D-system is 16.48 % with a standard deviation equal to 2.87 and compared with AHP the average FUS improvement is 68.91% with a standard deviation equal to 11.95.

Table 6.17(a): The Performance of 9 Experiments Using M & D systems

Scenarios	Sets	Number of sites	No of requests	M-system		D-system		Efficiency Based on TUPQ	Efficiency Based on FUS
				TUPQ	FUS	TUPQ	FUS		
1	A	50	10	144.60	1.13	145.39	4.09	0.54%	72.37%
2			20	288.74	4.12	313.27	4.4	7.83%	6.36%
3			30	406.72	4.81	436.28	6.09	6.78%	21.02%
4	B	100	20	222.29	2.68	227.18	2.75	2.15%	2.55%
5			30	343.20	3.56	367.88	4.20	6.71%	15.24%
6			50	705.515	6.00	777.04	6.32	9.20%	5.06%
7	C	200	30	279.43	2.47	303.72	3.09	8.00%	20.06%
8			50	464.10	3.46	488.75	3.48	5.04%	6.46%
9			70	644.33	3.35	695.19	3.53	7.32%	5.10%

Table 6.17 (b): The Performance of 9 Experiments Using M-system & AHP

Scenarios	Sets	Number of sites	No of requests	M-SYSTEM		AHP		Efficiency Based on TUPQ	Efficiency Based on FUS
				TUPQ	FUS	TUPQ	FUS		
1	A	50	10	144.60	1.13	390.15	10.10	62.94%	88.81%
2			20	288.74	4.12	669.87	11.94	56.90%	65.49%
3			30	406.72	4.81	995.46	10.27	59.14%	53.16%
4	B	100	20	222.29	2.68	662.07	9.33	66.43%	71.28%
5			30	343.20	3.56	1023.46	13.13	66.47%	72.89%
6			50	705.515	6.00	1737.53	12.11	59.40%	50.45%
7	C	200	30	279.43	2.47	1054.8	12.35	73.51%	80.00%
8			50	464.10	3.46	1750.10	10.95	73.48%	68.40%
9			70	644.33	3.35	2362.23	11.07	72.72%	69.74%

6.9 Summary

In this chapter, a new replica selection system (M-system) has been introduced in data grid environment. M-system integrates the QoS attributes in the replica selection decision making process. The QoS attributes are time, site availability, security, cost and the user's preferences. Addressing multiple users' requests in the scheduling queue simultaneously shapes the main importance of M-system which is demonstrated and mathematically modeled. Genetic algorithm is utilized to cater the complexity of the problem. Fair users' satisfactions (FUS) and the average user's preferred QoS (\overline{UPQ}) are two new proposed matrices to measure the performance of M-system. The robustness of M-system has been investigated and the experimental results are presented. The simulation's results have shown that M-system enhanced the performance of the grid environment and thus, reducing both FUSs and \overline{UPQ} s and increasing the efficiency to extents that could reach to 25%.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Overview

This chapter summarizes the whole research work undertaken in this thesis. The contributions in this thesis are emphasized with their supported experimental results. The limitations of this research are discussed. Ultimately, a number of potential future research directions are identified and suggested to recognize the proposed work.

7.2 Research Summary

This thesis is commenced by reviewing, describing and categorizing some aspects of data grid systems. A data grid has a number of distinctive features like the existence of applications with geographically-distributed locations, heavyweight computing necessities, huge datasets, controlled by different administrative domains, none homogeneous resources, and a lot of users sharing these resources and looking forward to work together. The point that has been presented in this thesis is that sharing single dataset by the many data grid users is infeasible. The solution for this problem is replication management system where identical copies of each dataset (replicas) are duplicated to increase data availability, accessibility and reliability.

On the other hand, replicas initiate the problem of replica placement in the large scale data grids to improve the performance of data access while ensuring efficient use of both bandwidth and storage resources. Moreover, replication can induce another disconcert problem and that is replica selection which is the core work of this thesis. Replica selection algorithm makes critical decisions that choose the best

replica location for the grid user among many replica locations based on some criteria. To this end, a family of efficient systems for dynamic replica selections system in data grids has been proposed (RsDGrid). The proposed system (RsDGrid) has been designed and employed to cover the need of such enhancement in the area, where huge datasets and limited resources are common in data grids. The growing of scientific applications in the domain has increased the value of data replication systems which in turn increased the significance replica selection.

The core problem of this thesis is how to select the best replicas within grid sites that achieve less jobs' times, higher QoS, more and almost equal users' satisfactions. The proposed new replica selection system consists of three components or systems namely, A-system, D-system, and M-system. Each of these systems has its own scope and specifications. RsDGrid is designed to switch among these systems based on the decision maker's requirements. The first two systems, A-system and D-system, represent the continuous improvement of the existing single user replica selection systems by incorporating new features, while M-system goes in a new direction by focusing on simultaneous multiple users' selection.

A-system's main feature is taking into account the site availability QoS parameter in the selection process. Site availability is a very important parameter to address in the replica selection process because the absence of such parameter could lead to faults or at least excessive delays in the jobs' turnaround time. Therefore, it is a continuation of what has been referred to in the literature that aims at focusing on one objective and that is decreasing jobs' turnaround time and decreasing faults. In addition to that one of the main objectives of the A-system is to avoid faults. Simulations have been carried out in the OptorSim grid simulator and the results have proved that A-system outperformed other systems in the literature.

On the other hand, D-system is also a single user approach that considers many different QoS parameters. The main distinction between A-system and D-system is that D-system handles heterogeneous QoS parameters and uses performance metrics other than time. Therefore, the focus in D-system is not only on time but rather on a mixture of QoS parameters, one of which is time. D-system always tries to locate the replica location that has the best ratings for all QoS parameters, and those ratings are

replica location that has the best ratings for all QoS parameters, and those ratings are almost equal to one another. On the other hand, D-system has an option that allows users to choose their preferences with respect to the QoS parameters. Each grid user could have his own preferences. For example, one user may prefer to select the replica location in a minimum turnaround time regardless to the other criteria. Other users may prefer the security criterion more than the turnaround time. While others may only be concerned with cost or targeting a balanced solutions. Therefore, the second approach of D-system considers the user's preferences or priorities, if any, and integrates the preferences into the selection process.

Moreover, the users' preferences can be exchanged among the users in a way that is similar to the stock market. Since the replica selection decision has conflicting criteria and is measured by heterogeneous values, the K-Means model has been employed to solve the heterogeneity of the criteria set in the replica selection system. Simulation results have proved D-system effectiveness in terms of quality of service compared with other systems proposed by previous researchers.

M-system is an upgrade of D-system and other previously proposed systems by facilitating multiple users' selection in order to achieve equal satisfactions amongst the grid users. In all previous replica selection algorithms, users' requests are fulfilled in a FIFO manner, one by one, based on their position in the scheduling queue. This could satisfy the first users in the queue in comparison to those who are at the back. Although it is a difficult task due to the production of huge search, considering all the users' requests to gain fair satisfaction can obtain better results. Therefore, the use of

a hybrid approach between M-system and the Genetic Algorithm (GA) has been proposed to overcome the huge searches involved in replica selection. Simulation results have proved that M-system can be used to solve this complex problem and performs better than the previous works.

7.3 Thesis Contributions

The main contribution in this thesis is improving the replica selection system in data grids in order to allow grid users to fetch their required replicas, reliably, intact, securely and in suitable time. Therefore, the thesis has a number of contributions that can be listed as follow:

- **Introducing a new QoS parameter site availability is considered in the replica selection decision.**

Each site can serve the users depending on the site's local policy, which allows the users to be served for a specific number of hours per day or night or even possibly only at the weekends. Hence, selecting the site with insufficient time, to complete the replica transfer could lead to disconnection. Depending on the fault tolerance approach, the job could be resumed by another site. This may also be prone to disconnection if site availability is not considered or it may be required to restart the entire process from scratch. The worst case that may cause the disconnection is the timeout.

- **Aggregating new QoS parameter the replica selection decision and presenting them as a single value.**

The selection parameters (turnaround time, site availability, security, and cost) are heterogeneous, so it is not possible to simply gather them together to get one value in order to be used in the selection process. Furthermore, the parameters may contradict each other. Therefore, notions from clustering specifically, K-means clustering algorithm by focusing on Euclidean distance, were employed in the selection process to sort out the complexity and heterogeneity of the problem.

- **A new technique that considers the users' preferences in the system to increase user's satisfactions.**

Each grid user has its own view on the QoS parameters set. Some users may prefer to get their required replicas in secure mode regardless the issue of time, while other users may prefer to get their required replicas in minimum time,

regardless of the security issues or cost. A new technique that deploys the users' preferences into the replica selection system has been proposed. Therefore, the system focuses on the preference of users to extensively satisfy them extensively.

- **A new multi user's model is proposed to achieve fair users' satisfactions.**

Typically, users' requests are fulfilled one by one in a First In / First Out (FIFO) manner without taking into account the preferences of the remaining requests in the queue. This could satisfy the front-users (in the scheduling queue) in comparison to those further to the back. Such an improper function results in an unfair distribution of users' satisfaction. Therefore, the new model considers all the users in the queue to achieve fair users' satisfactions. However this approach drastically increases the search dimension. Therefore, Genetic Algorithm (GA) is utilized, to omit the search complexity.

7.4 Limitation of the Thesis

In some cases, a site could show very fast download speed but its time availability is insufficient to complete the download for a certain replica. RsDGrid, in this case, turns away from this site and looks for another one with less speed but with sufficient availability. If RsDGrid utilizes such sites first and then switches to other sites to resume the transferring the replicas in a managed way, this could increase its performance. In another scenario, the best site may not be available for the moment but in a while. This pause of RsDGrid makes the overall performance better. However, the design of RsDGrid ignores this situation. Another limitation in RsDGrid is it is not applicable in parallel download to reduce more jobs' turnaround times.

7.5 Future Works

Based on the aforementioned analysis and the thorough discussion, the proposed solutions could be more developed in some directions to tackle the existing

limitations. In the context of data grid in reality there are two situations. The first is when there are a lot of requesters and a few replicas. The second is when there are a few requesters and many replicas. Based on these two situations the possible enhancement in this research has been summarized as follow:

For the first case, single download is recommended due to the lack of replicas with respect to the number of users where it is inefficient to allow a user to transfer from more than one site. In this scenario, replica transfer from several different grid sites one after another in order to reduce the job turnaround time is more recommended. At the moment, RsDGrid focuses only on one site which is highly expected to complete the replica movement with the assumption that switching to another site is very costly in terms of time which is true if the switching is not planned (disconnection detected by a fault tolerance system, protocol or the user). However, if the switching is planned, the set up for the next connection will take place in enough time prior to the disconnection from the first site as to make the switch smooth and enhance the continuity of the data transfer without any delay. Hence, there is a need to a new component that can decide the best sites to download the replica and in which sequence.

For the second case, the plan is to utilize the parallel transfer in order to decrease the jobs' turnaround times. Hence, there is a need to a new component that can count the number of requesters beside the number of available replica places. The component then can decide to which scenario, the first or the second, the current system status belongs. If the component decided the situation belongs to the second scenario, the system uses parallel download in order to select the required replica from many replica locations concurrently.

LIST OF PUBLICATIONS

- Jaradat, A., Patel, A., Zakaria, M. N., & Amina, M. A. (2013). Accessibility algorithm based on site availability to enhance replica selection in a data grid environment. *Computer Science and Information Systems*, 10(1), 105-132.
- Jaradat, A., Salleh, R., & Abid, A. (2009). Imitating k-means to enhance data selection. *Journal of Applied Sciences*, 9(19), 3569-3574.
- Jaradat, A., Amin, A. H. M., & Zakaria, M. N. (2011). *Balanced QoS Replica Selection Strategy to Enhance Data Grid*. Paper presented at the the 2nd International Conference on Networking and Information Technology Hong Kong, China.
- Jaradat, A., Amin, A. H. M., Zakaria, M., & Golden, K. J. (2012). An Enhanced Grid Performance Data Replica Selection Scheme Satisfying User Preferences Quality of Service. *European Journal of Scientific Research*, 73(4), 527-538.

REFERENCES

- [1] B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke, "Data management and transfer in high-performance computational grid environments," *Parallel Computing*, vol. 28, pp. 749-771, 2002.
- [2] B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, S. Tuecke, and I. Foster, "Secure, efficient data transport and replica management for high-performance data-intensive computing," 2001, pp. 13-13.
- [3] A. Chervenak, E. Deelman, C. Kesselman, B. Allcock, I. Foster, V. Nefedova, J. Lee, A. Sim, A. Shoshani, and B. Drach, "High-performance remote access to climate simulation data: a challenge problem for data grid technologies," *Parallel Computing*, vol. 29, pp. 1335-1356, 2003.
- [4] I. Foster, E. Alpert, A. Chervenak, B. Drach, C. Kesselman, V. Nefedova, D. Middleton, A. Shoshani, A. Sim, and D. Williams, "The Earth System Grid II: Turning climate datasets into community resources," 2002.
- [5] A. Chervenak, E. Deelman, I. Foster, L. Guy, W. Hoschek, A. Iamnitchi, C. Kesselman, P. Kunszt, M. Ripeanu, and B. Schwartzkopf, "Giggle: a framework for constructing scalable replica location services," in *the Supercomputing, ACM/IEEE 2002 Conference*, Baltimore, Maryland, 2002, pp. 1-17.
- [6] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets," *Journal of network and computer applications*, vol. 23, pp. 187-200, 2000.
- [7] I. Foster and C. Kesselman, *The grid: blueprint for a new computing infrastructure*: Morgan Kaufmann, 2004.
- [8] G. Cancio, S. M. Fisher, T. Folkes, F. Giacomini, W. Hoschek, D. Kelsey, and B. Tierney, "The DataGrid Architecture," *Data Grid TechReport DataGrid-ATF-01*, 2001.

- [9] X. H. Sun and M. Wu, "Quality of service of grid computing: resource sharing," 2007, pp. 395-402.
- [10] W. Xing, M. D. Dikaiakos, and R. Sakellariou, "A core grid ontology for the semantic grid," in *Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on*, 2006, pp. 178-184.
- [11] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed systems: concepts and design*: Addison-Wesley Longman, 2005.
- [12] C. S. Yeo, R. Buyya, M. D. de Assunção, J. Yu, A. Sulistio, S. Venugopal, and M. Placek, "Utility computing on global grids," *Handbook of Computer Networks*, pp. 110-130, 2006.
- [13] D. Abramson, J. Giddy, and L. Kotler, "High performance parametric modeling with Nimrod/G: Killer application for the global grid?," 2000, pp. 520-528.
- [14] W. Cirne, F. Brasileiro, J. Sauvé, N. Andrade, D. Paranhos, E. Santos-neto, R. Medeiros, and F. C. Gr, "Grid computing for bag of tasks applications," 2003.
- [15] R. Buyya, D. Abramson, and J. Giddy, "Nimrod/G: An architecture for a resource management and scheduling system in a global computational grid," 2000, pp. 283-289 vol. 1.
- [16] L. Childers, T. Disz, R. Olson, M. E. Papka, R. Stevens, and T. Udeshi, "Access grid: Immersive group-to-group collaborative visualization," 2000.
- [17] K. Seymour, A. YarKhan, S. Agrawal, and J. Dongarra, "NetSolve: Grid enabling scientific computing environments," *Advances in Parallel Computing*, vol. 14, pp. 33-51, 2005.
- [18] (2011, EU-Data Mining Grid. <http://www.datamininggrid.org/>, 2008. Last accessed: May 15, 2011.
- [19] M. Cannataro and D. Talia, "The knowledge grid," *Communications of the ACM*, vol. 46, pp. 89-93, 2003.
- [20] S. Graupner, J. Pruyne, and S. Singhal, "Making the utility data center a power station for the enterprise grid," *Hewlett Packard Laboratories, Tech. Rep. HPL-2003-53*, 2003.

- [21] R. Buyya and S. Venugopal, "The gridbus toolkit for service oriented grid and utility computing: An overview and status report," 2004, pp. 19-66.
- [22] B. Abbott, R. Abbott, R. Adhikari, A. Ageev, B. Allen, R. Amin, S. Anderson, W. Anderson, M. Araya, and H. Armandula, "Detector description and performance for the first coincidence observations between LIGO and GEO," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 517, pp. 154-179, 2004.
- [23] K. Holtman and C. collaboration, "CMS data grid system overview and requirements," Citeseer2001.
- [24] G. Allen, G. Daues, J. Novotny, and J. Shalf, "The astrophysics simulation collaboratory portal: A science portal enabling community software development," 2001, p. 207.
- [25] A. Stell, R. Sinnott, and O. Ajayi, "Supporting UK-wide e-clinical trials and studies," 2008, p. 15.
- [26] G. Aloisio, M. Cafaro, S. Fiore, and G. Quarta, "A grid-based architecture for earth observation data access," 2005, pp. 701-705.
- [27] D. B. Keator, J. Grethe, D. Marcus, B. Ozyurt, S. Gadde, S. Murphy, S. Pieper, D. Greve, R. Notestine, and H. Bockholt, "A national human neuroimaging collaboratory enabled by the Biomedical Informatics Research Network (BIRN)," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 12, pp. 162-172, 2008.
- [28] V. Astakhov, A. Gupta, S. Santini, and J. Grethe, "Data integration in the biomedical informatics research network (BIRN)," 2005, pp. 735-735.
- [29] (2012, The TeraGrid Project, <http://www.teragrid.org/>.<http://www.teragrid.org/>.
- [30] D. Bosio, J. Casey, A. Frohner, L. Guy, P. Kunszt, E. Laure, S. Lemaitre, L. Lucio, H. Stockinger, and K. Stockinger, "Next-generation eu datagrid data management services," *Arxiv preprint physics/0305134*, 2003.

- [31] W. Hoschek, J. Jaen-Martinez, A. Samar, H. Stockinger, and K. Stockinger, "Data management in an international data grid project," *Grid Computing—GRID 2000*, pp. 333-361, 2000.
- [32] R. Tuchinda, S. Thakkar, Y. Gil, and E. Deelman, "Artemis: Integrating scientific data on the grid," 2004, pp. 892-899.
- [33] (2012, The european data grid project, the datagrid architecture. <http://eueudatagrid.web.cern.ch/eu-datagrid/>, 2001. Last accessed: August 28, 2012.
- [34] (2012, Grid Physics Network (GriPhyN). <http://www.griphyn.org/>, 2001. Last accessed: Jun 8, 2012.
- [35] H. Stockinger, A. Samar, K. Holtman, B. Allcock, I. Foster, and B. Tierney, "File and object replication in data grids," *Cluster Computing*, vol. 5, pp. 305-314, 2002.
- [36] K. Ranganathan and I. Foster, "Identifying dynamic replication strategies for a high-performance data grid," *Grid Computing—GRID 2001*, pp. 75-86, 2001.
- [37] M. Lei, S. V. Vrbsky, and Q. Zijie, "Online grid replication optimizers to improve system reliability," 2007, pp. 1-8.
- [38] M. Tang, B. S. Lee, X. Tang, and C. K. Yeo, "The impact of data replication on job scheduling performance in the Data Grid," *Future Generation Computer Systems*, vol. 22, pp. 254-268, 2006.
- [39] C. Dumitrescu and I. Foster, "GRUBER: A Grid resource usage SLA broker," *Euro-Par 2005 Parallel Processing*, pp. 644-644, 2005.
- [40] S. B. Priya, M. Prakash, and K. Dhawan, "Fault tolerance-genetic algorithm for grid task scheduling using check point," in *the Sixth International Conference on Grid and Cooperative Computing*, Los Alamitos, CA, 2007, pp. 676-680.
- [41] S. Venugopal and R. Buyya, "An SCP-based heuristic approach for scheduling distributed data-intensive applications on global grids," *Journal of Parallel and Distributed Computing*, vol. 68, pp. 471-487, 2008.

- [42] P. Wendell, J. W. Jiang, M. J. Freedman, and J. Rexford, "Donar: decentralized server selection for cloud services," in *the ACM SIGCOMM 2010 conference* New York, NY, USA, 2010, pp. 231-242.
- [43] L. Guy, P. Kunszt, E. Laure, H. Stockinger, and K. Stockinger, "Replica management in data grids," 2002, pp. 278-280.
- [44] S. Vazhkudai, S. Tuecke, and I. Foster, "Replica selection in the globus data grid," in *the first IEEE/ACM International Symposium on Cluster Computing and the Grid*, Brisbane, Qld, 2001, pp. 106-113.
- [45] 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," 2010, pp. 187-194.
- [46] R. M. Rahman, R. Alhajj, and K. Barker, "Replica selection strategies in data grid," *Journal of Parallel and Distributed Computing*, vol. 68, pp. 1561-1574, 2008.
- [47] H. H. E. AL-Mistarihi and C. H. Yong, "Response Time Optimization for Replica Selection Service in Data Grids," *Journal of Computer Science*, vol. 4, pp. 487-493, 2008.
- [48] R. M. Rahman, K. Barker, and R. Alhajj, "Replica selection in grid environment: a data-mining approach," 2005, pp. 695-700.
- [49] I. Foster, "Globus toolkit version 4: Software for service-oriented systems," *Network and parallel computing*, pp. 2-13, 2005.
- [50] I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit," *International Journal of High Performance Computing Applications*, vol. 11, pp. 115-128, 1997.
- [51] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *International Journal of High Performance Computing Applications*, vol. 15, pp. 200-222, 2001.
- [52] J. Bresnahan, M. Link, G. Khanna, Z. Imani, R. Kettimuthu, and I. Foster, "Globus GridFTP: what's new in 2007," 2007, p. 19.
- [53] M. J. Wyatt, N. G. D. Sim, D. L. Hardy, and I. M. Atkinson, "Your SRB: a cross platform interface for SRB and digital libraries," 2007.

- [54] T. Ho and D. Abramson, "The griddles data replication service," 2005, pp. 8 pp.-278.
- [55] S. Krishnamurthy, W. H. Sanders, and M. Cukier, "Performance evaluation of a probabilistic replica selection algorithm," 2002, pp. 119-127.
- [56] A. Rajasekar, M. Wan, R. Moore, W. Schroeder, G. Kremenek, A. Jagatheesan, C. Cowart, B. Zhu, S. Y. Chen, and R. Olschanowsky, "Storage resource broker-managing distributed data in a grid," *Computer Society of India Journal, special issue on SAN*, vol. 33, pp. 42-54, 2003.
- [57] H. Stockinger, "Distributed database management systems and the data grid," 2001, pp. 1-1.
- [58] S. Venugopal, R. Buyya, and K. Ramamohanarao, "A taxonomy of data grids for distributed data sharing, management, and processing," *ACM Computing Surveys (CSUR)*, vol. 38, p. 3, 2006.
- [59] R. Chen, N. A. Phan, and I. L. Yen, "Algorithms for supporting disconnected write operations for wireless web access in mobile client-server environments," *Mobile Computing, IEEE Transactions on*, vol. 1, pp. 46-58, 2002.
- [60] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid," *International Journal of Supercomputer Applications*, vol. 15, pp. 200-222, 2001.
- [61] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, January 2002," *Online at: <http://www.globus.org/research/papers/ogsa.pdf>*, 2002.
- [62] G. Singh, S. Bharathi, A. Chervenak, E. Deelman, C. Kesselman, M. Manohar, S. Patil, and L. Pearlman, "A metadata catalog service for data intensive applications," 2003, pp. 33-33.
- [63] S. Singh, M. Cukier, and W. H. Sanders, "Probabilistic validation of an intrusion-tolerant replication system," Citeseer, 2003.

- [64] H. H. E. Al Mistarihi and C. H. Yong, "Replica management in data grid," *International Journal of Computer Science and Network Security*, vol. 8, pp. 22-32, 2008.
- [65] S. Tewari and L. Kleinrock, "Analysis of search and replication in unstructured peer-to-peer networks," 2005, pp. 404-405.
- [66] M. Zhong, K. Shen, and J. Seiferas, "Replication degree customization for high availability," 2008, pp. 55-68.
- [67] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, "Hybrid genetic algorithms: A review," *Engineering Letters*, vol. 13, pp. 124-137, 2006.
- [68] P. B. Hansen, *Operating system principles*: Prentice-Hall, Inc., 1973.
- [69] R. G. Guy, J. S. Heidemann, W. Mak, T. W. Page Jr, G. J. Popek, and D. Rothmeier, "Implementation of the Ficus replicated file system," 1990, pp. 63-71.
- [70] R. Guy, P. Reiher, D. Rather, M. Gunter, W. Ma, and G. Popek, "Rumor: Mobile data access through optimistic peer-to-peer replication," *Lecture notes in computer science*, pp. 254-265, 1999.
- [71] N. Narasimhan and L. E. Chairperson-Moser, *Transparent fault tolerance for Java remote method invocation*: University of California, Santa Barbara, 2001.
- [72] T. Page, R. Guy, J. Heidemann, D. Ratner, P. Reiher, A. Goel, G. Kuenning, and G. Popek, "Perspectives on optimistically replicated, peer-to-peer filing," *Software-Practice and Experience*, vol. 28, pp. 155-180, 1998.
- [73] G. Pierre, M. Van Steen, and A. S. Tanenbaum, "Dynamically selecting optimal distribution strategies for Web documents," *Computers, IEEE Transactions on*, vol. 51, pp. 637-651, 2002.
- [74] D. H. Ratner, "Roam: A scalable replication system for mobile and distributed computing," Citeseer, 1998.
- [75] Y. Saito and H. Levy, "Optimistic replication for internet data services," *Distributed Computing*, pp. 425-442, 2000.

- [76] M. Van Steen, "Distributed Systems Principles and Paradigms," *Network*, vol. 3, p. 26, 2003.
- [77] B. Tierney, J. Lee, L. T. Chen, H. Herzog, G. Hoo, G. Jin, and W. E. Johnston, "Distributed parallel data storage systems: A scalable approach to high speed image servers," 1994, pp. 399-405.
- [78] M. Rabinovich, Z. Xiao, and A. Aggarwal, "Computing on the edge: A platform for replicating internet applications," *Web content caching and distribution*, pp. 57-77, 2004.
- [79] V. Cardellini, M. Colajanni, and P. S. Yu, "Request redirection algorithms for distributed web systems," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 14, pp. 355-368, 2003.
- [80] J. Salas, F. Perez-Sorrosal, M. Patiño-Martínez, and R. Jiménez-Peris, "WS-replication: a framework for highly available web services," 2006, pp. 357-366.
- [81] C. Tan and K. Mills, "Performance characterization of decentralized algorithms for replica selection in distributed object systems," 2005, pp. 257-262.
- [82] C. Huang and T. Abdelzaher, "Towards content distribution networks with latency guarantees," 2004, pp. 181-192.
- [83] S. Buchholz and T. Buchholz, "Replica placement in adaptive content distribution networks," 2004, pp. 1705-1710.
- [84] W. Rao, L. Chen, A. W. C. Fu, and Y. Y. Bu, "Optimal proactive caching in peer-to-peer network: analysis and application," 2007, pp. 663-672.
- [85] Y. Zhao and Y. Hu, "GRESS—a grid replica selection service," 2003.
- [86] W. Cirne, F. Brasileiro, D. Paranhos, L. F. W. Góes, and W. Voorsluys, "On the efficacy, efficiency and emergent behavior of task replication in large distributed systems," *Parallel Computing*, vol. 33, pp. 213-234, 2007.
- [87] C. E. Palau, J. C. Guerri, M. Esteve, F. Carvajal, and B. Molina, "CCDN: campus content delivery network learning facility," 2003, p. 465.
- [88] J. P. Mulerikkal and I. Khalil, "An architecture for distributed content delivery network," 2007, pp. 359-364.

- [89] Akamai. Akamai, cambridge ma, USA. <http://www.akamai.com> [Online].
- [90] The Gnutella protocol specification. <http://www.gnutella.com>. [Online].
- [91] S. Sivasubramanian, M. Szymaniak, G. Pierre, and M. Steen, "Replication for web hosting systems," *ACM Computing Surveys (CSUR)*, vol. 36, pp. 291-334, 2004.
- [92] M. T. Ozsú and P. Valduriez, *Principles of distributed database systems*: Springer-Verlag New York Inc, 2011.
- [93] C. T. Yang, I. Yang, C. H. Chen, and S. Y. Wang, "Implementation of a dynamic adjustment mechanism with efficient replica selection in data grid environments," 2006, pp. 797-804.
- [94] S. Vazhkudai, S. Tuecke, and I. Foster, "Replica selection in the globus data grid," in *Cluster Computing and the Grid*, Brisbane, Qld. , Australia 2001, pp. 106-113.
- [95] F. García-Carballeira, J. Carretero, A. Calderón, J. D. García, and L. M. Sanchez, "A global and parallel file system for grids," *Future Generation Computer Systems*, vol. 23, pp. 116-122, 2007.
- [96] J. Zhang and P. Honeyman, "NFSv4 replication for grid storage middleware," 2006, p. 8.
- [97] Globus Toolkit web site at: <http://www.globus.org/> [Online].
- [98] I. Foster, T. Maguire, and D. Snelling, "Ogsa wsrp basic profile 1.0," ed, 2005.
- [99] C. Ferdean and M. Makpangou, "A scalable replica selection strategy based on flexible contracts," 2003, pp. 95-99.
- [100] M. Sayal, P. Scheuermann, and R. Vingralek, "Content replication in web++," 2003, pp. 33-40.
- [101] M. Guo, M. H. Ammar, E. W. Zegura, and F. Hao, "A probe-based server selection protocol for differentiated service networks," 2002, pp. 2353-2357 vol. 4.
- [102] M. Sayal, Y. Breitbart, P. Scheuermann, and R. Vingralek, "Selection algorithms for replicated web servers," *ACM SIGMETRICS Performance Evaluation Review*, vol. 26, pp. 44-50, 1998.

- [103] E. W. Zegura, M. H. Ammar, Z. Fei, and S. Bhattacharjee, "Application-layer anycasting: a server selection architecture and use in a replicated Web service," *Networking, IEEE/ACM Transactions on*, vol. 8, pp. 455-466, 2000.
- [104] R. Kavitha and I. Foster, "Design and evaluation of replication strategies for a high performance data grid," 2001.
- [105] R. Wolski, "Dynamically forecasting network performance using the network weather service," *Cluster Computing*, vol. 1, pp. 119-132, 1998.
- [106] S. Vazhkudai and J. M. Schopf, "Using regression techniques to predict large data transfers," *International Journal of High Performance Computing Applications*, vol. 17, p. 249, 2003.
- [107] S. Vazhkudai, J. M. Schopf, and I. Foster, "Predicting the performance of wide area data transfers," 2002, pp. 34-43.
- [108] C. ze Wu, K. gui Wu, M. Chen, and C. X. Ye, "Dynamic Replica selection services based on state evaluation strategy," 2009, pp. 116-119.
- [109] A. Shojaatmand, N. Saghiri, S. Hashemi, M. A. Dezfoli, I. Khouzestan, and I. Shiraz, "Improving Replica Selection in Data Grid using a Dynamic Ant Algorithm," *International Journal of Information*, vol. 3, p. 139, 2011.
- [110] K. C. Li, H. H. Wang, K. Y. Cheng, and T. Y. Wu, "Strategies Toward Optimal Access to File Replicas in Data Grid Environments," *Journal of Information Science and Engineering*, vol. 25, pp. 747-762, 2009.
- [111] J. F. Kurose and K. W. Ross, "Computer Networking: a top-down approach featuring the Internet, 2005," ed: Addison-Wesley, 1993.
- [112] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "Iperf: The TCP/UDP bandwidth measurement tool," ed: Version, 2005.
- [113] H. H. E. AL-Mistarihi and C. H. Yong, "On fairness, optimizing replica selection in data grids," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 20, pp. 1102-1111, 2009.
- [114] T. L. Saaty, "Decision making with the analytic hierarchy process," *International Journal of Services Sciences*, vol. 1, pp. 83-98, 2008.
- [115] C. A. Chung, *Simulation modeling handbook: a practical approach*: CRC, 2004.

- [116] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini, "Evaluation of an economy-based file replication strategy for a data grid," 2003, pp. 661-668.
- [117] D. G. Cameron, A. P. Millar, C. Nicholson, R. Carvajal-Schiaffino, K. Stockinger, and F. Zini, "Analysis of scheduling and replica optimisation strategies for data grids using optorsim," *Journal of Grid Computing*, vol. 2, pp. 57-69, 2004.
- [118] J. Zhou, Y. Wang, and S. Li, "A Scalable Replica Management Method in Peer-to-Peer Distributed Storage System," 2006.
- [119] W. H. Bell, D. G. Cameron, A. P. Millar, L. Capozza, K. Stockinger, and F. Zini, "Optorsim: A grid simulator for studying dynamic data replication strategies," *International Journal of High Performance Computing Applications*, vol. 17, pp. 403-416, 2003.
- [120] A. Lihua and L. Siwei, "Job-attention Replica Replacement Strategy," 2007, pp. 837-840.
- [121] R. J. Wilson, "The European DataGrid Project," *Colorado, USA, October*, vol. 22, 2001.
- [122] B. Jacob, I. B. M. C. I. T. S. Organization, and S. B. Online, *Introduction to grid computing*: IBM, International Technical Support Organization, 2005.
- [123] D. Thain, T. Tannenbaum, and M. Livny, "Condor and the Grid," 2003, pp. 299-335.
- [124] D. Cameron, J. Casey, L. Guy, P. Kunszt, S. Lemaitre, G. McCance, H. Stockinger, K. Stockinger, G. Andronico, and W. Bell, "Replica management in the european datagrid project," *Journal of Grid Computing*, vol. 2, pp. 341-351, 2004.
- [125] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys (CSUR)*, vol. 31, pp. 264-323, 1999.
- [126] J. H. Holland, *Adaptation in natural and artificial systems*: University of Michigan press, 1975.

- [127] D. Goldberg, "Computer-aided gas pipeline operation using genetic algorithms and rule learning, PhD Dissertation, University of Michigan," *Ann Arbor (MI)*, vol. 288, 1983.
- [128] R. J. Bauer, *Genetic algorithms and investment strategies* vol. 19: John Wiley & Sons Inc, 1994.
- [129] H. B. Amor and A. Rettinger, "Intelligent exploration for genetic algorithms: using self-organizing maps in evolutionary computation," 2005, pp. 1531-1538.
- [130] A. B. Korol', I. A. Preygel, and S. I. Preygel, *Recombination variability and evolution: algorithms of estimation and population-genetic models*: Springer, 1994.
- [131] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*: Addison-wesley, 1989.
- [132] P.-C. Chi, "Genetic search with proportion estimations," in *Proceedings of the Third International Conference on Genetic Algorithms*, 1989, pp. 92-97.
- [133] C. Vira and Y. Y. Haimes, *Multiobjective decision making: theory and methodology*: North-Holland, 1983.
- [134] A. Hans, "Multicriteria optimization for highly accurate systems," *Multicriteria Optimization in Engineering and Sciences*, vol. 19, pp. 309-352, 1988.
- [135] K. Kärkkäinen, *Shape sensitivity analysis for numerical solution of free boundary problems*: University of Jyväskylä, 2005.
- [136] H. Maaranen, *On heuristic hybrid methods and structured point sets in global continuous optimization*: University of Jyväskylä, 2004.
- [137] J. Koskinen, "Automated transient hypertext support for software maintenance," *Jyväskylä Studies in Computing*, vol. 4, 2000.
- [138] A. Smolianski, "Numerical modeling of two-fluid interfacial flows," PhD thesis, University of Jyväskylä, ISBN 951-39-0929-8, 2001.
- [139] N. Nahar, "Information technology supported technology transfer process: a multi-site case study of high-tech enterprises," *Jyväskylä Studies in Computing*, vol. 9, 2001.

- [140] P. Ifinedo, *Enterprise resource planning systems success assessment: an integrative framework*: University of Jyväskylä, 2006.
- [141] H. Jouni, *integration graphical information system models with visualization techniques*. Pekka Olsbo, Marja-Leena Tynkkynen: Publishing Unit, University Library of Jyväskylä, 2005.
- [142] A. Mursu, *Information systems development in developing countries: Risk management and sustainability analysis in Nigerian software companies*: University of Jyväskylä, 2002.
- [143] K. Miettinen, *Nonlinear multiobjective optimization* vol. 12: Springer, 1999.
- [144] (2013). <<http://edg-wp2.web.cern.ch/edg-wp2/optimization/optorsim.html>>.
- [145] M. Lei, S. V. Vrbsky, and X. Hong, "An on-line replication strategy to increase availability in Data Grids," *Future Generation Computer Systems*, vol. 24, pp. 85-98, 2008.
- [146] D. Zeinalipour-Yazti and N. Kyriacos, "Managing Failures in a Grid System using FailRank," Department of Computer Science, University of Cyprus 2006.
- [147] I. Foster, J. Gieraltowski, S. Gose, N. Maltsev, E. May, A. Rodriguez, D. Sulakhe, A. Vaniachine, J. Shank, and S. Youssef, "The Grid2003 production Grid: Principles and practice," in *13th IEEE International Symposium on High performance Distributed Computing, 2004*, Honolulu, Hawaii, 2004, pp. 236-245.
- [148] (2011, 6-11). *Open Science Grid Consortium*. Available: <http://www.opensciencegrid.org>
- [149] (2011, 28/10). *LCG Grid*. Available: <http://www.gridpp.ac.uk>.
- [150] M. Aggarwal, D. Colling, B. McEvoy, G. Moont, and O. Aa v. d., "A Statistical Analysis of Job Performance within LCG Grid," presented at the CHEP06, Mumbai, India, 2006.
- [151] (2012). *The Linux Information Project*. Available: <http://www.linfo.org/scalable.html>

- [152] D. H. Kim and K. W. Kang, "Design and implementation of integrated information system for monitoring resources in grid computing," in *10th International Conference on Computer Supported Cooperative Work in Design* Nanjing, 2006, pp. 1-6.
- [153] S. Fitzgerald, I. Foster, C. Kesselman, G. Von Laszewski, W. Smith, and S. Tuecke, "A directory service for configuring high-performance distributed computations," 1997, pp. 365-375.
- [154] K. Ranganathan and I. Foster, "Identifying dynamic replication strategies for a high-performance data grid," in *the Second International Workshop on Grid Computing* Denver,CO,2001, 2001, pp. 75-86.
- [155] S. Aberham, P. Baer, and G. Greg, *Operating System Concepts* Seventh ed. vol. 5. New York, NY, USA.: Wiley, 1973.
- [156] S. M. Ross, *Introduction to probability models*, 6th ed.: Academic Pr, 1997.
- [157] W. Sonnenreich and J. Albanese, "Network security illustrated," *Recherche*, vol. 67, p. 02, 2003.
- [158] E. Maiwald, *Network security: a beginner's guide*: McGraw-Hill Osborne Media, 2003.
- [159] V. Vijayakumar and R. S. D. W. Banu, "Security for resource selection in grid computing based on trust and reputation responsiveness," *International Journal of Computer Science and Network Security*, vol. 8, pp. 107-115, 2008.
- [160] S. T. Selvi, P. Balakrishnan, R. Kumar, and K. Rajendar, "Trust based grid scheduling algorithm for commercial grids," in *the International Conference on Computational Intelligence and Multimedia Applications*, Sivakasi, Tamil Nadu, 2007, pp. 545-551.
- [161] Y. Wang and J. Vassileva, "Trust and reputation model in peer-to-peer networks," in *the 3rd International Conference on Peer-to-Peer Computing* Linköping, Sweden, 2003, pp. 150-157.

- [162] S. Naseera, T. Vivekanandan, and K. Madhu Murthy, "Data Replication Using Experience Based Trust in a Data Grid Environment," *Lecture Notes in Computer Science, Distributed Computing and Internet Technology*, vol. 5375/2009, pp. 39-50, 2009.
- [163] E. U. Munir, J. Li, and S. Shi, "QoS sufferage heuristic for independent task scheduling in grid," *Information Technology Journal*, vol. 6, pp. 1166-1170, 2007.
- [164] S. T. Selvi, P. Balakrishnan, R. Kumar, and K. Rajendar, "Trust based grid scheduling algorithm for commercial grids," in *ICCIMA '07 Proceedings of the International Conference on Computational Intelligence and Multimedia* Tamil Nadu, India, 2007, pp. 545-551.
- [165] R. M. Almuttairi, R. Wankar, A. Negi, and C. Rao, "Smart Replica Selection for Data Grids using Rough Set Approximations (RSDG)," in *International Conference on Computational Intelligence and Communication Networks*, Bhopal 2010, pp. 466-471.
- [166] R. M. Almuttairi, R. Wankar, A. Negi, and C. Rao, "Replica Selection in Data Grids Using Preconditioning of Decision Attributes by K-means Clustering (K-RSDG)," in *Second Vaagdevi International Conference on Information Technology for Real World Problems*, vcon, 2010, pp. 18-23.
- [167] H. Sun and Y. Ding, "QoS scheduling of fuzzy strategy grid workflow based on the bio-network," *International Journal of Computational Science and Engineering*, vol. 6, pp. 114-121, 2011.
- [168] D. Kolossa, B. U. Köhler, M. Conrath, and R. Orglmeister, "OPTIMAL PERMUTATION CORRECTION BY MULTI-OBJECTIVE GENETIC ALGORITHMS," *Proceedings of ICA, San Diego, CA*, 2001.
- [169] Z. Michalewicz, *Genetic algorithms+ data structures*: Springer, 1996.
- [170] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary computation*, vol. 3, pp. 1-16, 1995.

- [171] A. Rubinov and R. Gasimov, "Scalarization and nonlinear scalar duality for vector optimization with preferences that are not necessarily a pre-order relation," *Journal of Global Optimization*, vol. 29, pp. 455-477, 2004.
- [172] D. Luc, "Theory of Vector Optimization: Lecture Notes in Economics and Mathematical Systems," *Vol*, vol. 319, 1989.
- [173] R. N. Gasimov, "Characterization of the Benson proper efficiency and scalarization in nonconvex vector optimization," *Lecture notes in economics and mathematical systems*, vol. 507, pp. 189-198, 2001.
- [174] A. Zomaya, R. Lee, and S. Olariu, "An introduction to genetic-based scheduling in parallel processor systems," *IEEE Transactions on Parallel and Distributed Computing Problems*, vol. 15, pp. 111-133, 2001.
- [175] E. Amaldi, A. Capone, and F. Malucelli, "Optimizing base station siting in UMTS networks," in *the 53rd Vehicular Technology Conference*, Rhodes, 2001, pp. 2828-2832 vol. 4.
- [176] S. Gaber, M. El-Sharkawi, and M. N. El-deen, "Traditional genetic algorithm and random-weighted genetic algorithm with GIS to plan radio network," *URISA Journal*, vol. 22, pp. 205-222, 2010.
- [177] K. Z. Gkoutioudi and H. D. Karatza, "Multi-Criteria Job Scheduling in Grid Using an Accelerated Genetic Algorithm," *Journal of Grid Computing*, vol. 10, pp. 1-13, 2012.
- [178] D. H. Kim and K. W. Kang, "Design and implementation of integrated information system for monitoring resources in grid computing," in *the 10th International Conference on Computer Supported Cooperative Work in Design*, Nanjing, 2006, pp. 1-6.
- [179] S. Fitzgerald, I. Foster, C. Kesselman, G. Von Laszewski, W. Smith, and S. Tuecke, "A directory service for configuring high-performance distributed computations," in *the 6th IEEE Symposium on High Performance Distributed Computing*, Portland, Oregon, 1997, pp. 365-375.

