A job response time prediction method for production Grid computing environments.

Hector Ariel Goyeneche

School of Electronics and Computer Science

A Job Response Time Prediction Method for Production Grid Computing Environments

Grid Computing Response Time Prediction (GRTP) Method

Hector Ariel Goyeneche

A thesis submitted in partial fulfilment of the requirements of the University of Westminster for the degree of Doctor of Philosophy.

2010

Abstract

A major obstacle to the widespread adoption of Grid Computing in both the scientific community and industry sector is the difficulty of knowing in advance a job submission running cost that can be used to plan a correct allocation of resources.

Traditional distributed computing solutions take advantage of homogeneous and open environments to propose prediction methods that use a detailed analysis of the hardware and software components. However, production Grid computing environments, which are large and use a complex and dynamic set of resources, present a different challenge. In Grid computing the source code of applications, programme libraries, and third-party software are not always available. In addition, Grid security policies may not agree to run hardware or software analysis tools to generate Grid components models.

The objective of this research is the prediction of a *job response time* in production Grid computing environments. The solution is inspired by the concept of predicting future Grid behaviours based on previous experiences learned from heterogeneous Grid workload trace data. The research objective was selected with the aim of improving the Grid resource usability and the administration of Grid environments. The predicted data can be used to allocate resources in advance and inform forecasted finishing time and running costs before submission.

The proposed Grid Computing Response Time Prediction (GRTP) method implements several internal stages where the workload traces are mined to produce a response time prediction for a given job. In addition, the GRTP method assesses the predicted result against the actual target job's response time to inference information that is used to tune the methods setting parameters.

The GRTP method was implemented and tested using a cross-validation technique to assess how the proposed solution generalises to independent data sets. The training set was taken from the Grid environment DAS (Distributed ASCI Supercomputer). The two testing sets were taken from AuverGrid and Grid5000 Grid environments

Three consecutive tests assuming stable jobs, unstable jobs, and using a job type method to select the most appropriate prediction function were carried out. The tests offered a significant increase in prediction performance for data mining based methods applied in Grid computing environments. For instance, in Grid5000 the GRTP method answered 77 percent of job prediction requests with an error of less than 10 percent. While in the same environment, the

most effective and accurate method using workload traces was only able to predict 32 percent of the cases within the same range of error.

The GRTP method was able to handle unexpected changes in resources and services which affect the job response time trends and was able to adapt to new scenarios. The tests showed that the proposed GRTP method is capable of predicting job response time requests and it also improves the prediction quality when compared to other current solutions.

Acknowledgements

First and foremost I offer my sincerest gratitude to Professor Stephen Winter who has given me the possibility of starting my research in this institution offering his supervision since the beginning of my study. He provided me with many helpful suggestions, important advice and constant encouragement during the course of this work.

I would also like to express profound gratitude to my advisors, Dr. Gabor Terstyanszky and Dr. Thierry Delaitre for they invaluable support, supervision and useful suggestions. They were always there to meet and talk about my ideas and to proofread and mark up my papers. Special thanks go to Dr. Gabor Terstyanszky for organizing my many research meetings, for chasing me up during my studies, and for his prompt corrections and suggestions during the writing up of my thesis.

I completed this thesis while working for the School of Electronic and Computer Science and the Information Systems and Library Services of the University of Westminster. Many colleagues have assisted and encouraged me in various ways during my course of studies. In particular, I would like to acknowledge my managers, Mr. Jonathan Hughes and Dr. Thierry Delaitre for allowing me to undertake my research activities. I could not have completed this journey without their support.

Thanks are due also to several researchers at other organisations that I visited during different stages of the study. I would particularly like to thank the people from the Technical University of Catalonia, Barcelona, for detailed discussion and encouragement in the area of performance prediction in Grid computing.

I whish to acknowledge my friends and colleagues from the Centre for Parallel Computing for supporting me through my research work.

Many thanks are due to all those who read this document and spent hours helping me amassing the information used here.

And finally, I wish to thank all my family and beloved friends for all these singular years.

Contents

Abstract 2
Acknowledgements 4
Contents 5
List of figures 9
List of tables 12
List of algorithms 14
Chapter 1 – Introduction 15
1.1. Problem Statement 16
1.1.1. Research Scene for Response Time Grid Prediction Method 18
1.1.2. Challenges for a Response Time Grid Prediction Method 19
1.2. Research Objective 20
1.3. Grid Computing Response Time Prediction method (GRTP) 21
1.3.1. GRTP Internal Phases 21
1.3.2. Research Tasks 23
1.3.3. Research Contributions 24
1.4. Structure of the Thesis 26
Chapter 2 – Grid Computing Prediction Background 28
2.1. Survey of Grid Computing Prediction Solutions 30
2.1.1. Intelligent Grid Scheduling Service 31
2.1.2. LaPIe: Performance Optimisation of Collective Communications 32
2.1.3. DIMEMAS: Performance Prediction of MPI Applications 33
2.1.4. Modelling Workloads for Grid Systems: Statistical Analysis and Markov-chains 34
2.1.5. ASKALON 35
2.1.6. Downey: Statistical Methods for Prediction of Runtime and Queue Times. 37
2.1.7. Dinda: Time Series for Host Load and Application Runtime Prediction. 38
2.1.8. Smith: Statistical Methods using Templates for Categorisation of Applications 39
2.2. Prediction Solutions Survey Analysis 42
2.2.1. Is another response time prediction method needed? 45
2.3. Chapter Summary and Conclusions 47
Chapter 3 – The Nature of Grid Computing Historical Data: GRTP Implementation Scenario. 49
3.1. Production Grid Computing Testbeds 49

3.1.1. Grid5000 50
3.1.2. DAS 51
3.1.3. AuverGrid 52
3.2. Proposed Workload Trace Definition 53
3.3. Workload Trace Data Fields Analysis 55
3.3.1. Information Theory Background 58
3.3.2. Workload Trace Fields Entropy and Maximum Entropy 65
3.3.3. Workload Trace Fields Dependencies Using Joint Entropy 69
3.4. Job Response Time Data Profile Analysis 73
3.4.1. Workload Header Catalogue Definition (WHC) and its instances (iWHC) 73
3.4.2. iWHC Response Time Data Analysis 74
3.4.3. Data Clustering Solutions Overview 77
3.4.4. Data Clustering Solutions applied to iWHC response time data 79
3.5. Chapter Summary and Conclusions 81
Chapter 4 – GRTP Method Presentation 83
4.1. GRTP Service and OGSA Execution Planning Service Interaction 83
4.2. General Overview of the proposed GRTP Method 86
4.3. Usage Scenarios 88
4.3.1. No Historical Data for an iWHC 89
4.3.2. iWHC has Historical Data but Uncertain Accuracy Level 89
4.3.3. iWHC has Historical Data with Certain Accuracy Level 90
4.4. Summary and Conclusions 90
Chapter 5 – Workload Trace Data Fields Analysis for the GRTP Method 91
5.1. Grid Prediction Workload Format 92
5.1.1. Proposed Grid Prediction Workload Format 93
5.2. Workload Meta-Model and the Creation of WHCs 95
5.2.1. Proposed Workload Meta-Model Method 95
5.2.2. Proposed WHC Creation Method 99
5.2.3. Proposed Method Running Cost Comparison 100
5.3. Selecting the Most Accurate iWHC 101
5.3.1. Proposed iWHC Selection Method: Discounted Accumulative Reward Function 103
5.4. Chapter Summary and Conclusions 104
Chapter 6 – Job Response Time Data Profile for the GRTP Method 106
6.1. Proposed iWHC Data Clustering Solution 107
6.1.1. Proposed Time Base Quality Threshold Clustering Algorithm 107

6.1.2. Analysing the Clustered Results 110
6.2. The Response Time Prediction Methods 113
6.2.1. Job Classification: Stable or Unstable 113
6.2.2. Proposed Response Time Prediction for Stable iWHCs. 116
6.2.3. Proposed Response Time Prediction for Unstable iWHCs 119
6.3. Post-mortem Activities 121
6.4 Summary and conclusions 122
Chapter 7 – GRTP Method Experimental Results and Analysis 124
7.1. Comparison Charts Explanation 124
7.2. Evaluation of Existing Solutions 126
7.3. GRTP Method Evaluation Assuming All Jobs as Stable 129
7.4. GRTP Method Evaluation Assuming All Jobs as Unstable 135
7.5. GRTP Method Evaluation Using a Job Type Method Function 140
7.6. GRTP Method Evaluation Using Different Setting Parameters 144
7.7. Setting Parameters Sensitivity of the Method 146
7.8. GRTP Method Analysis and Experiment Conclusions 147
Chapter 8 – Research Conclusions 150
8.1. Experimental result conclusions 151
8.2. Future work 153
References 155
Glossary of Acronyms 163
Appendix A: Dissemination of the research findings 165
A.1. Journals 165
A.2. Conferences 165
A.3. Chapters in books 167
A.4. News, dissemination and seminars 167
Appendix B: GRTP Method Implementation 168
B.1. GRTP Service Architecture 168
B.2. Response Time Prediction Service Packages 169
B.2.1. Response Time Prediction Engine Package 169
B.2.2. Workload Meta-model and Cataloguing Package 171
B.2.3. Similarity package 172
B.2.4. Clustering package 173
B.2.5. Predictor package 174
B.3. GRTP Database Entity-relationship Model 175

B.4. Testing of the GRTP Service Implementation 177

List of figures

Figure 1 Phases of the proposed Grid Computing Response Time Prediction method. 22
Figure 2 Mapping the research contributions into the proposed GRTP method 26
Figure 3 Relationship between OGSA, WSRF, and Web Services 30
Figure 4 Prediction methods' approaches. 44
Figure 5 Grid5000 sites topology 51
Figure 6 DAS sites topology 52
Figure 7 AuverGrid sites topology 52
Figure 8 Draft for a workload meta-model 57
Figure 9 Diagram of entropy, join entropy, conditional entropy, and mutual information 64
Figure 10 Maximum entropy: Entropy of the probability versus the probability. 64
Figure 11 UserID and GroupID entropy and joint entropy representation 70
Figure 12 JobID and QueueId entropy and joint entropy representation 71
Figure 13 UserID and JobID entropy and joint entropy representation 72
Figure 14 iWHC using jobId app162 and user user573 in Grid5000 74
Figure 15 Response time average (top) and the linear regression (bottom) for an iWHC 75
Figure 16 Catalogue polynomial fitting with 4 (top) and 5 (bottom) degrees for an iWHC. .. 76
Figure 17 Response time patterns inside an iWHC. 77
Figure 18 Taxonomy of clustering methods. 78
Figure 19 Interaction between the Response Time Prediction Service and the OGSA 85
Figure 20 GRTP method general graphical overview 87
Figure 21 GRTP method usage scenarios for an iWHC 88
Figure 22 Workload Trace Data Fields Analysis Steps in the GRTP method 91
Figure 23 Grid workload prediction model structure 92
Figure 24 GPWF and SWF entity relationship diagrams 94
Figure 25 Workload meta-model construction algorithm graphical representation 96
Figure 26 Weighted iWHC accuracy levels for different values of Gamma 104
Figure 27 Workload Trace Data Fields Analysis Steps in the GRTP method 106
Figure 28 Response time data profile in an iWHC. 109
Figure 29 TBQT clustering method for an iWHC 111
Figure 30 TBQT vs. QT: Clustering algorithm running performance comparison. 112
Figure 31 Data charts for an iWHC classified as Stable 115

Figure 32 Data charts for an iWHC classified as Unstable. 116
Figure 33 Data charts for a response time prediction method for a Stable iWHC. 117
Figure 34 Data charts for a response time prediction method for an unstable iWHC. 120
Figure 35 Sample chart: Results for a Method A in Sample Grid. 125
Figure 36 Sample chart: Shows how well a Method A has predicted against a Method B. .. 125
Figure 37 Smith method in DAS. 127
Figure 38 Smith method in Grid5000 (Chart A) and AuverGrid (Chart B). 128
Figure 39 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) in DAS 131
Figure 40 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) versus Smith method in DAS 132
Figure 41 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) in Grid5000 133
Figure 42 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) versus Smith method in Grid5000 133
Figure 43 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) in AuverGrid 134
Figure 44 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) versus Smith method in AuverGrid 135
Figure 45 GRTP method assuming unstable jobs (Gamma 0.8, both thresholds 3min) in DAS (Chart A), Grid5000 (Chart B) and AuverGrid (Chart C) 136
Figure 46 GRTP method assuming unstable jobs (Gamma 0.8, thresholds 3min) versus Smith method in DAS 138
Figure 47 GRTP method assuming unstable jobs (Gamma 0.8, thresholds 3min) versus Smith method in Grid5000 139
Figure 48 GRTP method assuming unstable jobs (Gamma 0.8, thresholds 3min) versus Smith method in AuverGrid 139
Figure 49 GRTP method using a job type function (Gamma 0.8, thresholds 3min) in DAS (Chart A), Grid5000 (Chart B) and AuverGrid (Chart C) 141
Figure 50 GRTP method using a job type function (Gamma 0.8, thresholds 3min) versus Smith method in DAS 142
Figure 51 GRTP method using a job type function (Gamma 0.8, thresholds 3min) versus Smith method in Grid5000 143
Figure 52 GRTP method using a job type function (Gamma 0.8, thresholds 3min) versus Smith method in AuverGrid 143
Figure 53 Most accurate method utilisation share picture 146
Figure 54 General implementation design of the response time prediction method..... 169

Figure 55 Response time prediction engine UML class diagram (relevant classes)..... 170
Figure 56 Workload meta-model and Cataloguing UML class diagram (relevant classes) .. 172
Figure 57 Ssimilarity UML class diagram (relevant classes) 173
Figure 58 Clustering UML class diagram (relevant classes) 174
Figure 59 Predictor UML class diagram (relevant classes) 175
Figure 60 GRTP Database entity-relationship model 176

List of tables

Table 1 Example of job ids in production Grid environment 19
Table 2 Example of historical workload data 19
Table 3 Parameters estimated by the prediction solutions presented in the survey 47
Table 4 Achievements and problems of reviewed solutions in Grid Environments 48
Table 5 Selected testbed characteristics 50
Table 6 A workload trace instance example 54
Table 7 Catalogues used by Gibbons 56
Table 8 Gibbons and Smith data fields used in their catalogues. 57
Table 9 Detailed calculus for the entropy example. 60
Table 10 Detailed calculus for the conditional entropy example. 62
Table 11 Entropy and maximum entropy of the testbeds' fields 67
Table 12 UserID (U) and GroupID (G) joint entropy values 70
Table 13 JobID (E) and QueueId (Q) joint entropy values 71
Table 14 UserID (U) and JobID (E) joint entropy values 72
Table 15 Conditional entropy values 72
Table 16 Grid Prediction Workload Format header example. 93
Table 17 Grid Prediction Workload Format detail example. 93
Table 18 Grid5000 workload meta-model construction example 98
Table 19 Grid5000 example list of WHCs 99
Table 20 Grid5000 example list of iWHCs for a prediction request 100
Table 21 List of WHCs created from the workload meta-model method 101
Table 22 List of catalogues using any combination of fields 101
Table 23 TBQT for iWHC(Grid5000)[{job=app510}{Status=0, user=user165}] 110
Table 24 ClusterWeight function for 3 clusters with different age 114
Table 25 Response time prediction example for a stable iWHC 118
Table 26 Response time prediction example for an unstable iWHC 120
Table 27 Smith method catalogues and its usability in DAS 127
Table 28 Catalogues for Smith method in Grid5000 and AuverGrid 129
Table 29 WHC list for GRTP method in DAS (Gamma 0.8, threshold 3min). 131
Table 30 GRTP method evaluation using different setting parameters 144
Table 31 Setting parameters sensitivity of the GRTP method 147

List of algorithms

Algorithm 1. Quality threshold (QT) clustering method. 81
Algorithm 2. Building the workload meta-model algorithm. 97
Algorithm 3. Time Based Quality Threshold (TBQT) clustering..... 108
Algorithm 4. Weighting a data cluster. 114
Algorithm 5. Function to classify an iWHC as Stable or Unstable. 114
Algorithm 6. Response time prediction method for a stable iWHC. 118
Algorithm 7. Response time prediction method for an Unstable iWHC. 119

Chapter 1 – Introduction

Grid computing concepts and technologies emerged within the scientific community with the objective of connecting resources in order to afford computational collaboration in distributed environments. Grid technologies have been studied in a wide range of e-science projects [1][2][3] and the resulting research has laid the foundation for national and international scientific Grid infrastructures [4][5].

Meanwhile, Grid computing gained popularity in the industry sector. Leading IT industry players, such as SAS[6], Oracle[7], IBM[8], and SUN[9], put Grid computing on their agenda with focus on producing solutions that reduce running costs and improve services standards. The IT activities of industry players have been instrumental in bringing Grid computing out of the science lab and onto the mainstream IT agenda; either through sponsoring academic initiatives, or by undertaking their own projects. The industry players' main motivation for pursuing Grid computing is to provide solutions that can scale out computing capacity on-demand in smaller units, as opposed to buying oversized computing systems for peak periods or uncertain growth. In fact, Grid computing was presented to companies as a solution to save costs and ensure business continuity.

There are many obstacles to overcome when aiming to develop solutions that efficiently manage computing components which are in different administrative domains, with a diverse range of software stack, and that are subject to dissimilar security and access control policies. The combined efforts of the scientific community and the IT industry players helped to create a number of Grid computing solutions [10][11][12] that tried to overcome the aforementioned problems in various ways.

Being able to request and expect a service level agreement (SLA) is a key point for commercial companies using Grid computing. A SLA is a contract where the level of a service (which in practice is usually the delivery or answering time) is formally defined.

A SLA can be negotiated by a service if its expected Quality of Service (QoS) [13] is known. QoS is a general term which is used in Grid computing to denote the level of performance that a client experiences when invoking an operation on a remote server. By enforcing QoSs the requirements coming from SLAs can be satisfied. Planning the necessary capacity to guarantee a QoS in a Grid environment is an important challenge because the global performance of the Grid solution is dependent on the local performance of the components that make up the Grid. These components are generally autonomous and join the Grid as part of a loose federation.

The ideal scenario for guaranteeing a QoS is when the response time of a job running in a specific set of resources is known in advance, because this information can be relied on to reduce the uncertainty in terms of running time and cost. Furthermore, this information can be used to pre-evaluate a client's SLA request in terms of the experienced QoS, and as a result, reject, adjust, or rectify it.

Therefore, the objective of this research is to present a method that is able to predict the response time of a job in a production Grid environments. The research area addressed in this dissertation is inspired by the concept of predicting future Grid behaviours based on previous experiences. Analogous solutions have been successfully employed in real distributed environments. For example, Web search engines rely on historical data and links created by other users to define how relevant a given Web page is; social network sites rely on historical user performance to qualify a network member; and auction sites predict how well a vendor may behave by learning from preceding experiences. These examples lead to the question of how historical information of previous usage of resources can be employed as input data for a Grid response time prediction system.

1.1. Problem Statement

Traditional distributed computing and Grid computing environments differ in several aspects. The term *traditional distributed computing* is used to define a type of distributed computing where the components are homogeneous, open, and controlled by a single organisation. Examples of this type of environment are clusters of computers and massive parallel computers. However, Grid computing environments are composed of non-dedicated and heterogeneous components that may be provided by non-centralised organisations. In this case, different clusters of computers and parallel computers can be connected together to create a Grid environment.

Several methods have been used by traditional distributed computing to assess and predict the behaviour of hardware resources. Traditional trace-driven methods [14] implement solutions that use traces of dynamic instructions without full-function simulation capability as input data. This type of solution processes an execution trace of a benchmark to generate measurements of the dynamic use of resources, the throughput at various pipeline stages, and ultimately the performance of the components. Instruction-level [15] methods execute one instruction at a time to obtain the component architectural state by tracking all cpu register and memory updates. This information is then bundled with its execution results to generate performance predictions.

Thread-level methods [16] employ a similar but faster solution, where functional instruction traces are used to analyse the resource performance prediction at execute a time.

In addition, some methods have been presented to understand the composition of a job in traditional distributed computing. Execution-driven [17] techniques modify application codes by inserting an additional instrumentation code. The resulting instrumented codes are executed on a host machine using actual input data. The execution causes the generation of events reflecting the behaviour of the original applications run on a hardware resource. These solutions predict the software performance by extrapolating the generated events on hypothetical modelled hardware resources.

Many of the traditional distributed computing solutions implement a combination of the methods described above to understand and predict the performance of hardware resources and software programmes. However, these solutions cannot be easily adapted for use in Grid computing environments. Trace-driven performance prediction methods are difficult to implement in Grid computing not only because of the large simulation times when there are few processing elements, but also due to the time needed to develop each trace of instructions. Instruction-level or thread-level methods are inadequate because the complexity of current microarchitectures has increased dramatically, and as a result, the solutions became too complex and time-consuming to use in environments composed of heterogeneous and dynamic elements. Execution-driven methods are also dismissed on the grounds of being extremely slow, too intrusive and machine specific, as well as memory intensive solutions.

Trace-driven, instruction-level, and thread-level prediction methods analyse in details different aspects of the target hardware microarchitecture and software. To consider any of these solutions for a Grid computing environment it is important to give an indication of how complex, time consuming, and intrusive these solutions are. Every new or updated processing node in every cluster of a Grid environment should be analysed and updated. Furthermore, every single new or updated job, software library, or third-party software must be measured and examined. The combination of a low-level analysis and a dynamic environment with thousands of nodes and jobs and with a high level of unpredictability makes these traditional solutions unsuitable for Grid computing environments.

However, some of these methods have been migrated and applied to Grid computing environments, such as PACE [18] and GriPhyN [19]. These solutions study different performance aspects by analysing the computing hardware and programme source codes. However, implementing these solutions imposes restrictions on the running environment. For instance, the source code of each software component must be available for composition analysis, and in many cases it is also required that the source code is written in a specific

programming language. The above restrictions present a significant barrier to a Grid environment. Other restrictions include the need for running an analysis tool in each new or modified Grid resource to generate a hardware model. These aforementioned limitations result in a solution that is non-applicable in real production Grid computing environments.

It is needless to say that a Grid response time prediction solution should not assume characteristics that are impractical in the target environment. Source code programs are not always available and they are not always written in the programming language required by the solution. Libraries and third-party software programmes, such as database systems, are frequently used and they are not always distributed as an open source. In addition, different administrative domain administrators may not agree to run a hardware analysis tool to generate Grid resource models whenever a change in hardware infrastructure is implemented. Furthermore, Grid computing environments are designed to be large and use a complex and dynamic set of resources. Therefore, a small granularity of measurements of the whole Grid infrastructure imposes a scenario where a response time prediction solution may be either difficult or ineffective to implement when producing running-time answers.

Bearing these prerequisites in mind, a response time prediction solution could be composed of a pre-programmed model that relates to the available input fields. The model should be dynamic in order to adapt to the quality and availability of the input data fields. The dynamic model could be manually modified by a user, or automatically by the model itself. In any event, an input data analysis that considers new data fields and searches for significant data patterns should be done so as to make decisions about changes in the pre-programmed model. These activities must not be intrusive to generate the input data, neither inflexible about the necessary input fields. As a conclusion, the pre-programmed models and data mining techniques should be combined to strengthen the proposed method's features, and in doing so, evolve into a better response time prediction solution.

1.1.1. Research Scene for Response Time Grid Prediction Method

The research scene presented in production Grid computing environments differs from traditional distributed computing in the aspect of hardware and software information availability. Benchmarking Grid computing resources is not usually possible because of security restrictions. Furthermore, jobs are considered *black boxes* which are created and tuned by different computer programmers. The jobs' source codes are rarely available for a prediction solution to implement any type of white box analysis before submission.

When ready, those *black boxes* are deployed in Grid environments and used by the community. Many users called these jobs using different parameters. These parameters change the final response time of the job. In some cases these parameters are passed to the job as input fields and, in other cases, the job is internally modified and redeployed (sometimes with a different name). The list shown in Table 1 is a real example of different job's ids that were deployed in a production Grid environment. In this data sample it can be seen the disassociation between the job id and the real meaning of the job. Also, the example shows how the parameters may be passed to the jobs.

User	JobId
user584	app578
user584	app588
user024	Int-m=23-j=4
user024	Int-m=23-j=23

Table 1 Example of job ids in production Grid environment

Up to this point, the research scene suggest that the most important source of information, and maybe the only one available is the historical workload data created by previous job submissions. Table 2 extends the previous data sample with further fields extracted from the historical workload. The historical data presented by these fields, which are heterogeneous from Grid site to Grid site, present several challenges for a response time prediction method.

UserID	JobID	GroupID	QueueID	PartitionID	JobStructure	UsedResources
user584	app578	group6	queue7	G1/site6	UNITARY	395
user584	app588	group6	queue7	G1/site6	UNITARY	370
user024	Int=23-j=4	group8	queue10	G1/site0	UNITARY	781
user024	Int=23-j=23	group8	queue10	G1/site0	UNITARY	781

Table 2 Example of historical workload data

1.1.2. Challenges for a Response Time Grid Prediction Method

The Grid computing environments open up a new scenario where input data is provided by different autonomous sources and where it is difficult to impose significant requirements or restrictions. The first challenge facing Grid prediction methods is how to manage heterogeneous and dynamic data. Within this challenge two different issues need to be addressed: a method that cleans and homogenises the data; and a standard workload format able to hold dynamic changes.

Once the data has been cleaned, the information that is relevant to the solution represents the next challenge. In traditional computing environments, most of the required information can be specified beforehand and retrieved from the different sources. As a result, these solutions can rely on any necessary input field. However, in Grid computing all the relevant information is not always available. Furthermore, relevant information can appear at any time or be hidden in unfamiliar data fields. Consequently, the challenge is to analyse a dynamic set of input fields and decide which of them are relevant to a given solution at a given time.

Having defined the appropriate input fields, the next challenge is to produce a solution that predicts a job response time. At this stage the test is to identify a pre-programmed model that relates the input fields to a method, or combination of methods, that are able to work with the available input data to achieve the prediction objective.

Given that Grid environments are dynamic, the last challenge is to adapt the prediction solution to cope with Grid site changes. This final challenge implies that all sections in the proposed method should be presented in a way that can be adjusted to changes and tuned during its life cycle.

1.2. Research Objective

The main objective of this research is the prediction of a *job response time* in production Grid computing environments. In traditional computing, response time is usually defined as the interval between a user's request and the system response [20]. When this definition is applied to Grid computing there are a number of additional factors that need to be considered, such as data upload, environment creations, security check, security delegation, the response time of Grid middleware, and post data cleaning.

For the purposes of this thesis, the job response time is defined here as the interval between the beginning of a job request (submission) and the end of the corresponding response from the Grid middleware (generation of results). No other pre and post activity in charge of preparing the job environment, such as uploading input files or download output files, are considered part of the job response time. Although there are cases in which these activities are necessary for the execution of the job, they are mostly carried out at a time well before the job submitting occurs or well after the job has generated its results. Therefore, those activities are out of the scope of the objective of this research.

The objective of this research is selected to focus on improving the Grid resource usability and the administration of Grid environments. These attributes can be achieved if Grid users, Grid Middleware, and other jobs have advance knowledge of the time that another job may take

to run. From the Grid architecture point of view, the response time prediction is crucial to achieving efficient job scheduling and management. The predicted data can be used to allocate resources in advance and inform forecasted finishing time and running costs before submission. SLA requests can be evaluated in terms of the expected QoS that, in turn, is based on the predicted response time. Without this feature, a Grid user may see a job finished at some point in the future without a realistic indication of its completion time or resource utilisation.

1.3. Grid Computing Response Time Prediction method (GRTP)

The core original contribution of this research is the Grid Computing Response Time Prediction (GRTP) method. This section introduces the internal phases found in the GRTP method; it also lists the research tasks carried out in this research; and it finally presents the full list of research contributions.

1.3.1. GRTP Internal Phases

The method can be summarised in the following four internal phases (Figure 1):

- *Data representation phase*: This phase has two main objectives, the cleaning of the input data and its posterior normalisation using a dynamic workload model. Data cleaning methods focus on removing data noise that are irrelevant or loosely relevant to the subsequent data analysis [21]. Data noise is the product of low-level data errors that result from an imperfect data collection process. The data cleaning process only focuses on removing data errors given that the analysis that determines how relevant a data field is carried out in the next phase. Once the cleaning is done, a dynamic normalised workload model is populated.
- *Data modelling phase*: This phase is composed of data mining techniques that search the normalised data space to understand the nature of the input data and to prepare a meta-model for the next phase. This phase can be seen as a dynamic two-step data mining process that firstly trims the input data width by selecting only the fields that contain relevant information and, secondly, it reduces the number of data records.
- *Response time prediction phase*: This phase uses a set of calculus and data mining activities to create the final response time prediction for a given job. The type of the job is initially analysed to decide the appropriate response time prediction function. Two different types of

response time prediction functions are presented: a function for stable jobs and another one for unstable jobs.

– *Evaluation and inference*: This solution proposes a method for the assessment of the response time prediction results. Once the target job has finished, the predicted response time is evaluated against the actual target job's response time. This evaluation provides a measurement of accuracy for the prediction function. The measurement of accuracy is used in the inference phase to both adjust the prediction method and propose the tuning of the predictor method's setting parameters.

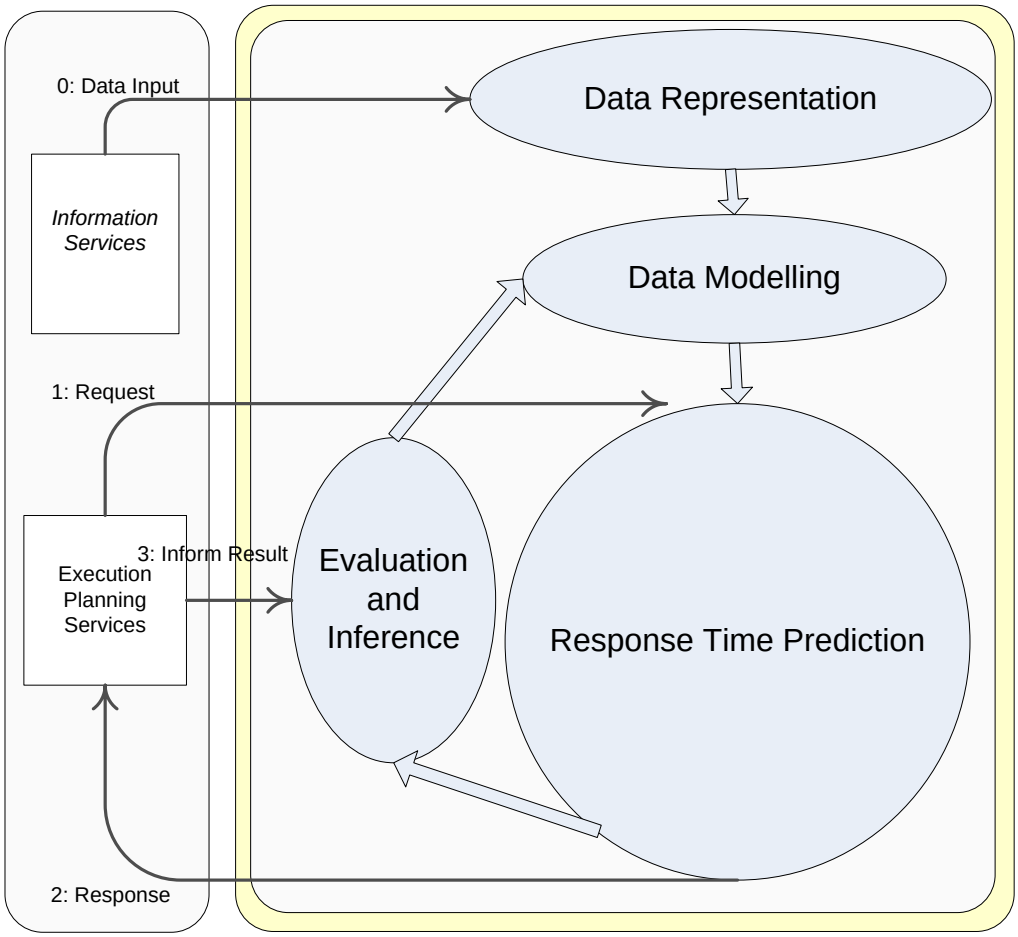


Figure 1 Phases of the proposed Grid Computing Response Time Prediction method.

1.3.2. Research Tasks

This section outlines the research tasks that guided the research process:

T1. Investigate traditional and Grid computing prediction solutions to establish the state of the art in this field, by establishing the achievements and limitations of each method.

T2. Identify which are the achievements and contributions from existing prediction solutions. Also, determine the existing limitations that need to be overcome.

T3. Since the aim of this research is to produce a solution that works in production Grid environments, the solution should neither generate its input data in an intrusive manner nor demand an impractical list of input fields to perform a prediction analysis. Therefore the solution should be able to work with the available data produced by different Grid sources e.g. their workload traces. This argument implies a research task that investigates and implements the cleaning and normalisation of heterogeneous Grid workload traces originating from different Grid sources.

T4. If the workload traces are dynamic, the data entity model should be able to handle this. Therefore, another task is to define a Grid prediction workload format able to hold a dynamic set of input fields needed for a job response time prediction.

T5. A dynamic set of data fields can grow and change with time, making a solution unmanageable in a production Grid computing environment. As a result, it is important to investigate and provide a solution to keep the complexity of the data model under control without losing information.

T6. It is not just the spectrum of data fields that can be narrowed without losing information; the amount of historical data records can also be reduced. In this case the objective is to work only with the data set that is related and relevant to the target job. Therefore, the task is to determine how to select the correct data set.

T7. Assessing how a set of data fields and records can be combined together to predict a job's response time is an important task in this research. This task requires the analysis of different modelling and data mining methods.

T8. A dynamic solution is suitable for Grid environments where resource services can change without centralised control. A task that evaluates the predicted values and infers which section of the model should be adapted to cope with a new environment is also required. Another challenge of this task is to identify to which extent these changes can be managed automatically.

T9. The final task is to demonstrate how the response time predictor can be used by Grid users as well as other Grid components, such as Grid Schedulers, to improve the overall Grid environment utilisation.

1.3.3. Research Contributions

This research makes five distinct contributions to scientific knowledge in the area of prediction of job response time in production Grid environments:

C1. **Grid prediction workload format:** The computing workload study and normalisation area have been studied by a number of authors [22][23][24][25]. Their common approach is to define a set of fields where the workload trace data can be contained. These current workload formats have a fixed number of fields, and the semantic of the data is given by the name of each field. When the computing environment evolves, the different workload formats are forced to adapt accordingly. In the case of Grid computing environments, the data sources are in a continuous state of evolution. This evolution is driven by the technology change (new hardware and software), the volatility of resources and services, and the internal changes in the administrative policies that publish the data. The first contribution of this research deals with data generated by distributed sources defining a workload format that accepts changes in services and resources. To achieve this, the Grid Prediction Workload Format is defined (Figure 2, callout C1) in a manner that allows the inclusion and exclusion of any data field. The Grid Prediction Workload Format:

- Allows any new data field to be included into the workload format when it is produced instead of discarding it, which is what happens with current formats. In this scenario, any new data field that may potentially provide relevant information is automatically available for the prediction method.
- Traditional semantic views used in Grid solutions restrict a data field's value to the field's name. The proposed workload format disregards the field name and allows qualifying the data fields based on the quality of the information that they contain.

C2. **Workload meta-model method**: This contribution created a workload meta-model which can be seen as an interface to access the Grid Prediction Workload format. The workload meta-model method creates a layer on top of the workload format by summarising the data fields and putting them into a hierarchy based on the quality of information of each field. The hierarchy is automatically adapted (extended or reduced) according to the data generated by the Grid environment. Only new relevant information is used to extend the

meta-model while old or obsolete information is removed to reduce it. The Workload meta-model method does not only ensure that new fields are used only when they become relevant, but it also reduces the overall complexity of the method by excluding irrelevant data fields (Figure 2, callout C2).

C3. **Historical data similarity**: The contribution C2 provides the needed number of fields connected in a meaningful hierarchy. This new contribution determines what subset of records of the historical information should be considered to predict the target job's response time. The predictor may produce wrong results if all historical records are used; therefore only a subset of the historical records based on the target job should be selected. This contribution analyses different statistics and data mining options and implements a reinforcement learning method to select an appropriate subset of records. Reinforcement learning has the main advantage that it does not need to be supervised, it also deals with aged data, and the selection of subsets is automatically tuned over time (Figure 2, callout C3).

C4. **Job's response time predictor:** The contributions C2 and C3 produce a subset of fields and records that are relevant to the job. This contribution uses this subset to predict the response time. The predictor determines the type of job and selects the prediction methods accordingly. This contribution has three original components: a function that determines the type of a job, a data clustering method for Grid computing workload traces, and two response time prediction methods (Figure 2, callout C4).

C5. **Grid Computing Response Time Prediction method (GRTP):** The final contribution of this research defines a framework for predicting job response time and embodies the proposed method. In this framework all previous contributions are put together to produce a response time answer for a given Grid computing job request (Figure 2, callout C5).

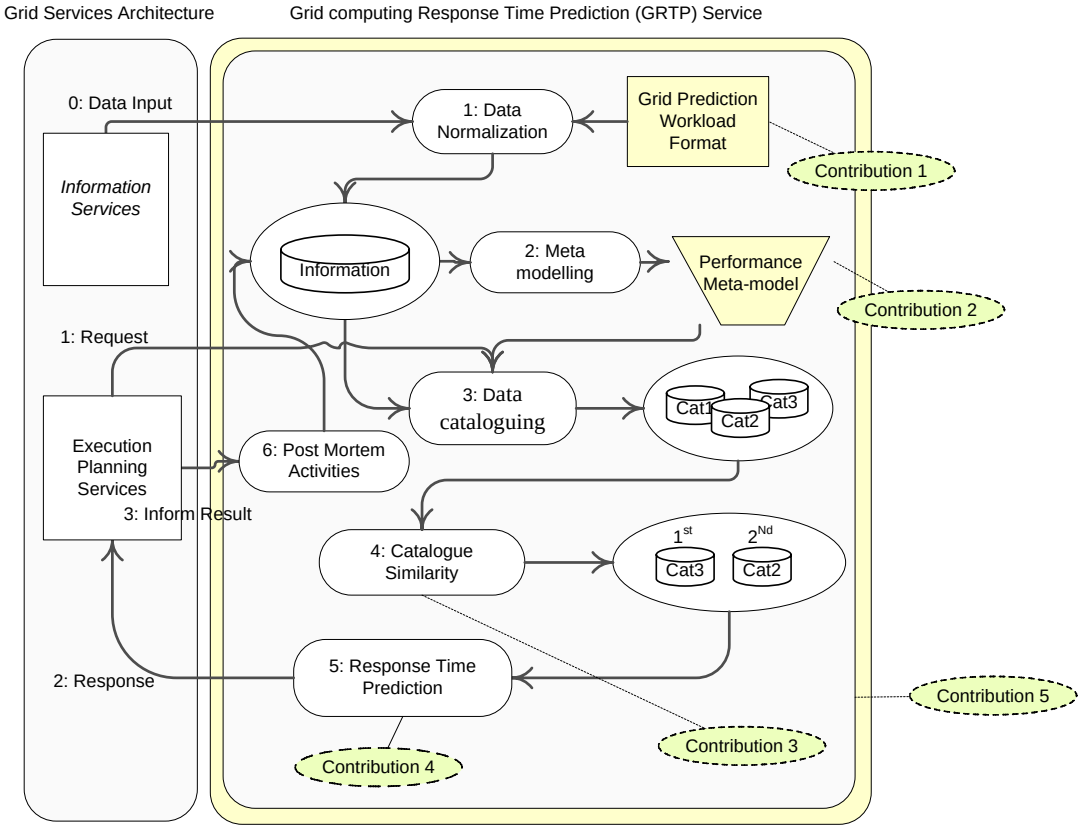


Figure 2 Mapping the research contributions into the proposed GRTP method

1.4. Structure of the Thesis

The remainder of this thesis is structured as follows:

- Chapter 2 reviews the state of the art with respect to current Grid prediction solutions. The objective of this chapter is to summarise the achievements and limitations of current solutions and introduce the points that GRTP should consider.
- Chapter 3 presents a detailed analysis of the type of data available from production Grids. The analysis exposes the existing type of the data, describing in detail the rules and quality of information that GRTP may count as input data.
- Chapter 4 introduces GRTP describing and justifying each of its components. This chapter helps the reader to familiarise with the proposed solution. It is recommended that this chapter is read before the following two chapters, where a detailed explanation of the method is given.
- Chapter 5 describes the first section of GRTP. This section is primarily dedicated to dealing with workload trace data management issues. This part of the solution takes a Grid

workload trace as input parameter and produces a set of records that are similar to the predicted target job as output parameter.

- Chapter 6 describes the second section. In this chapter, the response time prediction functions are presented. This part of the solutions takes the output produced by the data management section (Chapter 5), and produces a final response time prediction for a target job.
- Chapter 7 presents a list of validation results produced by applying GRTP to production Grid computing environments. Those results are also compared against other existing prediction solutions.
- Chapter 8 summarises the research work, highlights the novel aspects produced in this thesis, and presents suggestions for future work.
- Finally, Appendix A describes how GRTP has been implemented. The appendix helps to demonstrate the different implementation characteristic that GRTP had to consider at implementation time.

Chapter 2 – Grid Computing Prediction Background

Parallel computing has been a reality for many years and its use over the past decade has been accompanied by the vast increase in the use of networked computing. This phenomenon has mostly been driven by the advance in the IT communication structure, the cost reduction in communication areas and the improvement and low cost of computer hardware.

The creation of clusters of homogeneous workstation established a new icon in the parallel computing area. Many ideas of massive parallel computers were applied to clusters of computers producing admirable results, especially in cost-benefit aspects. Portable message-passing environments, such as Parallel Virtual Machine (PVM) [26] or Message Passing Interface (MPI) [27], permitted a homogeneous collection of networked computers to be viewed by an application as a single distributed-memory parallel machine. The possibility of setting a cluster of computers as a single parallel machine was a significant help for parallel programmers.

The important achievements in computer clusters confirmed that collaboration, data sharing, and new models of interaction are essential points of parallel computer systems. It also highlighted that the utilisation of idle resources capacity provides an opportunity for users to have substantially more computational power within and across enterprises.

When the idea of Grid computing was first discussed, Internet technology was already a reality offering communication and information exchange among resources. However, the Internet did not provide a coordinated use of resources. Grid computing emerged to cover this deficit and to offer what it is considered the next-generation Internet.

From 1997 onwards, the open source Globus Toolkit (GT) emerged as the de facto standard for Grid computing. Focusing on usability and interoperability, GT defined and implemented protocols, APIs, and services used in hundreds of Grid deployments worldwide. By providing solutions to common problems such as authentication, resource discovery, and resource access, GT accelerated the construction of real Grid applications. And by defining and implementing standard protocols and services, GT pioneered the creation of interoperable Grid systems.

In 1999, the publication of *The Grid: Blueprint for a new computing infrastructure* [28] defined Grid computing as an infrastructure that combines computers, storage systems and other devices to enable the deploying and running of advanced applications. These applications are distinguished from parallel programs running in clusters by the simultaneous use of large

numbers of heterogeneous resources, which could be provided by multiple administrative domains with a non-dedicated structure in a complex system of communication.

The nature of Grid computing, which tries to take broadly distributed, heterogeneous computing and data resources and aggregate them into an abstracted set of capabilities, demanded open standards for integration and interoperability. The first step towards a solution that addresses this problem was the Web Services Architecture [29] definition, which is seen as the foundation of Grid computing.

The Web Services Architecture is defined within the Word Wide Web Consortium (W3C) for the Web Services Architecture Working Group. Web Services is a technology that allows applications to communicate with each other in a platform and programming language independent manner. Web Services have a clean separation between interface (what the service does) and implementation (how it does it). A Web Service interface describes, using the Web Service Description Language (WSDL) [30], a collection of operations that can be accessed over the network using SOAP messages [31].

Web Services are stateless. This means that the Web Service cannot remember information, or maintain state, from one invocation to another. Furthermore, Web Services do not address fundamental issues in distributed computing relating to how to name, create, discover, monitor, and manage the lifetime of services.

With so many potential services and so many interactions between them, there was a potential for chaos. The solution to this problem is standardisation which can be achieved by defining a common interface for each type of service call.

The Open Grid Services Architecture (OGSA) [32], initially presented by the Globus Alliance and IBM in 2002 and currently developed by The Global Grid Forum, presented a true community standard with multiple implementations (including, in particular, the OGSA-based GT 3.0, released in 2003). Building on and significantly extending GT concepts and technologies, OGSA firmly aligns Grid computing with broad industry initiatives in service-oriented architecture and Web services.

OGSA aims to define a common, standard, and open architecture for Grid-based applications. The goal of OGSA is to standardise practically all the service calls that a client can commonly find in a Grid application (job management services, resource management services, security services, etc.) by specifying a set of standard interfaces for said services. At the time of writing, November 2009, this set of standard interfaces is still under development. However, OGSA has already defined a set of requirements that must be met by these standard interfaces.

Web Services was the underlying technology chosen to create the OGSA architecture. But OGSA needed a stateful service to work. Giving Web Services the ability to keep state

information whilst at the same time keeping them stateless was achieved by storing the state information completely in a separate entity called a WS-resource.

The Web Services Resources Framework (WSRF) is a collection of different specifications all relating to each other, which manage the WS-resources. WSRF specifies how resource properties are defined and accessed. WSRF supplies some basic mechanisms to manage the lifecycle of WS-resources, it supplies a way of grouping services or WS-resources together, it provides a way of reporting faults when something goes wrong during a WS-service invocation, it provides a notification to subscribers if a change occurs in the Web service (or, more specifically, in one of the WS-resources), and it also provides a method to address Web Services that is much more versatile than plain URIs.

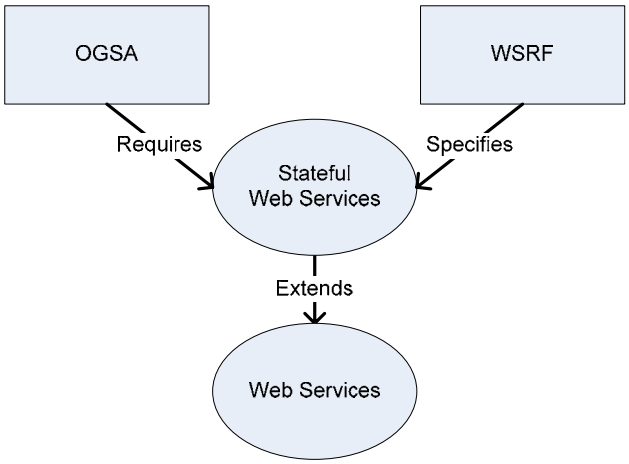


Figure 3 Relationship between OGSA, WSRF, and Web Services

2.1. Survey of Grid Computing Prediction Solutions

This section presents a comprehensive analysis of different Grid computing prediction methods. The solutions presented in this section have been selected to widely represent different approaches in the research area.

The analogies are presented in a standard format as follows:

- **Description:** A description of the solution and how the estimations are calculated.
- **Predictor analysis:** Analysis of the solution highlighting its achievements and limitations.
- **Publications:** List of publications describing this model.

2.1.1. Intelligent Grid Scheduling Service

Description: The objective of the Intelligent Grid Scheduling System (ISS) is to provide a middleware infrastructure allowing optimal scheduling of jobs in a computational Grid. According to data collected on the machines in the Grid, on the behaviour of the applications, and on the performance requirements demanded by the user, the best-suited computational resource is detected and allocated to execute the job. The resources and jobs are parameterised to allow tailoring a computational Grid to a set of applications. The monitoring information collected during execution is put into a database and reused for the next resource allocation decision. In addition to providing scheduling information, the collected data affords the detection of overloaded resources.

ISS presents a mathematical model that has two main components; computational nodes, and jobs. A single computational node is characterised by the peak performance of the processor, the peak main memory bandwidth and the communication network that is characterised by the average network communication bandwidth. The behaviour of a well-equilibrated job (a job where all the parallel tasks have quite the same amount of work and communication) on a computational node is characterised by the number of floating point operations and the number of words and messages sent to other nodes.

These two components, the communications and processing times of the job and the characterisation of the resource, are mathematically combined to calculate a coefficient which expresses if a given hardware is suitable to efficiently run a job. The aim of the mathematical model is to select the most appropriate resource by maximising the processor and memory used and communication bandwidth.

Predictor analysis: To verify this model, the exact data on the resource's cpu usage should be collected. The monitoring module measures and collects the relevant execution quantities (Mflops/s rate, memory needs, cache misses, communication and network relevant information) during the job execution. These quantities are computed through a direct access to hardware counters. At the end of the execution, this information is sent to the ISS service that is used for future job submissions.

The outcome of this method suggests which hardware is most suitable for a given job. However, this method has two weaknesses when used in a production Grid environment; each new job or resource needs to be intrusive-monitored beforehand, and only jobs which are well-balanced and whose computations and communication times do not overlap can be predicted.

Publications: [33][34][35][36]

2.1.2. LaPIe: Performance Optimisation of Collective Communications

Description: The popularity of heterogeneous parallel processing environments like clusters and computer Grids has emphasised the impact of network heterogeneity on the performance of parallel applications. Collective communication operations are especially concerned by this problem, as communication heterogeneity interferes directly on the performance of the communication strategies. Therefore, the aim of LaPIe is the study the optimisation of collective communications in heterogeneous systems like computational Grids.

LaPIe combines both topology discovery and performance prediction to choose the best communication scheduling that minimises the overall communication time for a given operation Firstly LaPIe models the performance of collective communications in homogeneous clusters using network parameters such as latency and bandwidth. Based on the previous model LaPIe uses topology discovery techniques to decompose the Grid environment into islands of homogeneous clusters (logical cluster). Finally, a communication scheduling method is assigned to each logical cluster depending on the cluster's characteristics. Once a prediction on the communication inside each logical cluster is set, a communication scheduling for the overall Grid communication pattern is presented.

Predictor analysis: Different experiments were used to prove the quality of the predictions. At a logical cluster level, the predictions with real communications were compared while varying message sizes (zero bytes to 1MB), number of processes (1 to ~50) and network infrastructures (fast ethernet, giga ethernet and myrinet). The average error rate was around 5 to 10 percent. At Grid level, different scheduling heuristics were also evaluated. It was observed that predictions follow the same behaviour of real communications. These results confirmed that performance predictions and scheduling heuristics can be used to minimise the execution time of collective communications.

Even if LaPIe does not predict the behaviour of jobs executed in Grid environments, LaPIe highlights how communication heterogeneity interferes directly on the performance of the jobs. One of the most interesting contributions from LaPIe is the definition of logical clusters. This definition moved a rather complex scenario comprised of heterogeneous network elements into several virtual clusters composed of homogenised elements. The new scenario makes easy the prediction task. This contribution can be used not only at the level of network communication, but also at the level of Grid resource behaviour.

Publications: [37][38][39][40][41][42][43][44]

2.1.3. DIMEMAS: Performance Prediction of MPI Applications

Description: Dimemas is a performance prediction tool for message-passing jobs, and it enables the user to develop and tune parallel applications on a workstation whilst providing a performance prediction on the parallel target machine. The Dimemas simulator reconstructs the time behaviour of a parallel application on a machine modelled by a set of parameters. The supported target architecture classes include networks of workstations, single and clustered SMPs, distributed memory parallel computers, and even heterogeneous systems.

Dimemas can be used for a wide range of purposes: prediction of application runtime on a specific target machine; study of load imbalances and potential parallelism; sensitivity analysis for architectural parameters; identification of application modules that are worthwhile to optimise

To produce the prediction data, Dimemas require the definition of the network and resource models. The network model includes the number of nodes in the system, network type, network bandwidth, number of connections, collective communication type, and collective communication configuration. The resource model requires the definition of the resource architecture by providing the number of processors, input/output links, local communication start-up, remote communication start-up, relative processor speed, and local bandwidth. In addition to the network and node models, Dimemas also requires a trace file of the applications to be studied containing the computation operations and the calls to MPI primitives.

Predictor analysis: Dimemas goes through the execution trace file and re-estimates the time to execute each computation and communication operation based on the different parameters given in the architecture model. The estimated values are divided into global statistics where the execution time and the speed-up of the entire simulated environment can be analysed and per-process statistics where the estimation of the time required for each computation and communication process is analysed.

When the entire data fields that modelled the network and nodes are present together with the trace file of the target job, the system obtains an average prediction error range between 10 and 20 percent. The predicted results are accurate and the tool has the capability of moving the prediction resource targets as long as the entire information regarding resources, network and jobs are available beforehand and well described.

However, from the dynamic Grid computing point of view, this solution presents the same weakness found in the Intelligent Grid Scheduling System (ISS) described previously; to impose a monitoring method on each new job for the prediction method to work in a dynamic environment is an important restriction. In addition, the static definitions of the network and

nodes work against a dynamic environment proposed by current Grid computing. These points need to be addressed before implementing Dimemas in a prediction Grid environment.

Publications: [45][46]

2.1.4. Modelling Workloads for Grid Systems: Statistical Analysis and Markov-chains

Description: Understanding of computing workload is an important aspect for job scheduling. The scheduler can improve its quality if the characteristics of the workload running on such resources are known. Therefore a representative workload model can be used for performance evaluation. This research is focused on the studies and production of workload models.

The first model is produced by studying the submission patterns and user behaviours on existing workload traces from parallel computers. Based on this analysis, the research deduced that common patterns can be found which can be used to classify users into groups. It showed that a set of 3-4 user groups are sufficient to adequately model a whole supercomputer workload. As a result, a new workload model, called MUGM (Mixed User Group Model), which maintains the characteristics of individual user groups was proposed.

The second workload model proposes the creation of different Markov chains for each Job parameters such as the runtime and the required number node. From there, the correlation between the job parameters is created by the combination of the different Markov chains. In order to do this, a novel approach of transforming the states in the different Markov chains is presented.

Both workload models are also combined by statistical analysis. The association of users to similar job submissions is used to identify groups of users. The groups are modelled based on the statistical parameters of their job submissions. Analysis is performed on job parallelism, runtime, re-occurrence of jobs and arrival times.

Predictor analysis: When workload modelling allows the creation of new synthetic workloads after the original one, the simulations provide more realistic results compared to existing statistical modelling approaches. In turn, these new synthetic workload can be used to evaluate new scheduling strategies.

In the MUGM model, the job submission process has a direct association with individual user groups. This information can be exploited for individualised quality criteria considered by scheduling strategies. Furthermore, additional workload parameters can be modelled with regards to the individual scheduling objectives of these user groups.

Moreover, the Markov-chain-based user-group modelling can be applied to predict future job submissions. Such information can be used to improve scheduling decisions in an online environment. Markov chains model incorporates temporal dependencies and the correlation between job parameters.

The quality of the modelling methods has been evaluated using existing real workload traces. The modelled workload submissions have been compared to existing workload models which are usually based on individual statistical modelling of different job parameters. The modelling results showed improved similarity with real traces. However, these comparisons are based on individual workloads with different level of improvements. The main advantage lies in the ability to classify and model user groups without existing group information just by similarities of user behaviours. This allows further individual modelling of these user groups and corresponding prioritisation and consideration in scheduling.

Publications: [47][48][49][50]

2.1.5. ASKALON

Description: The goal of ASKALON is to simplify the development and optimisation of Grid applications. The ASKALON project has a set of tools, services, and methodologies to make Grid application development and optimisation for real applications an everyday practice.

The ASKALON components are a Resource broker that targets negotiation and reservation of resources; Resource monitoring supports the monitoring of Grid resources by integrating and extending existing Grid resource monitoring tools; Information service is a general purpose service for scalable discovery, organisation, and maintenance of resource and application-specific data (including online and post-mortem); Workflow executor service targets coordinated activation, and fault tolerant completion of activities onto the remote Grid sites; (Meta)-scheduler performs appropriate mapping of single or multiple workflow applications onto the Grid; Performance prediction is a service through which techniques for accurate estimation of execution time of atomic activities and data transfers, as well as of Grid resource availability are performed

The Performance prediction tool that estimates the job execution time and the queue wait time consists of the following phases:

- Automatic Training Phase: Automatically collects the training set, on the Grid-sites, according to the designed experiments and different loads on cpu and memory. This model obtains the available Grid-sites through Grid resource management and brokerage system of ASKALON and uses the ASKALON GLARE model for automatic deployments.

- Performance sharing and translation using soft-benchmarks: A mechanism used to translate job execution time information inter- and cross-platform.
- Performance Prediction: Predicts the job execution time on different Grid-sites with different problem sizes under different loads on memory and cpu, using the minimum training set and performance translation model.

Predictor input:

- Inputs for Experimental Design: Grid-site sample space, problem-size sample space, machine-size sample space. Information about the Grid-sites is obtained from Grid resource management and brokerage system of ASKALON, while information about the application is specified by the user of the application.
- Inputs for Automatic Training Phase: Selected Grid-sites, discrete problem sizes and machine-size sample space. Selection of Grid-sites from the Grid-site sample space is automatic through the initial runs of Automatic Training Phase.
- Inputs for Performance Prediction: application name, activity name, problem-size, Grid-site, machine-size and current environment parameters (load on cpu and memory). The prediction engine connects to an information base, consisting of minimum training set yielded from automatic training phase. The information about the current environment of the machine (like cpu and memory background load) is obtained through NWS.

Predictor analysis: A minimum training set and information from previous actual runs is used for prediction. If the direct information is not available in the database it is supplied by one of the identical machines. In case the data is not available from an identical machine, a mechanism of translation is used. Simple statistical methods of piece-wise straight line fit and b-splines are used for interpolation. Effect of background load on memory and cpu is adjusted before calculating the final prediction.

It was observed in the experiments that most of the jobs were predicted with 10 percent of error and the standard deviation of the prediction was 2 percent at its highest. All the training phase and performance prediction experiments were conducted with 3 real-world applications on the Austrian Grid. The method yields a reduction in training phase compared to exhaustive analysis.

Publications: [51][52]

2.1.6. Downey: Statistical Methods for Prediction of Runtime and Queue Times.

Description: Downey's technique models malleable jobs (jobs that can change the number of processors on which they are executing at runtime in response to an external command) using the average parallelism of a programme and its variance in parallelism. The procedure characterises all applications in the workload, then models the cumulative distribution functions of the runtime in each category, and finally uses these functions to predict application runtime. A two-part function is presented based on the variance in the degree of parallelism. Downey is able to summarise a job behaviour when the model fits the observed speedup curves by saving only the average parallelism of a programme and its variance in parallelism.

The solution estimates the remaining queue time for a job that is at the head of the queue. The input data used for the estimate is the current status of the cluster of computers, including all the jobs that are running with their respective information (average parallelism, running time and associated processors).

Downey uses two different prediction techniques based on statistical estimators; the mean and the median. Downey expects to predict the time until an additional resource becomes available. The main characteristics of the two estimators are:

- Downey uses an equation that computes the median of the remaining queue time exactly by enumerating all possible outcomes (which jobs complete and which are still running), and that calculates the probability that the request will be satisfied before a given time (t). The equation uses the cumulative distribution function of the distribution lifetimes (that by construction of the workloads is previously known: a uniform-log distribution). Finally, the median value is found by setting this probability equal to 0.5 and solving it for the median queue time, obtaining the value of t.
- Using the equation that calculates the probability that a given job will finish before a time t, Downey approximates that the conditional cumulative distribution indicates what fraction of a job's processors will be available at this time.

Predictor analysis: It is mainly focused on the malleable jobs model. The applications are characterised describing its speed-up on a family of curves that are parameterised by a job's average parallelism and its variance. The author researches two kinds of jobs; jobs with high variance and jobs with very low variance. The workloads used in the experiments and simulations are based on these typologies. It does not take into account other application characteristics that can significantly modify their behaviour (MPI, OpenMP or OpenMP+MPI applications, distributed applications, dependencies, etc).

The solution was evaluated using abstract workload models focused on a specific group of clusters in computing centres. In those evaluations it was shown that using predicted queue times to choose cluster size reduces the turnaround time of individual jobs, even with inaccurate predictions. For instance, the average time saving per job is 13.5 minutes while the average job duration is 78 minutes.

Publications: [53][54]

2.1.7. Dinda: Time Series for Host Load and Application Runtime Prediction.

Description: Dinda proposes the prediction of application performance using the correlation of the running time of compute-bound tasks on a host and the computational load on the host. The solution estimates the host load and the application runtime. To do so, Dinda uses two different types of input values:

- Historical information about the host load of the jobs.
- The t-nominal time of the application to be predicted is needed. The t-nominal is the time that a given tasks would take to complete on an unloaded host.

This research is not focused on a specific type of applications. The main idea is to use the computational availability of the host to predict the applications runtime. The prediction method uses time series techniques. The main characteristics of the prediction methods are:

- Time series: autoregressive, moving average, autoregressive moving average models, and autoregressive integrated moving average.
- A simple ad hoc windowed mean predictor called BM. This also includes a function that predicts that the last measurement of load called LAST.
- The time series allows to make n-step-ahead predictions which implies that future prediction t+k, where t is actual time, is also based in the future prediction of t+k-1.
- The prediction quality is evaluated using the mean squared error, which is the average of the square of the difference between predicted values and actual values. It is important to remark that there is a mean squared error associated with every lead time k. Two-step-ahead predictions (k=2) will have a different (probably higher) mean squared error than one-step ahead predictions, and so on.

Predictor analysis: The predictions are returned with an interval of confidence. The returned interval will increase if the distance from the actual time until the predicted time increments.

Publications: [55][56][57][58][59][60][61]

2.1.8. Smith: Statistical Methods using Templates for Categorisation of Applications

Description: Smith's work had a highly significant repercussion in the area of performance prediction. Smith predicts a job's runtimes using historical information. The history is clustered by classifying jobs using *catalogues*. *Catalogues* are defined as a number of data fields that are used to filter the historical information to retrieve a group of records from the workload data. Smith's work starts on Downey's [53][54] research and improves Gibbons [61] work. These authors also propose predictions using catalogues and statistical estimators. However, in those cases, the catalogues are statically modelled producing an inaccurate prediction technique in dynamic environments.

Smith's proposal says that similar applications are more likely to have similar runtimes than applications that have nothing in common. Therefore, instead of using fixed models, Smith proposes the use of data mining algorithms to select the historical data and predict the response time of a job. This technique applies non-restricted catalogues and statistical estimators. The catalogues are created applying search techniques for identifying good patterns for a given workload. The predictions are carried out using the jobs as input parameters that are matched to the different set of catalogues, and computing the mean estimator and the linear regression to correct the previous errors.

Predictor analysis: The evaluation in this proposal has been performed using the workload of Argone National Laboratory (ANL), the Cornell Theory Center (CTC) and the San Diego Supercomputer Centre (SDSC). The performance prediction results are 30 percent better than the results obtained with Downey [53] and Gibbons [61]. However, the flexibility offered by Smith's method introduces potential errors that are produced by the use of any catalogue together with data generated in heterogeneous environments, such as Grid computing. In this scenario a job has a high risk of being predicted by a set of non related historical data resulting in too many unrelated jobs being grouped together.

In [62] Smith presented a method that selected the catalogue with the smallest confidence interval for the job response time. A confidence interval gives an estimated range of values which is likely to include an unknown population parameter. As a result, analysing the confidence interval for each catalogues can provide an indication of how spread the records are. Selecting the catalogue with the smallest confidence interval ensures that it contains records whose response time is within a close distance, and therefore, it may be concluded that it provides a more accurate measure of the job response time. However, even if the smallest confidence interval guarantees an answer taken from a small value range, it does not guarantee

that it selects the right historical records when a heterogeneous data source is used. Therefore, Smith's research is more suitable for homogeneous computer clusters.

Publications: [62][63]

2.1.9. eNANOS: Statistical Analysis of Workloads to Backfill Scheduling Techniques

Description: eNANOS is based on statistical and data mining techniques that other authors like Smith [62][63] have proposed previously. In this case, the objective is to apply data mining techniques to backfill scheduling techniques. The studies and techniques presented have been designed to be applied to local centres and to specific clusters of computers.

The predictors developed in this scope are mainly focused on predicting the performance of the following variables related to the job executions:

- The run time of the jobs.
- The memory required by the jobs.
- The queue of the system to which the user would submit a given job.

The prediction is intended to be used in distributed systems and becomes useful in situations where the broker or scheduler considers a target job with several different possible submission queues. There are two different sets of inputs that vary depending on the objective of the predictor:

- The job identification and the user are required in order to predict the queue. However, as it is internally constructed as a decision tree, it needs feedback information to update or recomputed its internal structure. Thus, the user/scheduler has to provide the workload feedback once the jobs are executed; this workload format is based on the standard workload format proposed by Feitelson.
- The job executable/script, the user and the number of processors (optional) are required in order to predict the memory and the runtime. Similarly to the queue predictor, the appropriate entity of the system has to provide feedbacks about the different entries of the workload of the centre about to the jobs are being finished.

The prediction methods use statistical estimators such as mean, median, or linear regression. The historical information of the workload is used to estimate such values. However, when a prediction is required, not all of the historical information is used for the computation (each entry of this historical data contains information about the execution of given job in the

Standard Workload Format: user, group, number of processors, run time, etc.). The entries of historical data used for the prediction are selected using a set of static catalogues.

- The queue which to submit the jobs to is predicted using the data mining algorithms of decision trees. Two different sets of decision trees are used the CD5 [64] and ID3 [65].
- The job runtime predictor is implemented using the clustering algorithm of the K-Nearest Neighbours, K-Means and X-Means. In this case, instead of using static catalogues the predictor creates a set of job typologies using the mentioned algorithms. Each time that a prediction is required for a given job, the statistical estimators presented previously are used in the set of jobs that belongs to the same group that the given job.

Predictor analysis: The evaluation of the predictors has been carried out using workloads from the Barcelona Supercomputing Centre.

The memory usage prediction was accurate for 35 percent of the applications (predicting with an error of less than 5 percent in almost all its executions), the prediction was poor in 30 percent of the cases (predicting with an error smaller than 80 percent in almost all their executions), and the 35 percent left was spread among the other percentages in between. With regards to the runtime, 50 percent of applications were predicted with an error smaller than 55 percent, and the rest of applications were predicted with bigger errors.

Publications: [66][67][68][69]

2.1.10. NWS: Statistical Analysis of Time Series for Predictions of Network and Computational Performance

Description: The NWS (Network Weather Service) model uses statistical analysis of time series to generate short-time performance predictions of various network and computational resources.

The NWS system is composed of four processes: Persistent State process, Name Server process, Sensor process and Forecasting process. Sensors distributed in the environment periodically measure monitored resources and send the measurements to the Persistent State process, where they are stored with a time stamp. Network sensors are grouped in Cliques and rely on active network probes, which gather: TCP connection time, end-to-end round-trip latency, and bandwidth. Cpu sensors gather cpu load characteristics using system utilities (vmstat and uptime) and additionally run active probes. The Forecaster gets the time series needed from a Persistent Storage and performs the forecast using all the methods, the output of the method with the lowest error is then used as a forecast.

Estimated values:

- CPU availability
- TCP end-to-end throughput
- TCP end-to-end latency

Predictor input: Previously measured values with a time stamp (time series): cpu usage, memory usage, TCP end-to-end bandwidth and latency, connection time. All the above values are gathered by monitoring sensors and stored by the system.

Prediction method: There are several statistical methods for generating predictions based on previous measurements which are described in detail in [73]. The groups of methods are:

- Mean-based methods that use some estimate of the sample mean,
- Median-based methods,
- Autoregressive methods.

Predictor analysis: The accuracy of all predictors is evaluated using prediction error as an accuracy measure. The predictor exhibiting the lowest cumulative error measure at any given moment is chosen to generate a forecast. In this way, the NWS automatically identifies the best forecasting technique for any given resource.

The prediction quality of the presented methods varies for different resources; the mean-based predictors are better for throughput time series, and the median-based predictors are better for latency time series.

Publications: [70][71][72][73][74][75]

2.2. Prediction Solutions Survey Analysis

The solutions presented in this survey have contributed to the understanding of the prediction field in distributed environments. All of them have been, under certain circumstances, tested and validated in a distributed environment. The question that arises at this point is whether there is a real need to propose another method in the same field. The intention of the coming analysis is to answer this question through the study of the level of efficiency and relevance of the aforementioned methods in production Grid environments. The analysis will help to observe which of the methods are the most suitable.

To put this analysis into context, the two most relevant restrictions for prediction methods found in production Grid environments are enumerated:

- The components of Production Grid environment evolve. This evolution needs a prediction method that should be able to adapt to changes in Services and Resources. Not only may

new Services and Resources appear or disappear in Grid environments, but also the available input data for the prediction method may also change in quality, quantity and shape.

– The process used to retrieve the input data should be non-invasive with regards to the Grid sites, especially when dealing with the security policies implemented in the Grid sites.

The present survey started with three methods, each of them presented mathematical models relating different input fields. The EPFL model [33] selects the most appropriate Grid resource for a given job via the characterisation of applications and machines using a single value called Gamma. The LaPIe [37] model has the objective of predicting the scheduling that minimises the overall communication time. DIMEMAS [45] characterise Grids of parallel machines by the use of several parameters, as network latency and bandwidth, organisation of the parallel machine (number of nodes, processors per node, internal latency and bandwidth).

The disadvantage common to all these solutions is that, in order to produce a predicted result, they need the entire model's input fields to be available. Also, all the models require a precise normalisation of the input data for assuring the accuracy of the resulting conclusions. Consequently, it is difficult to implement any of these solutions in dynamic and heterogeneous Grid environments.

Following the same idea, ASKALON [51] presented a mathematical model together with several other components that interact with each other to help in the creation of a normalised input data. As a result, the ASKALON performance prediction service offers a worst case accuracy of 90 percent. However, this level of precision comes at a price: ASKALON is invasive towards the Grid sites; every component of the ASKALON architecture must be in place to obtain the inputs for each of the phases (experimental design, automatic training and performance prediction).

All the solutions (EPFL, LaPIe, DIMEMAS and ASKALON) present an ad hoc explicit relationship among different input parameters to compute a prediction function. The processor type and speed, the quantity and type of internal operations of the Jobs, and the bandwidth and speed of network communications are often associated in an ad hoc manner to forecast a job or network behaviour. These solutions adapt the models to the input data to reach a final prediction method (Figure 4, Approach I). As a result, these are effective solutions as long as the required input data is available and normalised. However, the models need further ad hoc intervention in order to cope with the evolution of Grids, and they are intrusive with regards to the Grid environment.

Using a different perspective, (Figure 4, Approach II) other solutions use a general model in preference to ad hoc relationships between input fields. These solutions choose a well-known data-mining or mathematical model that is subsequently applied to the input data. For instance, the NWS [70] model uses statistical analysis of time series to generate short-term performance predictions of various network. Peter A. Dinda [55] also puts forward the usage of the time series and a windowed mean for carrying out host load estimations and job runtime prediction. Following Dinda's idea, Allen B. Downey [53] used a speedup characterisation model for each job. By relaxing the ad hoc relationships that are used by the methods initially discussed, Dinda, Downey, and NWS managed to discover more information inside the input data and gain further knowledge about the distributed environment. Nevertheless, they are still not able to automatically adapt to significant input data changes.

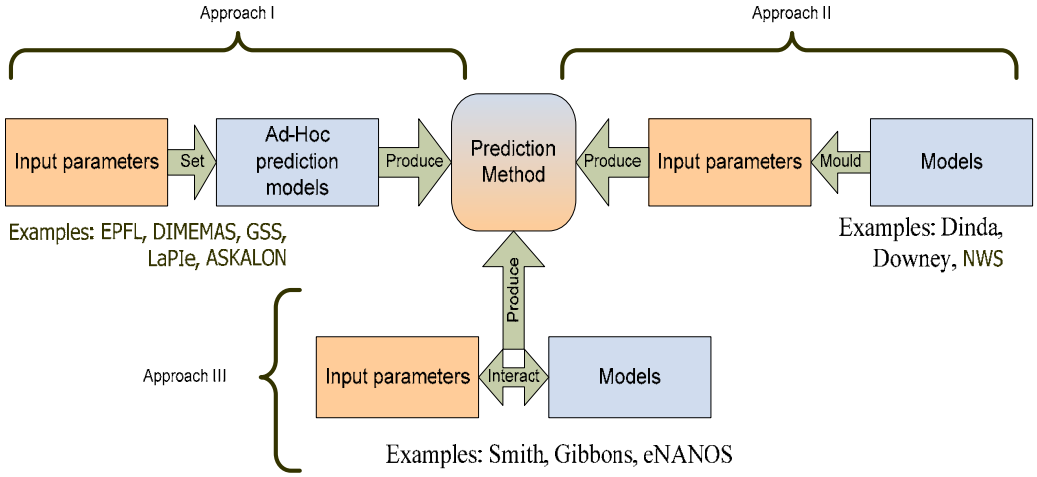


Figure 4 Prediction methods' approaches.

Going one step further, another approach was presented where the information is discovered using data mining methods and the prediction models are changed accordingly. These types of solutions are especially interesting in the evaluation framework because they can be applied to different kind of data independently of its nature (contrary to mathematical algorithms where a normalisation of input fields is needed). Moreover, the most interesting characteristic is that these types of solutions are able to derive or discover new knowledge and, using this, it can produce an autonomic learning mechanism. Therefore, these techniques are highly suitable for deriving information and knowledge in Grid environments due to their volatilised nature (Figure 4, Approach III).

eNANOS [66] presents an early example of using statistical and data mining techniques together with models. The main disadvantage of eNANOS is that its estimators need a

mandatory set of data fields in order to work. This restriction is not presented in Smith's solution [62], which is the most relevant example of this type of solution as it produced the most accurate result. Smith's method is mostly based on Gibbons' work [61] which consisted of a static clustering of the workloads and a later usage of the mean and median inside a cluster. The difference between these analyses is that Smith, unlike Gibbons, uses all possible combinations of data fields and statistical estimators. The historical records are selected applying search techniques for identifying good patterns in a workload trace for a given job.

2.2.1. Is another response time prediction method needed?

In general, the major challenge to current solutions is the lack of control of the quantity, type, and quality of the input data. This challenge presents the most significant restriction when implementing a method in a Grid environment. Many of the requirements that a prediction method would like to demand cannot be always translated to production Grid computing sites, as these sites by definition are loosely coupled, heterogeneous, and geographically dispersed, and therefore lacking a central management.

From the above analysis one can conclude that any prediction solution implemented for a Grid environment should rely on dynamic input data gathering, which is non-invasive with regards to the Grid sites' security policies.

Historical workload traces are used by [53][54][61][62]. Of them, Smith's idea [62] has proved to be the most effective and accurate method. Smith is able to deal with distributed environments without much intrusion and without imposing input data fields restrictions. All the solutions referred to have been effectively tested and compared in the Argonne National Laboratory and the Cornell Theory Center. However, unlike a production Grid environment, each of these workloads is internally homogeneous with a fixed distinctive set of fields. Therefore, the question is: *how well these methods may perform in a dynamic environment*?

To answer this question, a detailed analysis of Downey, Gibbons and Smith's solutions in a production Grid environment [76], the National Grid Service (NGS) [77] was conducted. The motivation behind this effort was to assess how well these techniques perform, and whether a different method can be applied to obtain a better result.

The experiment proves that out of Downey, Gibbons, and Smith solutions, Smith's produced the most accurate prediction results. However, having said that, Smith's predictions were inaccurate. In 50 percent of the job predictions, the average error was 47 percent. Even more, the other 50 percent of the job predictions presented an average error of 106 percent.

The study revealed that Smith's idea is prone to produce prediction errors when non-normalise data is used as input parameters. Smith initially uses any possible combination of data fields to propose historical information filters. Out of them, only one filter is selected. The selected filter must have the minimum response time confidence interval. The study showed that many prediction requests selected filters with a minimum confidence interval that put together non-related historical data records. As a result, prediction errors were produced.

A case to prove this argument is given by a filter composed of two fields, the user identification and the Grid node. The results showed that, according to the solutions, these two fields were the available choice to predict an important number of response time requests, without taking into account the jobs' identification. Consequently, two different jobs were treated together because they were submitted by the same user in the same Grid node. This experiment showed that the freedom of data field selection that worked well in a cluster of computers must be restricted to neutralise the heterogeneous data sensitivity issues.

The study also exposed the fact that using any field without restriction to select a set of historical data disregarding the weight in information that they have, resulted in non-important fields being put together to predict a job response time. As a consequence, this situation generated misjudgements in response time predictions. For instance, the same study showed that many of the fields provided by the Grid sites were not carrying any relevant information, and that some fields included ambiguous information. It was detected that 46 percent of the jobs have names or identifications assigned by the Grid middleware, instead of by the Grid user, and 14 percent of the jobs are either the same executable using slightly different names or using the same name for different executables. This point exposed a second constraint not covered by current solutions: a preliminary data mining analysis should be performed to understand the type and quality of data that a Grid node provides and thereafter define a set of restrictions based on them.

Furthermore, the study revealed that none of the methods considered any information hidden in the data fields. For instance, it is a common Grid user routine to include a set of argument inside the executable script. This information is unseen and not available to Smith's analysis. However, an input argument change may produce a different response time that Smith cannot consider for future analysis.

These results suggest that a dynamic and non-intrusive response time prediction method is needed to improve the disadvantages of the solutions previously presented.

	Runtime	Queue waiting time	Resource requirements	Resource load	Communication time
EPFL Gamma Model	+		+		
LaPIe					+
DIMEMAS	+				+
Modeling Workloads for Grid Systems	+		+		
ASKALON	+	+			
Downey		+			
Dinda	+			+	
Smith	+				
eNANOS	+	+	+		
NWS				+	+

Table 3 Parameters estimated by the prediction solutions presented in the survey

2.3. Chapter Summary and Conclusions

In this chapter several Grid prediction solutions have been presented. Certainly, these solutions have contributed to the understanding of how a job behaviour can be predicted in distributed environments and they also have achieved accurate prediction results during certain circumstances. The challenges that these solutions confront in production Grid environment was also discussed. (See achievements and challenges summary in Table 4)

The solutions that have ad hoc models adapted to the input data (Figure 4, approach I) put emphasis on the reliability and stability of the measurements where a great deal of work is done on establishing the data validity and data model population. In general, when the validity and data model population issues are solved, these solutions are accurate. Nevertheless, in Grid computing environments the required data is not always available and, to obtain it, intrusive actions need to be taken.

Other solutions that use well-known data mining or mathematical models tend to adapt the input data to the models (Figure 4 approach II). They are able to discover further information in the Grid environments but, the solution is no longer viable when the input data cannot be adjusted to the proposed model.

Alternatively, another type of solutions has models that interact with the input data. The models are adapted to the data and the data is adjusted to the models (Figure 4 approach III). These solutions offer the most suitable possibilities in Grid environments provided that a set of restrictions in the selection of fields and historical data records based on the type and quality of

the input data are in place. These solutions have been tested and analysed in a production Grid environment [76].

From this analysis, the conclusion is that another response time prediction method is needed to improve the prediction results obtained by current solutions. The new method should be able to understand the type and quality of data that a Grid node provides. It needs to put the required data field's restrictions in place to neutralise the heterogeneous data sensitivity issues. And also the method must be able to discover any hidden information in the data fields.

	Solution achievements	Challenges faced in production Grid Environments
Gamma Model	Accurate predictions.	Each component should be well balanced. Function cost needs all input parameters.
LaPIe	Uses few network parameters such as latency and bandwidth. Divides the prediction scope into known sections.	Only collective communication performance is predicted.
DIMEMAS	Off-line evaluation. Simple to run parameter studies.	Requires manual configuration of nodes and network models. Requires manual mapping of applications into nodes
Modeling Workloads for Grid Systems	First approach to study submission patterns and user behaviour.	Parallelism of a job required. User-estimated job length/runtime required.
ASKALON	Able to predict Grid sites with different problem sizes under different loads.	Part of a Grid Application Development and Computing Environment. Requires all ASKALON components in the Grid environment.
Downey	New technique to describe speedup on a family of curves parameterised by a job's average parallelism and its variance.	Models only malleable jobs. Requires current status of the cluster.
Dinda	Uses historical information about Grid sites load.	Mathematical approach uses manly time series techniques. Sensible to non-normalised data.
Smith	The most dynamic and non-intrusive solution. Uses dynamic catalogues and statistical estimators.	Applicable to clusters. Problems with non-normalised data.
eNANOS	Demonstrated that prediction methods (such as Smith's) can be used to backfill scheduling techniques.	Applied only to the scheduling process.
NWS	Novel idea of uses of statistical analysis of time series to generate short-time predictions.	Sensors distributed in the environment periodically measure monitored resources and send the measurements.

Table 4 Achievements and problems of reviewed solutions in Grid Environments

Chapter 3 – The Nature of Grid Computing Historical Data: GRTP Implementation Scenario

The objective of this chapter is to illustrate the special characteristics that production Grid environment workload traces offer. These characteristics should be used to set the implementation scenario for Grid response time prediction solutions. The objective is achieved by undertaking a detailed data analysis generated by three different production Grid sites. The result exposes a set of intrinsic rules and dependencies that a response time prediction method ought to consider. The conclusion of this chapter is used as a foundation for the decision making process of the proposed GRTP method.

3.1. Production Grid Computing Testbeds

The Grid testbeds selected to analyse the nature of its workload logs and validate the proposed GRTP method was provided by The Grid Workload Archive [78]. The Grid Workload Archive makes available anonymised workload traces from different Grid environments and provides a virtual meeting place where practitioners and researchers can exchange Grid workload traces.

From the available list of Grid sites, three different production testbeds were selected: Grid'5000, DAS, and AuverGrid. The reason for selecting these particular testbeds was to cover a wide spectrum of possible scenarios found in production Grid environments (refer to Table 5 for the characteristics of each Grid site). The selection is justified as follows:

- Grid5000 was selected because it contains a significant number of distributed heterogeneous Grid sites. Each site contains several clusters of computers and an important number of jobs. Overall, Grid5000 is the largest Grid available in terms of nodes, jobs and users. Grid5000 also features a distinctive heterogeneous environment with different type of resources distributed in a wide area.
- Following the idea of using a diverse testbed selection, DAS was chosen as it represents a mid-size Grid environment in terms of number of nodes and hosted jobs, but has a high level of utilisation rate based on the number of jobs per number of processors. In contrast to Grid5000, DAS is less heterogeneous and it is based in a physically smaller geographic area.

– AuverGrid was chosen as it is a testbed that sits in between the previous two Grids in number of processors and users, but has a lower rate of utilisation. AuverGrid is more heterogeneous and distributed than DAS. Therefore, AuverGrid can be seen as a heterogeneous environment with a high level of available computing capacity.

Testbeds	No. Sites	No. Processors	No. Users	No. Jobs
Grid5000	9	5000	1000	>1M
DAS	5	200	500	>1M
AuverGrid	5	500	500	500K-1M

Table 5 Selected testbed characteristics

3.1.1. Grid5000

Grid5000 is a Grid platform consisting of 9 sites geographically distributed in France. Each site comprises one or several clusters, with a total of 15 clusters inside Grid5000. Within these sites, 17 laboratories are involved nationwide with the objective of providing the Grid research community a testbed allowing experiments in all the software layers between the network protocols up to the applications for research in Grid Computing.

The Grid5000 backbone network infrastructure is provided by the French National Telecommunication Network for Technology, Education and Research (RENATER). The design of the interconnection between the Grid5000 sites has been addressed within the RENATER backbone using an Ethernet over MPLS (EoMPLS) using a full mesh topology solution.

Grid5000 users can run Grid experiments in real life conditions, addressing critical issues of Grid system/middleware, such as programming, scalability, fault tolerance and scheduling. Users can investigate innovative approaches using Grid Service aggregation, peer-to-peer solutions, desktop Grids and active Grids.

Grid5000 employs a high level of security where the communications between sites are isolated from the Internet and vice versa. It provides a software infrastructure allowing users to access the Grid from any site and have a simple view of the system using a single account. Consequently Grid5000 can be seen as a vast cluster of clusters.

Figure 5 Grid5000 sites topology

Grid5000 was selected because of its large amount of nodes, geographically distributed site topology and intensive utilisation. Grid5000 represents a typical heterogeneous Grid, where it is possible to find a large variety of processors, type of network, model of nodes and storage capabilities. The workload trace is composed of 1,020,195 job submissions within a 30 month period.

3.1.2. DAS

DAS (Distributed ASCI Supercomputer) is a wide-area distributed Grid designed by the Advanced School for Computing and Imaging, located at the Delft University of Technology in the Netherlands. The objective of DAS is to provide a common computational infrastructure for researchers within the Advanced School for Computing and Imaging. The researchers work on various aspects of parallel, distributed, and Grid computing, and large-scale multimedia content analysis. DAS is used for researching parallel and distributed computing by five Dutch universities: Vrije Universiteit Amsterdam, University of Amsterdam, Delft University of Technology, University of Leiden, and the University of Utrecht.

DAS consists of four clusters located at four different universities. The first cluster contains 128 nodes and the other three clusters have 24 nodes (200 nodes in total). The nodes within a local cluster are connected by a Myrinet network, which is used as a high-speed interconnect, mapped in user-space. In addition, Fast Ethernet is used for the OS network (file transport). The four local clusters are connected by wide-area ATM networks so the entire system can be used as a 200-node wide-area distributed cluster.

Figure 6 DAS sites topology

DAS was selected in order to test and validate the response time solution in a Grid environment that contains a small amount of distributed nodes with an intensive utilisation. Even though its size in terms of number of nodes may seen small in comparison to Grid5000, the fact that DAS is a less heterogeneous environment but with a higher utilisation intensity compared to Grid5000, makes it ideal as a testbed that contrasts with Grid5000

The DAS workload trace is composed of 1,124,769 job submissions within a 20 month period.

3.1.3. AuverGrid

AuverGrid is a production Grid platform consisting of 5 sites geographically located in the French Auvergne region. The objective of the AuverGrid platform is to deploy a Grid resource centre based on a regional scale to benefit public and private partners. The main idea is to meet the needs of researches, public institutions and companies in the Auvergne region. The AuverGrid project is also part of the EGEE project (Enabling Grids for E-science in Europe).

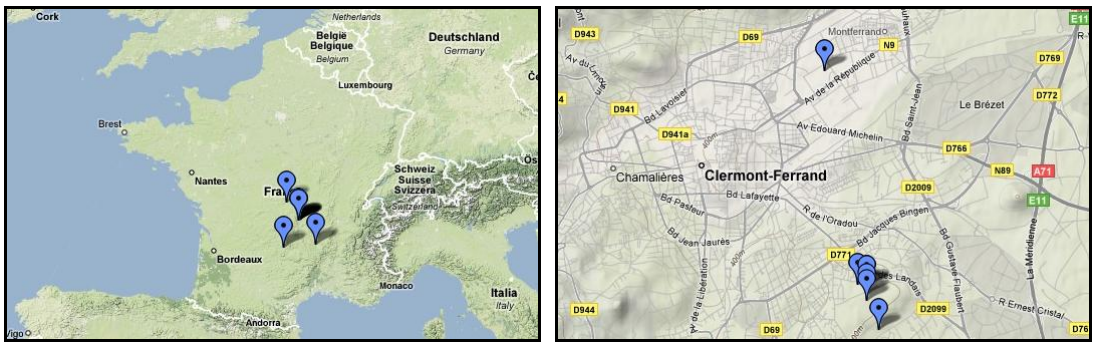


Figure 7 AuverGrid sites topology

AuverGrid was selected because it sits between DAS and Grid5000 in size and composition. AuverGrid contains an important geographical concentration of clusters within a single site, having the rest of the nodes distributed across the region. The utilisation level is the lowest among all selected testbeds, but AuverGrid contains more nodes than DAS. This composition makes AuverGrid a mix of the two previously presented testbeds. The workload trace at AuverGrid is composed of 404,176 job submissions within a 20 month period.

3.2. Proposed Workload Trace Definition

The area of workload normalisation and definition has been extensively studied by Feitelson, a pioneer in this field. Feitelson carried out several studies for computer systems evaluation [22] and presented a number of log analyses for different high performance computer centres in [23], and a general job modelling in [24]. Furthermore, Tsafrir produced subsequent work in the area of log pre-processing and cleaning before moving on to the analysis phase in [25]. Both authors highlighted a set of phenomena, such as workload flurries, that should be taken into account when analysing workloads. Additionally, they proposed a set of techniques for identifying and filtering such anomalies.

Feitelson's work evolved in the definition of a Standard Workload Format (SWF) [79]. SWF is defined as a way of easing the use of workload logs and offering, as a result, a workload library in a portable format. However, given that the SWF is focused on parallel job submission, it lacks support for Grid computing features. For that reason the Grid Workload Format (GWF) [80] was created as an extension of the SWF with the objective of providing a virtual meeting place where practitioners and researchers can exchange Grid computing traces.

Both approaches have in common that they define the data fields beforehand. The Grid sites require an ad hoc mapping of their internal workload traces to populate the SWF or GWF, discarding any other data that is not included in the definition. SWF and GWF are primarily using a fixed number of fields because they were created to help most type of research topics in the computing area without focusing on any particular case.

However, workload trace formats can be efficiently defined by thinking in advance about the requirements of the solution that will use them. In Grid response time prediction, the idea is to make the most of any available data that can potentially boost the quality of the solution without restricting the number of fields.

For instance, in the SWF and GWF definitions the QueueID can be read as the representation of the system queue where a job has been submitted. This definition is easily deduced from the field's name. But, it is not possible to see how much information this field provides. For

instance, if a new data field called ServerRoomStatus starts to be published by a Grid site both the SWF/GWF should discard this new entry. Despite the fact that the ServerRoomStatus may express different meanings, such as the temperature of the server room or the status of the electric power supplier, it may offer information that is as relevant as any other field defined by the SWF or GWF.

From the response time prediction point of view, an important aspect is to understand how much information the data fields offer irrespective of the importance given by the name of the field. Given that the workload trace format provided by the production Grid testbed used in this research is dynamic; the available data fields are diverse, and the fields may appear or disappear without previous indication, an approach that can fit any workload trace is taken. The workload trace, as exampled in Table 6, is proposed with two identifiable sections:

- A workload record header composed of
  - Grid identification: A Grid site identification
  - Job identification: A unique name or identification of the job. A job identification is usually given by the owner of the job. The identification is typically a label composed of numbers and characters.
  - Job execution number: A number that indicates the historical sequence of submission in the workload log. A job identification can have several job execution numbers in a Grid workload.
  - Job response time: the time that the job has taken to complete.
- A workload record body is composed of a list of fields and value pairs.

[*gid*	*jid*	*jen*	*jrt*	(f_k	v_k)]
Grid5000	app578	1097	372	GroupID	group6
				QueueId	queue7
				PartitionID	G1/site6
				Status	1
Grid5000	app578	1104	395	GroupID	group6
				QueueId	Queue10
				Status	2
				Nproc	14

Table 6 A workload trace instance example

Definition: A workload trace (wT) with q records is defined as $[gid, jid, jen, jrt, \{(f_0, v_0)..(f_k, v_k)\}]_{1..q}$ where the gid is the identification of the Grid site, jid is the identification of the job, jen is the job execution number, jrt is the job

response time value, and $\{(f_0, v_0)..(f_k, v_k)\}$ is list of fields and value pairs of the workload record body where f_i is the i field name and v_i is the field's instantiated value.

3.3. Workload Trace Data Fields Analysis

Having defined a flexible workload trace (wT), the next challenge is to select the most appropriate historical records that can be used for predicting a job submission. The difficulty in developing techniques based on historical data is that two previously submitted jobs might be compared in several ways. Jobs may be judged similar because they are submitted by the same user, at the same time, on the same computer, with the same arguments, on the same number of nodes and so on. It can be also consider more esoteric parameters such as the executable size or the Grid user's home directory. The objective of this section is to start defining a method that answers this question.

Similar historical records have been applied by Downey in [53][54] for modelling malleable jobs (jobs that can change the number of processors on which they are executing at run time in response to an external command) using the average parallelism of a programme and its variance in parallelism. In the above publications, the described procedure characterises all applications in the workload, then models the cumulative distribution functions of the runtime in each category, and finally uses these functions to predict application runtime.

But, Downey's solution selects similar records using very few data fields and this produces inaccuracies in the response time prediction. To solve this problem in [61] Gibbons introduced the idea of *catalogues*. *Catalogues* are defined as a number of data fields that are used to filter the historical information to retrieve a group of records from the workload data. An example of Gibbons' catalogues is shown in Table 7. After retrieving the data using the catalogues, Gibbons applied the mean and the linear regression to the response time field as final predictor functions.

One of the main problems faced by Gibbons model is that too many unrelated jobs are grouped together as the number of available catalogues is fixed. Gibbons himself suggested that an improvement to one of his solution's would be to vary the catalogues in the experiments to determine the sensitivity of each parameter.

Gibbons' Catalogues
{UserID, JobID, NProc, How long ago the application was executed }
{JobID, NProc, How long ago the application was executed}
{NProc, How long ago the application was executed}
{UserID, JobID}
{JobID }

Table 7 Catalogues used by Gibbons

Consequently in [62][63] Smith presented a solution based on Gibbons' idea where he used a *non-restrictive set of catalogues*. The *non-restrictive set of catalogues* algorithm produces a powerset of the full list of data fields (For instance Smith's column in Table 8). Each catalogue in the non-restrictive set is finally used to filter the historical data before a predictor method (Mean or Linear regression) is applied to the response time of the selected records.

However, in spite of being a practical idea that has been proven with homogeneous workloads (Argonne National Laboratory and the Cornell Theory Center), the non-restrictive set of catalogues method does not produce accurate results in heterogeneous production Grid environments.

The main problem is found in the underlying data structure. Heterogeneous production Grid environments produce non-normalised workload traces. Non-normalised data may be illogical and inconsistent and it can cause a number of problems to the client system. A logical and efficient data structure design is just as critical. A poorly data model may provide erroneous information, may be difficult to use, or may even fail to work properly. The concept of database normalisation was first introduced by Edgar Frank Codd in [81]. Normalisation is the process of efficiently organising data in a database. There are two main objectives of the normalization process: eliminate redundant data and ensure data dependencies make sense. A normalised data set is suitable for general-purpose querying and free of certain undesirable characteristics. Normalised data should be the starting point for further development.

Therefore, the results produced by Smith are affected by the non-normalised input data. The algorithm that selects the filter to choose historical data in the prediction analysis establishes its decision based on the historical data traces. If the algorithm has the chance to employs any possible combination of fields and the historical data is non-normalised, the algorithm may choose a wrong set of filter because the data at a specific moment was, by chance, offering the best prediction result. The outcome is that the selected historical data is not related to the prediction request. This situation produces job prediction results that are difficult to explain. This argument was demonstrated in [76] where it was also concluded that the issue that needs to

be tackled is to restrict the set of catalogues presented by Smith to make sure that the solution produces results that are explainable and non sensitive to heterogeneous data.

Fields	Used in Gibbons catalogues	Used in Smith catalogues
UserID	Y	Y
JobID	Y	Y
NProc	Y	Y
How long ago the application was executed	Y	Y
JobStructureType		Y
QueueID		Y
Class		Y
Loadleveler Script		Y
Arguments		Y
Network Adaptor		Y
Submission Time		Y
StartTime		Y
RunTime		Y

Table 8 Gibbons and Smith data fields used in their catalogues.

Also, in [76] a first idea of restricting the non-restrictive set of catalogues was tested using two set of fields; a *binding level* that concentrates only on the relevant Grid fields, and an *extended level*, that uses the remaining set of fields. The binding level contained the fields UserID, JobID, and JobParameters**.** The extended level contained the remainder of the available fields. In this case, the catalogues are created using one or more fields from the binding level together with zero or more fields from the extended level. This approach, pictured in Figure 8, was tested using the NGS production Grid environment and compared against Smith's solution. The result showed that the restriction partially neutralised the heterogeneous data sensitivity producing in average a 54 percent improvement in the prediction error.

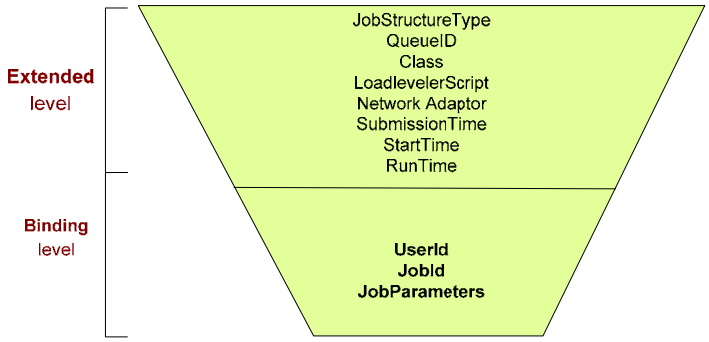


Figure 8 Draft for a workload meta-model

Even if the prediction error was reduced, the method and justification that builds the workload meta-model structure was mentioned as a point to be reviewed in [76]. In Figure 8, the meta-model is defined based on the expert's know-how and understanding of the environment. It was demonstrated that the job identification must be part of the binding level, but it was not clear how many other fields should be included in the model.

At the same time, another point noticed in the experiment was the increasing running cost of the overall prediction method, given that the number of fields affects the prediction service's performance. Even though the job forecasting is not a real-time system, a predicted response time should be provided in an acceptable time. Therefore, reducing the complexity without compromising the quality of the solution is an important factor that must be considered.

The following section presents a workload trace data analysis that aims to demonstrate how the data fields in a workload trace can be quantified and related to another fields based on the field's information. To achieve this objective, formal information theory fundamentals are used to evaluate the quality of information provided by the workload fields. This process sets the foundations of a meta-model creation method.

3.3.1. Information Theory Background

Information theory is a branch of applied mathematics and electrical engineering involving the quantification of information. Claude Elwood Shannon founded information theory with a paper published in 1948 [82]. Since its origin, information theory has broadened to find applications in many areas, including statistical inference, natural language processing, and cryptography. In the following section a set of information theory definitions is presented to understand the subsequent analysis of the Grid computing historical data.

3.3.1.1. Entropy

One of the most important features of Shannon's theory is the notion of entropy [82]. When given a system whose exact description is unknown, its entropy is defined as the amount of information needed to exactly specify the state of the system (to the full extent that it can be described in the universe itself). There are more possible states for a system to occupy when it has a higher degree of freedom and more constituents.

Entropy, which can be also seen as a measure of the inherent randomness in a set of observed data, is a key concept in information theory. In this research, the entropy is used to

measure and reason around the complexity of the data values that a specific Grid site publishes for a predictor's input field.

Consider that a Grid site provides a data field with different values, the importance of this field to a response time predictor would depend on the information contained in the different record instances. To encode the field's values, a number of bits can be used in the following way: if the field has only one or two possible values, only one bit would be needed to encode it, but if the field has more values, more bits are needed. In particular, what the entropy value measures is how many bits are needed to describe all the different status that the new field may have.

As investigated and presented in [76], production Grid environments publish data fields that, in some cases, are not carrying relevant information. For example, some fields include repeated values or are ambiguous, as in the case when the job identification is assigned with the same default value by the Grid middleware instead of using the client's identification. These issues have been highlighted as the main reason for why current methods are not performing well in a production Grid environment. Fields with repeated or default values have a lower level of uncertainty and therefore a low entropy. Consequently, a field with different or non-default values (high entropy) have more chances to provide more information than a field with few or one value (low entropy).

Definition: Let X be a discrete random variable with n outcomes $\{x_i : i = 1,..,n\}$ and $p(x_i) \to [0,1]$ denotes the probability mass function of x_i, the entropy $H(X)$ of a discrete random variable X id defined by:

Formula 1). $$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i)$$

The log is to the base 2 and the entropy is expressed in bits. By convention $0 \log 0 = 0$, this is justified by continuity since $v \log v \to 0$ as $v \to 0$. Thus, adding terms of zero probability does not change the entropy.

An example to illustrate the usability of the entropy definition is to consider a Grid site that has one job submission queue in its infrastructure. In this scenario, the job manager must submit all jobs to the only available queue. Therefore, the resulting workload trace queue field for this Grid site has an entropy of 0 (This is calculated using 1 as the field's probability and $-\sum_{i=1}^{1} 1 \log 1 = 0$). If the same calculus is done for another Grid site with eight queues where the job manager equally distributes all the jobs across the queues, the obtained queue field entropy

value is 3. This is because each queue has a probability value $\frac{1}{8}$ and the resulting entropy is expressed as $-\sum_{i=1}^{8}\frac{1}{8}\log\frac{1}{8}=3$ (See Table 9 for a detailed calculus of this second example).

Queue	Queue Probability	Log Probability	Queue x Log
1	0.125	-3	-0.375
2	0.125	-3	-0.375
3	0.125	-3	-0.375
4	0.125	-3	-0.375
5	0.125	-3	-0.375
6	0.125	-3	-0.375
7	0.125	-3	-0.375
8	0.125	-3	-0.375
		Entropy of the Queue:	-(-3)

Table 9 Detailed calculus for the entropy example.

3.3.1.2. Joint Entropy

The definition of entropy of a single variable can be extended to a pair of variables. The joint entropy measures how much entropy is contained in a joint system of two random variables. Since both variables can be considered a single vector-valued random variable, there is nothing new in this definition.

In a Grid environment, joint entropy represents the information that two different fields can provide together to a response time predictor.

Definition: The joint entropy $H(X,Y)$, X of a pair of discrete random variables X with n outcomes $\{x_i : i=1,..,n\}$ and Y with m outcomes $\{y_j : i=1,..,m\}$ and with a joint probability mass distribution $p(x_i, y_j) \rightarrow [0,1]$ is defined as:

Formula 2). $$H(X,Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j)\log p(x_i, y_j)$$

To exemplify the joint entropy definition, a Grid site that has one submission queue and two available cpu type is proposed. If the Grid job manager equally distributes all the jobs across the

two cpu type, the joint probability of the queue and cpu type is $\frac{1}{2}$ and therefore the entropy value is expressed as $-\sum_{i=1}^{1}\sum_{j=1}^{2}\frac{1}{2}\log\frac{1}{2}=1$

3.3.1.3. Conditional Entropy

Conditional entropy is another definition used in this research. Conditional entropy quantifies the remaining entropy (i.e. uncertainty) of a random variable Y given that the value of a second random variable X is known. The conditional entropy plays an important role in a Grid environment when one is trying to understand how much information a particular field can add if another field is already considered by the response time predictor.

Definition: Consider two random variables *X* and Y a pair of discrete random variables X with n outcomes $\{x_i : i=1,..,n\}$ and Y with m outcomes $\{y_j : i=1,..,m\}$ with a joint probability mass function $p(x_i, y_j) \to [0,1]$,a conditional probability mass function $p(x_i / y_j) \to [0,1]$, and marginal probability mass functions $p(x_i) \to [0,1]$ and $p(y_i) \to [0,1]$, the conditional entropy $H(X / Y)$ is defined as:

Formula 3). $$H(X / Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j)\log p(x_i / y_j)$$

The condition entropy is illustrated using two Grid sites together: a Grid site that has one job submission queue and a second Grid site with eight submission queues. In this example it is supposed that the jobs are firstly equally distributed across the Grid sites and secondly equally distributed across the queues in each Grid site. Therefore, not knowing the Grid site in advance, the probability for a job to arrive to the first Grid site is $\frac{1}{2}$, and within this Grid site, the final probability to arrive to the only site queue is $\frac{\frac{1}{2}}{1}=\frac{1}{2}$. For the second Grid site, the probability for a job to arrive is $\frac{1}{2}$ (equally distributed across sites) and within this second Grid site, the

probability for a job to arrive to a queue is $\frac{\frac{1}{2}}{8} = \frac{1}{16}$. However, if the Grid site is known the probability for the first Grid queue is 1 and for the second Grid queues is $\frac{1}{8}$. As a result the conditional entropy value of the queue knowing the Grid site is 1.5. For this example a detailed set of calculus is shown in Table 10.

Grid	Queue	Probability of the Grid and the Queue	Probability of the Queue knowing the Grid	Log of Probability of the Queue knowing the Grid	Queue x Log
1	11	0.5000	1.0000	0.0000	0.0000
	21	0.0000	0.0000	0.0000	0.0000
	22	0.0000	0.0000	0.0000	0.0000
	23	0.0000	0.0000	0.0000	0.0000
	24	0.0000	0.0000	0.0000	0.0000
	25	0.0000	0.0000	0.0000	0.0000
	26	0.0000	0.0000	0.0000	0.0000
	27	0.0000	0.0000	0.0000	0.0000
	28	0.0000	0.0000	0.0000	0.0000
2	11	0.0000	0.0000	0.0000	0.0000
	21	0.0625	0.1250	-3.0000	-0.1875
	22	0.0625	0.1250	-3.0000	-0.1875
	23	0.0625	0.1250	-3.0000	-0.1875
	24	0.0625	0.1250	-3.0000	-0.1875
	25	0.0625	0.1250	-3.0000	-0.1875
	26	0.0625	0.1250	-3.0000	-0.1875
	27	0.0625	0.1250	-3.0000	-0.1875
	28	0.0625	0.1250	-3.0000	-0.1875
				Entropy of the Queue knowing the Grid site:	-(-1.5000)

Table 10 Detailed calculus for the conditional entropy example.

3.3.1.4. Theorem chain rule

The chain rule says that the entropy of a pair of random variables is the entropy of one plus the conditional entropy of the other. This theorem is often used in this research to calculate the conditional entropy using the joint entropy and entropy definition.

Theorem Chain rule:

$H(X,Y) = H(X) + H(Y/X)$ therefore

Formula 4). $$H(Y/X) = H(X,Y) - H(X)$$

3.3.1.5. Mutual Information

Mutual information is a measure of the amount of information that one random variable contains about another random variable. It is the reduction in uncertainty of one random variable due to the knowledge of the other.

Definition: Consider two random variables X and Y a pair of discrete random variables X with n outcomes $\{x_i : i = 1,..,n\}$ and Y with m outcomes $\{y_j : i = 1,..,m\}$ and with a joint probability mass distribution $p(x_i, y_j)$ and marginal probability mass functions $p(x_i)$ and $p(y_j)$. The mutual information $I(X;Y)$ is the relative entropy between the joint distribution and the product distribution $p(x_i)\ p(y_j)$, i.e.,

Formula 5). $$I(X;Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j)\log\frac{p(x_i, y_j)}{p(x_i)p(y_j)}$$

3.3.1.6. Information Diagram

There is an analogy between Shannon's measures of the information content of variables and a measure over sets that assist in the understanding of the different concepts presented above; namely the joint entropy, conditional entropy, and mutual information can be considered the measure of a set union, set difference, and set intersection, respectively [83].

Figure 9 shows a diagram where the previous definitions are presented for two variables X and Y:

- The information provided by a variable X is expressed by one set diagram named H(X).
- The information provided by Y is expressed by another set diagram named H(Y).
- The union of both entropies, represented here as the set union, expresses the information offered by both variables together. This is the joint entropy of X and Y and it is named H(X,Y).
- The conditional entropy of X given Y is seen as what information X can add to the information that Y has already provided. In this diagram conditional entropy of X given Y is the complement of the set Y and it is named H(X/Y).
- And finally, the mutual information of X and Y is the intersection of the two sets. In other words, is the information that may be provided by X or Y, named I(X:Y).

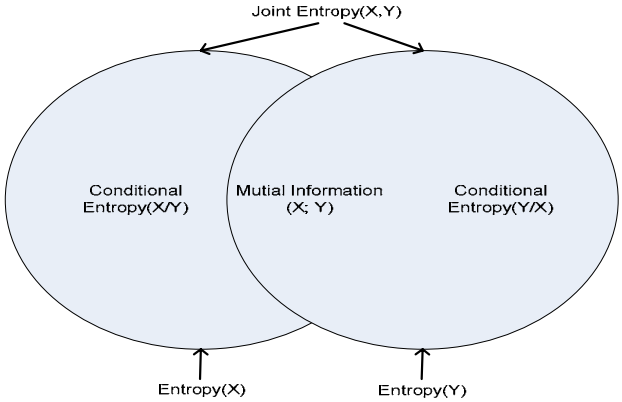


Figure 9 Diagram of entropy, join entropy, conditional entropy, and mutual information

3.3.1.7. Maximum Entropy

Another information concept used in the next section is the maximum entropy. For the subject of this research, the maximum entropy is an important notion that is used to understand how much information a given field can potentially provide to the predictor.

The maximum entropy procedure consists of seeking the probability distribution which maximises the information entropy, subject to the constraints of the information. Entropy maximisation takes place when the probabilities are uniform and the sum of them is one [84].

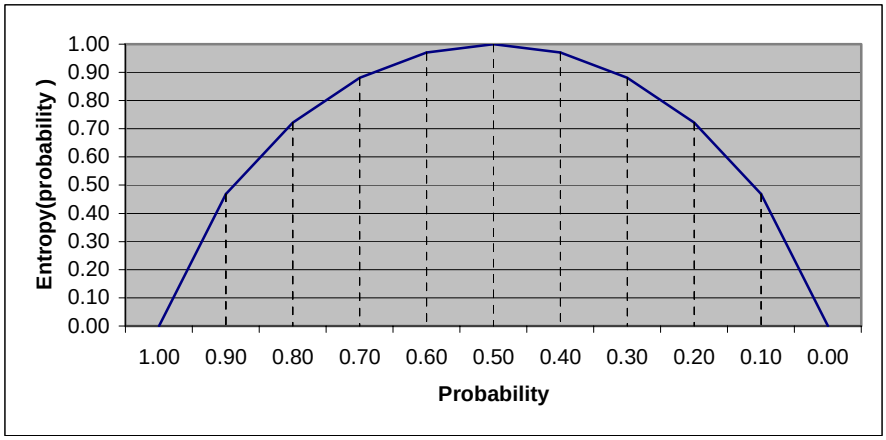


Figure 10 Maximum entropy: Entropy of the probability versus the probability.

Having the maximum entropy noted as H() and the example:

$$X \begin{cases} 1 \text{ with probability p,} \\ 0 \text{ with probability 1-p.} \end{cases} \quad \text{then} \quad \mathrm{H}(X) = -p\log(p) - (1-p)\log(1-p)$$

The graph of the function H(p) is shown in Figure 10. The figure illustrates a concave function of the distribution. In particular, H(X) = 1 when p = 1 / 2, and equals 0 when p = 0 or 1. When p = 0 or 1, the variable is not random and there is no uncertainty. Similarly, the uncertainty is at its maximum when p = 1/2, which also corresponds to the maximum value of the entropy.

The examples presented for the definitions of entropy, joint entropy, and conditional entropy are assuming that the utilisation of the different resources (Grid sites, queues and cpu type) is equally distributed. This situation offers a scenario where the use probability for each component is maximised and therefore the entropy, joint entropy, and conditional entropy were maximised. Certainly, production Grid environments do not usually behave in this way. There are resources with different usability than others. The result of the information theory functions presented in this section can be used to analyse different aspects of the workload trace data. For instance, the maximum entropy can show the potential amount of information that a field or set of fields can provide and the entropy can show the real amount of information that a field is providing.

3.3.2. Workload Trace Fields Entropy and Maximum Entropy

The following analysis aims to show what type of data is available in production Grid sites. The analysis highlights the restrictions that the GRTP method should consider and it also explains why many of the solutions listed in the previous chapter are not suitable for production Grid environments.

It has been previously mentioned that the entropy is a measure of uncertainty of a random variable. The point of zero entropy is set when it is known precisely which state a data instance is. In the case of data fields, this happens when all the values of its record instances are the same. The point of maximum entropy occurs when all the values are uncertain. This is the case when all the possible values are uniformly distributed across the different records.

If this concept is applied to a Grid workload trace data, a field that has all its entries with the same value is certainly predictable and therefore the entropy of this field is zero. For instance, if a Grid site always publishes the QueueID in its workload trace with the same value (because there is a single queue in the system), this field will be worthless for the predictor in terms of quality of information. Given that all jobs will use the same submission queue, the predictor will not be able to search and divide the historical data using this data field. However, if more submission queues are added to the Grid site and the data is published accordingly, the entropy

of this field will no longer be zero and the information of the queue will start playing a different role for the predictor.

The first step in this analysis is to observe how relevant the available fields are. This analysis is performed using the data entropy method (Formula 1) as a measure of information. Given that the entropy value is the number of bits can be used to encode the content of the field, a relevant field should observe a significant entropy value. Therefore, the integer part of the field's entropy number should be enough to analyse its data quality. Having said that, the entropy field results expressed in this research are presented using four decimal places for the fractional part to demonstrate how minimum the information of certain field can be in production Grid environments.

The entropy for a field $f_{t|0\le t\le k}$ that belongs to a workload wT defined as $[gid, jid, jen, jrt, \{(f_0, v_0)..(f_k, v_k)\}]_{1..q}$ is calculated as follows:

- v_t is a random variable with n outcomes $\{v_{t,i} : i = 1,..,n\}$ for the field f_t. The probability $p(v_{t,i})$ is given by the number of times that this value has occurred divided by the current number of records of the workload trace. The probability is calculated in Grid5000 as follows: the workload data for this testbed is composed of 1,020,195 job submissions (q). One of the values of a data field UserID (f_t) is *user144* ($v_{t,i}$).*User144* has submitted 26,006 jobs. Therefore, the probability $p(v_{t,i})$ / $v_{t,i}$=user144 is $\frac{26,006}{1,020,195}$.
- v_t is a variable with n outcomes $\{v_{t,i} : i = 1,..,n\}$ for a field f_t. The entropy of v_t is calculated using the Formula 1): $H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i)$. Following the calculus, the value for the function $p(v_{t,i}) \log p(v_{t,i})$ for the user *user144* ($v_{t,i}$) is - $\frac{26,006}{1,020,195} \log(\frac{26,006}{1,020,195})$. This is one result for the specific value *user144*. In Grid5000, when adding all possible results for every distinct value of the field UserID, the final entropy of this field is 4.4205.

The entropy results for a list of data fields are shown in Table 11. The entropy results calculated in Table 11 show that the UserID is definitely the data field that provides more information, followed by a mixed set of results. In some cases the JobID offers a good quality of

data information and in another case the QueueID field, where the job has been submitted, affords good quality data information.

These results show that some data fields, which one might consider useful to include on an ad hoc basis (e.g. the JobStructure or the Status) in a prediction model, are in reality providing little or no information. This is an important argument that demonstrates that static or ad hoc prediction models can be suitable for one Grid environment but they may fail to produce accurate results in another environment. Another outcome shown in Table 11 is the significant number of data field that have zero as entropy value. The explanation is that all these fields are provided by the Grid environments with the same value in all job submissions.

Finally, these results show that not all the fields have the same quality of information across the different Grid sites. The JobID presents a better data quality in Grid5000 than in AuverGrid. The opposite result can be seen for the QueueId. This point strengthens the idea of having a recurrent method that defines which fields are important for a predictor.

		Entropy	Maximum Entropy	Entropy	Maximum Entropy	Entropy	Maximum Entropy
		Grid5000		DAS		AuverGrid	
Fields	JobID	3.6255	9.9366	2.1468	13.146	0.9710	3.9069
	UserID	4.4205	8.9099	5.0665	8.3794	6.6729	8.6582
	Nproc	1.9753	8.3219	2.9154	6.3399	0.5841	1.0000
	QueueId	1.1681	6.5999	0.0058	1.5850	3.1945	3.7004
	LastRunSiteID	2.5164	3.9069	2.0066	2.3219	2.0545	2.3219
	GroupID	2.2920	3.3219	2.3716	3.585	2.3444	3.0000
	OrigSiteID	2.2920	3.3219	2.0066	2.3219	2.0545	2.3219
	Status	0.9405	2.3219	0.0117	1.0000	0.7176	1.5850
	JobStructure	0.0000	0.0000	0.4491	1.0000	0.0000	0.0000
	PartitionID	0.0000	0.0000	0.0000	0.0000	0.2985	1.0000
	UsedNetwork	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	UsedDiskSpace	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	UsedResources	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ReqPlatform	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ReqNetwork	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ReqLocalDiskSpace	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ReqResources	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	VOID	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ProjectID	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Table 11 Entropy and maximum entropy of the testbeds' fields

In addition to the entropy, the maximum entropy is also displayed in Table 11. The maximum entropy helps to understand the potential information that each data field may have. A field's maximum entropy can be determined by calculating the entropy when all the probabilities are uniform and the sum of them is one (Figure 10). The maximum entropy for a

field $f_{t|0\le t\le k}$ that belongs to a workload $wT=[gid, jid, jen, jrt,\{(f_0,v_0)..(f_k,v_k)\}]_{1..q}$ is calculated as follows:

- v_t is a variable with n outcomes $\{v_{t,i}: i=1,..,n\}$ for the field f_t. The uniform probability of all $p(v_{t,i})$ is given by the number of records of the workload trace divided by n (the number of distinctive values of f_t), and divided again by the number of records of the workload trace. This calculus determines a uniform probability for each value that when added for all of them the result is one. For instance, the workload data for Grid5000 is composed of 1,020,195 job submissions (q) and it has 481 distinct UserIDs (481 is the n value for the field f_t = UserID). This division is 2120.99 that divided again by the number of records gives a uniform probability for all values $\frac{\frac{1,020,195}{481}}{1,020,195}=\frac{1}{481}=2.08x10^{-3}$.
- Thus, the resulting maximum entropy value is calculated using the uniform probability for all distinct values that a field has. In the previous example, the maximum entropy for the field f_t =UserID is 8.9099.

An analysis of the data fields' maximum entropy provides an idea about the potential information that the data field may provide. The potential information of each field can be used to predispose the use of certain data fields in a prediction method. Even more, the actual data field information (entropy) compared against the potential information (maximum entropy) also helps to understand the nature of the Grid computing historical data.

The maximum entropy analysis of the testbeds reveals that half of the available fields potentially do not provide any information, and that the other half only uses part of the maximum potential information that can be provided. The JobID and the UserID are the predominant fields that may possibly offer the maximum quality information in all testbeds following the maximum entropy column. At the bottom of each table, the Status shows that few different stages are presented, as expected, and therefore this field is a candidate to be used by the predictor together with other fields.

As a result, the maximum entropy analysis introduces a new picture; from the initial complete list of data inputs, there are few input fields that may potentially provide relevant information.

Taking the entropy and maximum entropy conclusions into account, the new scenario shows that a response time prediction method that bases its estimations on a predefined or ad hoc

model that contains relationships of the fields based on the semantics of them exposed by their field's names is prone to fail.

3.3.3. Workload Trace Fields Dependencies Using Joint Entropy

This section analyses if there is any interaction and dependency between the workload trace data fields. At a first glance, there are some intuitive pairs that can be understood as dependent, such as a user and the group that it belongs to. Consequently, discovering these relationships and proving how much information each field is adding to another is a factor in gaining a better understanding of the nature of the data generated by production Grid environments.

The joint entropy (Formula 2) measures how much information is contained in a joint system of two variables and therefore can be also seen as how much information is gained by adding another field to the system. The joint entropy for a field $f_{t|0\le t\le k}$ and $f_{r|0\le r\le k}$ that belongs to a workload $wT=[gid, jid, jen, jrt, \{(f_0, v_0)..(f_k, v_k)\}]_{1..q}$ is calculated as follows:

- v_t is a variable with n outcomes $\{v_{t,i} : i=1,..,n\}$ for the field f_t and v_r is a variable with m outcomes $\{v_{r,j} : j=1,..,m\}$ for the field f_r. The probability of any $p(v_{t,i}, v_{r,j})$ is given by the number of times that these two values have occurred divided by the number of records of the workload trace. For instance, the workload data for Grid5000 is composed of q=1,020,195 job submissions and one of the values of the data field f_t=UserID is $v_{t,i}$=user127 and another value of the data field f_r=GroupID is $v_{r,j}$=group6. Both values appear in 1,913 job submissions. The joint probability of them would be $\frac{1,913}{1,020,195}=1.87x10^{-3}$.
- v_t is a variable with n outcomes $\{v_{t,i} : i=1,..,n\}$ for the field f_t and v_r is a variable with m outcomes $\{v_{r,j} : j=1,..,m\}$ for the field f_r. The joint entropy of $v_{t,i}, v_{r,j}$ is calculated using the Formula 2): $H(X,Y)=-\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j)\log p(x_i, y_j)$. Following the previous example the value for the function $p(v_{t,i}, v_{r,j})\log p(v_{t,i}, v_{r,j})$ for the user x_i=user127 and group y_j=group6 is $\frac{1,913}{1,020,195}\log(\frac{1,913}{1,020,195})$. Adding all results for every distinct value, the joint entropy would be 4.4205.

The joint entropy calculated for Grid5000 for the UserID and the GroupID fields have the same value that the entropy calculated in the previous point for the field UserID for the same testbed. This result means that the information contained in the GroupID field is included in the UserID field. In other words, it means that if the UserID field is used in a prediction model, the inclusion of the GroupID would not provide any extra information.

To analyse these relationships between data fields, a further set of experiments are presented. The first test is done using the fields UserID and the GroupID. The entropy and joint entropy of the pair is shown in Table 12 and also a graphical representation is pictured in Figure 11. These numbers show that the Grid user is strongly tied to the Group to which it belongs. DAS is the only Grid site that adds information when the group is also published; the other two sites have the group embedded into the user.

Grid5000			DAS			AuverGrid		
Entropy(U,G)	Entropy (U)	Entropy (G)	Entropy(U,G)	Entropy(U)	Entropy(G)	Entropy(U,G)	Entropy(U)	Entropy(G)
4.4205	4.4205	2.2920	5.0730	5.0665	2.3716	6.6729	6.6729	2.3444

Table 12 UserID (U) and GroupID (G) joint entropy values.

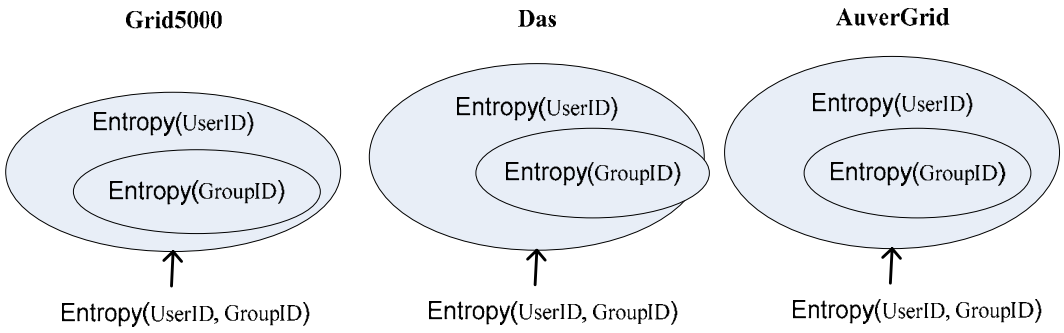


Figure 11 UserID and GroupID entropy and joint entropy representation

Even if the dependency of the UserID and the GroupID is obvious, this impression may be challenged by the concept that the data is coming from a Grid environment where a Grid user may belong to different groups in several Virtual Organisations and therefore break this apparent relationship. But, it was demonstrated that, in this aspect, current Grid computing environments tend to tie users and groups as it is usually done in traditional parallel environments.

To understand which other fields are tied together, the same data analysis is also carried out. This experiment studies the dependency between the JobID and the QueueID where a job was actually run. The joint entropy results are presented in Table 13 and a graphical representation in Figure 12.

Grid5000			DAS			AuverGrid		
Entropy(E,Q)	Entropy(E)	Entropy(Q)	Entropy(E,Q)	Entropy(E)	Entropy(Q)	Entropy(E,Q)	Entropy(E)	Entropy(Q)
3.7717	3.6255	1.1681	2.1518	2.1468	0.0058	3.8786	0.9710	3.1945

Table 13 JobID (E) and QueueId (Q) joint entropy values

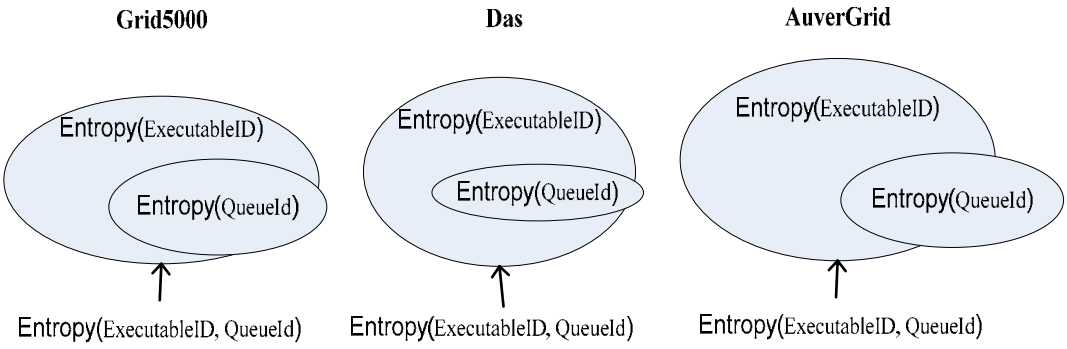


Figure 12 JobID and QueueId entropy and joint entropy representation

This analysis presents a different scenario, where the joint entropy is significantly greater than any sole entropy. A particular situation is presented in this case where the bigger entropy is provided by the JobID in Grid5000 and DAS opposite to bigger entropy given by the QueueID in AuverGrid. Therefore, the JobID almost includes the QueueID in Grid5000 and DAS, being the opposite in AuverGrid.

This experiment shows that even if the dependency is not as strong as in the previous experiment, there still is a link between this new pair of fields. Nevertheless, there is an indication that in some Grid sites the jobs are assigned to different queues for its submissions. But, the most interesting point to highlight from this experiment is that there is not a unique sequence in which to link fields. Each Grid site has different dynamic data fields that are, in entropy terms, most important and that also include another fields.

Having analysed the Grid user and its groups on the one hand and the executable and the queue where they are run on the other, the next step is to see if there is any connection between them. This final experiment intends to show to which extent a Grid user tends to submit different jobs. The joint entropy results are presented in Table 14 and a graphical representation in Figure 13.

The conclusion is that most of the users are usually working with a single application across its Grid computing life cycle. Some of them are varying, but the majority keeps on submitting the same application.

Grid5000			DAS			AuverGrid		
Entropy(U,E)	Entropy(U)	Entropy(E)	Entropy(U,E)	Entropy(U)	Entropy(E)	Entropy(U,E)	Entropy(U)	Entropy(E)
4.8014	4.4205	3.6255	5.4633	5.0665	2.1468	6.9807	6.6729	0.9710

Table 14 UserID (U) and JobID (E) joint entropy values

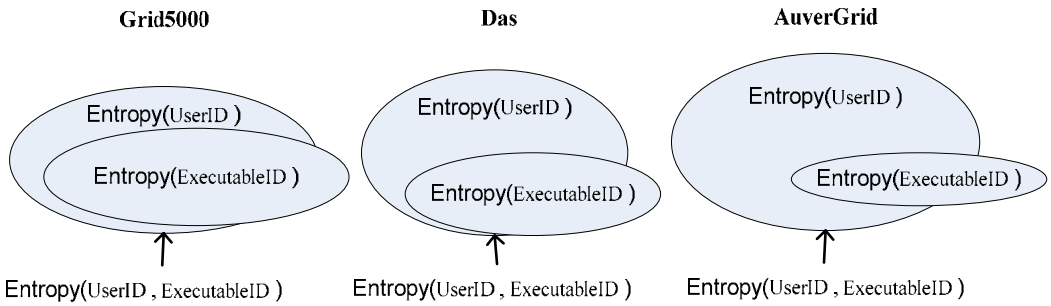


Figure 13 UserID and JobID entropy and joint entropy representation

A final breakdown of values is presented using the conditional entropy method in order to provide a bigger picture. The conditional entropy quantifies the remaining entropy of a random variable given that the value of a second random variable is known (See Figure 9). The analysis intends to demonstrate how much information the addition of another field can represent to the system, and discover if there is a data field that can be used as a centre point to build a data-model that grows depending on the information that is added by other components.

To calculate the conditional entropy of Y and X, the theorem chain rule (Formula 4) is used utilising the already calculated $H(X,Y)$ and $H(X)$.

	Grid5000	DAS	AuverGrid
(X/Y)	H(X/Y)	H(X/Y)	H(X/Y)
UserID/GroupID	2.1284	2.7013	4.3285
GroupID/UserID	0.0000	0.0065	0.0001
JobID/QueueId	2.6035	2.1460	0.6841
QueueId/JobID	0.1462	0.0050	2.9076
UserID/JobID	1.1760	3.3166	6.0097
JobID/UserID	0.3810	0.3968	0.3079

Table 15 Conditional entropy values

The results displayed in Table 15 shows that in real Grid computing environments, there is a high likelihood that if the Grid user who is about to submit a job is known, information about the group the user belongs to will not add much extra value. However, this is not the case when the job and the queue are analysed. In two cases the job goes together with the queue and therefore knowing the job, the inclusion of the queue to the system does not add any significant

information. In AuverGrid, where the schedule policy may be different, the queue adds information to the system.

3.4. Job Response Time Data Profile Analysis

The previous section presented a comprehensive analysis of the quality of information provided by the data fields in a workload trace data. In this section a further data analysis that pictures the profile of a job's response time values is presented.

The previous analysis showed that some data fields provide relative small information (Table 11) or that the information is already included into another field (Table 15). In this scenario, a prediction method can reach a point where the data fields are not able to supply the required information to perform a response time prediction. For this reason, the aim of the next analysis is to deduce what other indications about a job can be extracted from the historical workload record headers (Section 3.2.), which, in turn, are composed of the Grid identification, the job identification, the job execution number, and the job response time.

3.4.1. Workload Header Catalogue Definition (WHC) and its instances (iWHC)

A Workload Header Catalogue (WHC) is a list of fields that can be used to retrieve from a workload trace a list of records composed of a Grid identification (gid), a job identification (jid), a job execution number (jen), and a job response time (jrt). In other words, a WHC is a filter that must have the Grid identification and job identification. Any other field can be included into the WHC but they are not mandatory.

Definition: Consider a workload trace wT as $[gid, jid, jen, jrt, \{(f_0, v_0)..(f_k, v_k)\}]_{1..q}$, a WHC is proposed as $whC(gid)[\{jid\}\{f_0,.., f_h\}]$ where gid is the Grid identification, jid is the job identification, and $\{f_i : i = 1,.., h\}$ are a list of fields.

The WHC can be applied to a workload trace via the instantiation of each WHC defined field. This means that the Grid identification and job identification should have a value assigned as well as the rest of the WHC fields. As a result, an Instantiated Workload Header Catalogue (iWHC) is created when a data value, which will be used as a filter to query the workload trace, is assigned to all defined fields of WHC

Definition: Consider a WHC $whC(gid)[\{jid\}\{f_0,..,f_h\}]$, an iWHC is proposed as $iwhC(gid=v_{gid})[\{jid=v_{jid}\}\{f_0=v_0,..,f_h=v_h\}]$ where gid , jid , and $\{f_i : i=1,..,h\}$ are assigned to data values v_{gid} , v_{jid} ,and $\{v_i : i=1,..,h\}$ respectively.

3.4.2. iWHC Response Time Data Analysis

To graphically represent what an iWHC is, in Figure 14 an example of the Grid5000 is shown. In this instance two filter fields are used: the JobID app162 and the UserID user573.

Figure 14 shows that, for a specific job e.g. user, group, queue, etc, the response time axis of the data chart has several variations. This multi-pattern phenomenon is a common example observed in production Grid computing environments. There may be various reasons why the app162 submitted by user573 performed as shown, but reason for these changes are not contained within the workload data logs.

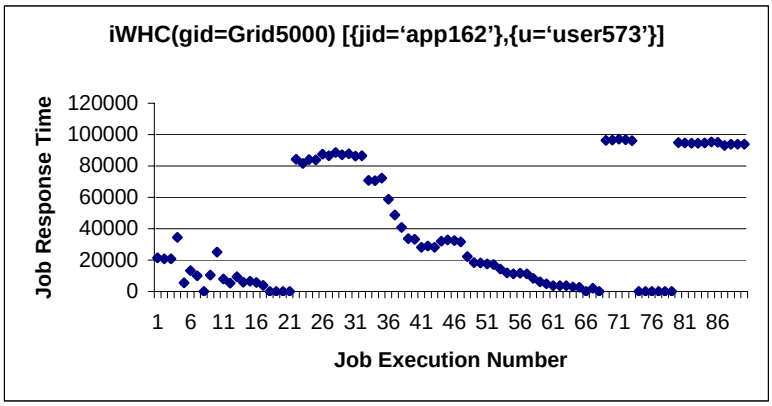


Figure 14 iWHC using jobId app162 and user user573 in Grid5000

In order to understand an iWHC internal behaviour it is necessary to implement a technique that has two goals: the first one is to identify the nature of the phenomenon represented by the sequence of observations, and the second, and most important, is to is to predict the future values of the series of job response time. Both of these goals require that the pattern of the observed data is identified and described.

Unfortunately, there are no proven automatic techniques to identify trends in data series. However as long as the trend is monotonic (consistently increasing or decreasing) or seasonal (a concept defined as the correlational dependency of order *k* between each *i*'th element of the series and the (*i*-*k*)'th element) this analysis can still be performed.

Many monotonic series data can be adequately approximated by a linear function by using mathematical or polynomial functions. However, the iWHC contains considerable variances that even smoothing [85] the series leaves the resulting data without a consistency trend. This idea is shown in Figure 15 where two solutions, the statistical average and linear regression, are applied to the previous example.

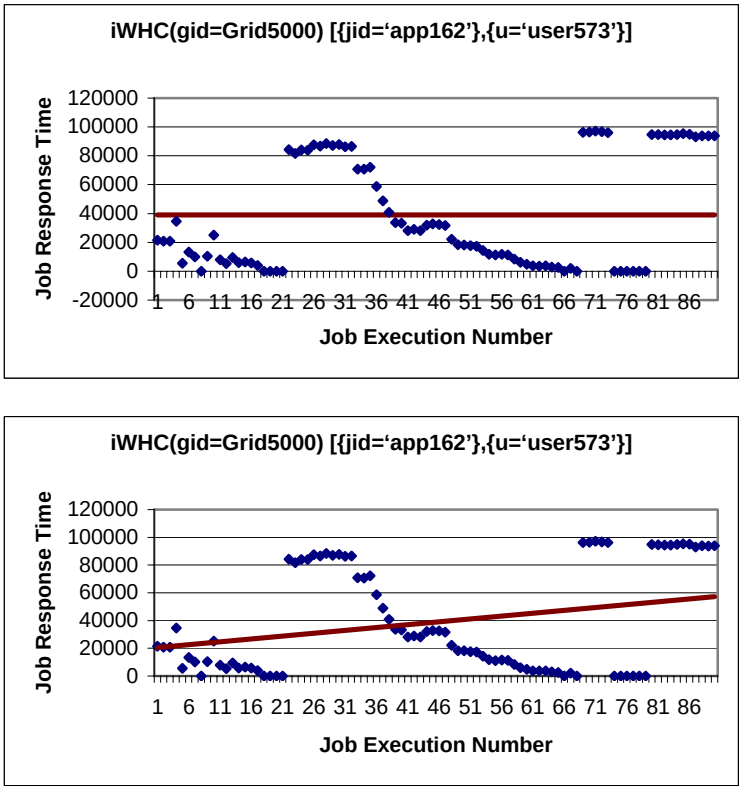


Figure 15 Response time average (top) and the linear regression (bottom) for an iWHC

Following the same experimental line, another test using a polynomial method was carried out. The aim was to determine a useful curve that best fit the response time in an iWHC. In this case, the analysis determines the coefficients of the polynomial that leaded to the match the data line. The more data that is available, the higher the degree of the polynomials that can be set. The determination of the coefficients was produced by executing examples with different polynomial degrees (Figure 16).

The polynomial analysis represented a step forward for the prediction method. Graphically the line fits the values in a more accurate way than the examples presented before. But, the point is that finding a function that fits a historical response time series does not always mean that it will accurately predict the future. In other words, a polynomial is an option that via adjusting the number of degrees to its maximum allowed by the discrete data can be used to fit a discrete

set of data. The problem is that in the occasional sudden response time changes (often found in Grid computing data), the polynomial analysis tends to produce inaccurate prediction errors.

This point can be appreciated in Figure 16 where a 4th and 5th degree polynomials are shown. The 5th degree polynomials fits better the past behaviour of the iWHC but neither of them is able to adapt to the latest sudden response time change.

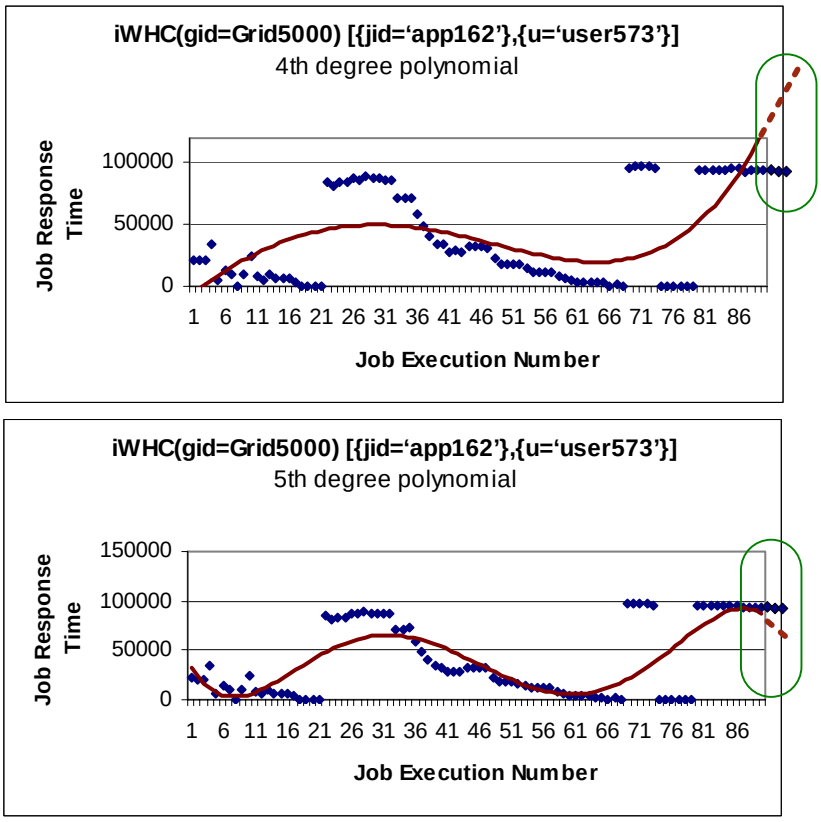


Figure 16 Catalogue polynomial fitting with 4 (top) and 5 (bottom) degrees for an iWHC.

The data analysis that tries to fit the curve has several disadvantages when future changes that directly affect the data values in the series cannot be controlled. This is often the case in current production Grid environments workload data. The logistic model provides little basis for extending trends into the future. The fundamental difficulty is that a single logistic curve assumes a fixed limit to the variable being modelled, and those limits can be altered through changes in technology or social factors. Thus, while a particular curve may fit historical observations, it does not match the future behaviour.

Therefore, rather than using a method that fits a curve, such as the statistical average, the response time linear regression, or the polynomial analysis, another method should be used. This method is suggested in Figure 17 where it can be seen that the series does not follow a

specific trend and it does not have seasonality but in contrast, there are identifiable data clusters where the trend is usually stable.

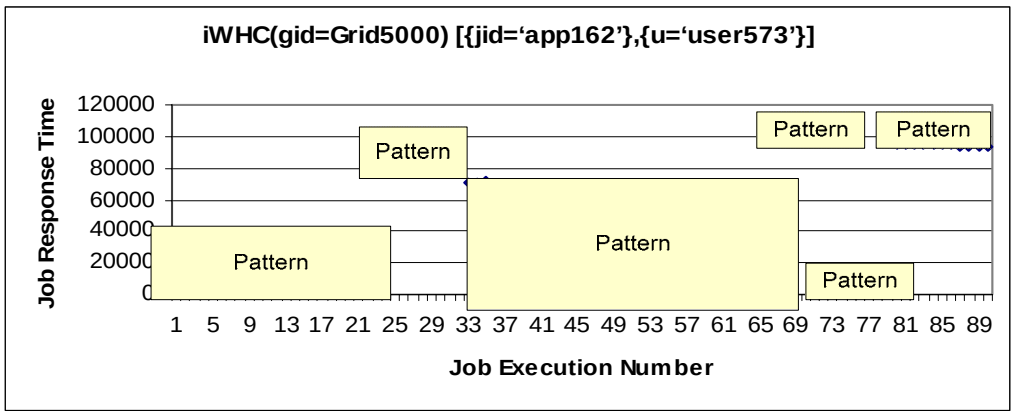


Figure 17 Response time patterns inside an iWHC.

The suggestion is that the iWHC data can be understood by finding similar patterns and selecting clusters of continuous data values. As a result, instead of using a function that fits whole iWHC, the idea is to put in place a method that denotes that something has changed in the historical executions of a job. The idea is not understanding why the response time has changed; but rather understanding when that change has occurred and use this information for future response time predictions.

3.4.3. Data Clustering Solutions Overview

Data clustering [86][87][88][89][90][91] is the unsupervised classification of patterns into groups. Data clustering has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps in exploratory data analysis. Still, clustering is a difficult combinatorial task and differences in assumptions and contexts make the transfer of useful generic concepts and methodologies slow.

Amongst the several data clustering method's taxonomies suggested, such as [92] or [93], the simplified taxonomy suggested in [94] (Figure 18) is used to organise the discussion regarding the different clustering methods.

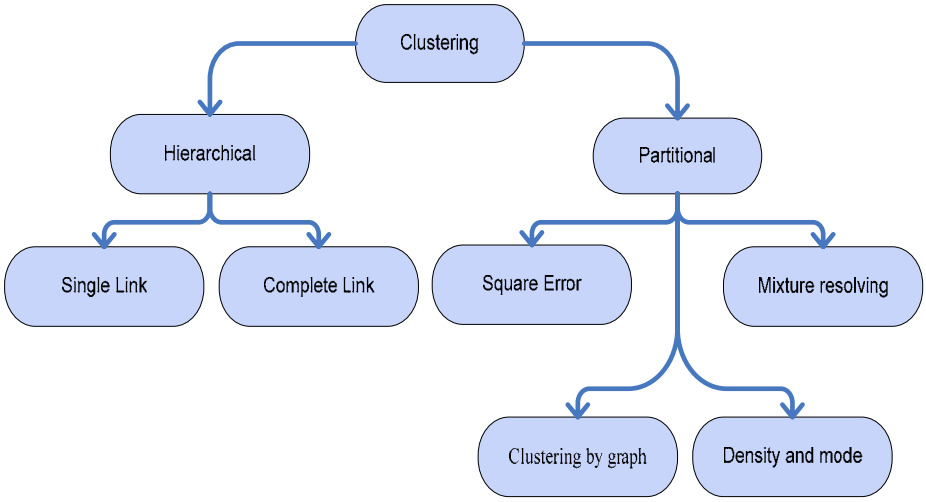


Figure 18 Taxonomy of clustering methods.

This taxonomy divides clustering methods into two basic types: hierarchical and partitional. Each of the two basic types has a wealth of subtypes and different algorithms for finding the clusters.

- Hierarchical clustering proceeds successively by merging the most similar clusters into larger ones. The clustering methods differ in the logic employed to decide which two small clusters are merged, or which large cluster is split. The result of a hierarchical clustering is a tree of clusters called a dendrogram, which displays the relationship between the clusters.
    - A clustering of data items into disjoint groups is obtained by cutting the dendrogram at a desired level [95][96].
    - In single linkage clustering (or nearest neighbour) the distance between two clusters is computed as the distance between the two closest elements in the two clusters.
    - In complete linkage clustering (or the farthest neighbour method) the distance between clusters is computed as the maximal object-to-object distance.
- Partitional clustering attempts to directly decompose the data set into a set of disjoint clusters. Typically the decompose criteria involves minimising some measure of dissimilarity in the samples within each cluster, while maximising the dissimilarity of different clusters.
    - Square error: The objective of the method is to obtain a partition that, for a fixed number of clusters, minimises the square-error, where square-error is the sum of the Euclidean distances between each pattern and its cluster centre. The most representative example of this type of method is K-means [97].
    - Clustering by graph: These algorithms treat the patterns as points in a pattern space, so that distances are available between all pattern pairs. A complete graph is formed

by connecting each pattern with all its neighbours. The edge weights are distances between pairs of patterns. The algorithm removes the inconsistent edges to form connected components and call them clusters. Inconsistent edges are those whose weight is significantly larger than the average of nearby edge weights. Zahn's method [98] is the most representative example of this type.

- Mixture resolving methods assume that the data items in a cluster are drawn from one of several distributions (usually Gaussian) and attempt to estimate the parameters of all these distributions. The Mixture resolving methods make rather strong assumptions regarding the distribution of the data [99][100].
- Clustering by density estimation and mode seeking: Clusters are viewed as regions of the pattern space in which dense patterns are separated by regions of low pattern density. Clusters can be identified by searching for regions of high density, referred to as modes, in the pattern space. [101]

3.4.4. Data Clustering Solutions applied to iWHC response time data

The decision of selecting a particular clustering solution for the iWHC data analysis should be based on the following four requirements:

- The data in each cluster must always consist of continuous data values. Continuous records in an iWHC with similar response times are in fact successive historical records of a job where it performed in a clear stable state and no other factor affected the job's response time performance. The main objective of the clustering method should be observing the stable state of these jobs via the creation of clusters. These observations are used to understand the type of iWHC, the historical response time tendency, and the future response time.
- The number of clusters is not known in advance: The response time found in an iWHC varies from execution to execution. New response time data clusters may appear at any given point. Consequently, the number of clusters is not known in advance, so the method should be able to automatically discover them.
- The clusters should meet a guaranteed threshold. The threshold may be adjusted to alter the number of clusters that the prediction service should analyse. The smaller the threshold, the bigger the number of clusters. The larger the number of clusters, the more costly the prediction service is. The smaller the threshold, the more accurate the prediction of the

response time may be. Therefore, a threshold is a significant variable that can be used for tuning the service based on running cost versus prediction accuracy.

– The amount of data grows over the time. For that reason, the proposed solution must have a cost that has a linear relation to the number of records to be analysed.

By considering the above requirements alongside the clustering taxonomy, it can be concluded that pure hierarchical clustering methods have a disadvantage in that they do not take into account the sequence in time when the data was produced. Furthermore, hierarchical clustering methods do not guarantee a given threshold for each cluster. However, having said that, a hierarchical clustering method can be modified so that it only links together continuous data values and uses a linkage function that guarantees a given threshold. The dendrogram can be cut when all clusters meet the desire threshold.

Partitional clustering square error methods, such as K-means [97], have the advantage that they are simple and fast, which allow them to run on large datasets. The main problem when applying K-means to an iWHC is that it does not yield the same result with each run, and it assumes a fix number of k clusters a priori. Since this is rarely the case, techniques for finding an appropriate number of clusters had to be devised. The problem is partly solved for methods based on the squared error by adding a regularisation term to the cost function, or by using a cluster validity index to select the appropriate number of clusters a posteriori [102]. This is an important issue for partitional clustering in general and it has the main disadvantage of adding an significant computation cost to the solution. This is especially true in the case where several iWHC should be clustered and analysed in a single prediction request.

All the remaining partitional clustering methods have been developed for specific problems that do not fit in with the nature of the data found in iWHC. For instance, density-based algorithms are mostly developed for spatial data mining and make use of less dense regions to divide clusters.

However, Quality Threshold (QT) Clustering (Algorithm 1) offers an alternative method of partitioning data. QT that was initially described in [103] and it was used to cluster gene expression patterns from renal carcinomas in [104]. In the QT clustering algorithm, each row or element is compared in a pair-wise fashion with every other row or element and correlation coefficients are computed. Elements are clustered together so that all of the elements within a cluster must be more highly correlated with a single central element than the input quality threshold. The algorithm then computes the largest cluster that it can create using the input quality threshold. It subsequently removes those rows/columns from consideration and then computes the next largest cluster, and so on until all of the elements that can be clustered together within the threshold limits are clustered.

QT_Clustering(dataCatalogue C, threshold t)
If $(|C| \leq 1)$ then Output C
Else
For each $d \in C$ do
Set $flag = TRUE$; Set $A_i = \{i\}$
While $((flag = TRUE) and (A_i \neq C))$
Find $j \in (G - Ai)$ such that the diameter $(A_i \cup \{j\})$ is minimum
If $(diameter(A_i \cup \{j\}) > d)$ then
Set $flag = FALSE$;
Else
Set $A_i = A_i \cup \{j\}$
Identify set $C \in \{A_1, A_2, ...A_{|G|}\}$ with maximum cardinality
Output C
Call QT_Clustering(G-C, t)

Algorithm 1. Quality threshold (QT) clustering method.

The QT clustering method does not need the number of clusters specified a priori, and also considers a level of the quality threshold that guarantees that all data is within that range. Unfortunately QT requires more computing power than *k*-means and therefore it cannot be run on large iWHC. For n records, the costs of running QT is described as n^n for one interaction and this should be repeated at most $(n-1)$ times more, which is an order of n^{n+1}.

The conclusion is that the existing methods are not entirely meeting the full list of clustering criteria. Therefore, there is a need of adjusting an existing method to provide the expected iWHC response time data clustering solution.

3.5. Chapter Summary and Conclusions

The objective of this chapter was to illustrate the quality and shape of the data that can be found in production Grid workload traces. Those traces can be used as the non-intrusive input data for a response time prediction solution. As a result, the main conclusions from this data analysis are summarised as follows:

– In [76] it was demonstrated that there are key characteristics that are not always normalised in Grid environments. Some of these key characteristics are hidden or not shown. This issue has been highlighted as one of the reasons that current methods are not performing well in

production Grid environment. This situation generates misjudgements in the prediction and concludes that it is necessary to have a dynamic method which chooses the input fields. Consequently, a field's entropy analysis can be used to set a scenario for each Grid site that shows how important each available field is in terms of information

- The results of the entropy experiments demonstrate that if intuitively data fields, such as the JobStructure or the Status, are used in an ad hoc manner when creating a prediction model, the resulting solution will not perform well in production Grid environments. Furthermore, as the quality of information is different from one Grid site to another, the use of static or ad hoc prediction models can be suitable for one Grid site, but they may fail to produce accurate results in another site.
- The joint entropy and conditional entropy experiments show that there are fields that are strongly tied. Considering only the largest of them in terms of the entropy value would be enough to have the same information as considering both fields. Also, this experiment shows that the pairs of data fields that add information to the system are different from Grid site to Grid site. Therefore, each Grid site has to have a dynamic method which is used to link them together in order to produce a quality of information hierarchy.
- A response time multi-pattern phenomenon is commonly found in production Grid computing environments when a number of historical records are retrieved from the workload trace. This phenomenon follows the response time changes which cause is hidden or non-shown in the workload data logs. As a result a further data analysis to expose those changes is needed.
- Techniques to identify trend components and curves that may fit an iWHC data in particular, do not provide any guidance on how the future behaviour may be. The analysis suggested that instead of fitting a line, another type of analysis should be carried out. The idea is to identify clusters of stable data that helps to predict a job response time.
- The survey in clustering data methods suggests that an existing method should be adjusted to provide the expected iWHC response time data clustering solution.

Chapter 4 – GRTP Method Presentation

The main purpose of this chapter is to introduce the GRTP method before moving on to the next two chapters where a detailed explanation of the method is presented.

The foundations of the GRTP were first presented in [105]. In this paper, the proposal pictured a system that allows complex analysis of data providing data retrieval and summarisation functionalities with modelling of rules. In this chapter, a final picture of the method is presented starting with a broad description of the internal stages that the method is proposing, followed by the interaction between the GRTP service and the OGSA Execution Planning Service, finalising the chapter with the usage scenarios for the solution.

4.1. GRTP Service and OGSA Execution Planning Service Interaction

It is correct to think that the Grid computing story is not quite over yet. Grid technology is evolving rapidly and standards, frameworks, implementations, and applications are changing constantly. OGSA is intended to facilitate the use and management of distributed, heterogeneous resources. The utility provided by such an infrastructure is done as a set of services divided into the following groups:

- Execution management services are concerned with the problems of instantiating units of work and managing them until completion; they find execution candidate locations and select an appropriate location for execution.
- Data services can be used to move data as required; manage replicated copies; run queries and updates; and federate data resources. They also provide the capabilities necessary to manage the metadata that describes this data, in particular, the provenance of the data itself.
- Resource management services perform several forms of management on resources in a Grid. For instance management of the resources themselves (rebooting a host).
- Security services describe objectives, models, functional capabilities, and properties of the security services, and its interactions with the rest of OGSA.
- Self-management services were conceived as a way to help reduce the cost and complexity of owning and operating an IT infrastructure. In a self-managing environment, system components are self-configuring, self-healing and self-optimising.

- Information services provide the dynamic data used for status monitoring; the relatively static data used for resource discovery; and any data that is logged in a Grid environment.

The execution management services group of capabilities specifies how to find a resource where a job can execute based on it satisfying resource restrictions, such as memory, cpu and binary type, available libraries, and available licenses. In addition, it considers any running policy restriction in place that may further limit the candidate set of execution locations.

Once the resources able to execute the job have been identified, the next step is to establish which of them may actually execute the job. Answering this question involve algorithms that attempt to meet a SLA. The Execution Planning Service (EPS) use functions that build execution schedules that in turn try to optimise an objective, such as execution time, cost, reliability, etc. The final picture is completed when the schedule generated by the EPS is ultimately executed by the job manager.

In this research, it is foreseen that the EPS can use the job response time predictor in order to optimise the execution schedule.

The interactions between these services are represented in Figure 19, where it is shown that:

- After a Client has requested the execution of a job (step 0), the job manager creates the appropriate job description and calls the EPS to obtain a schedule of execution (step 1).
- The EPS calls a Candidate Set Generator (step 2) which determines where the job can be executed based on binary availability and policy settings. The input information is provided by the Grid information services (step 3).
- Having a list of possible resources where the job can be executed (step 4), the EPS can check with the GRTP Service and find out the predicted job running time in each of the resources (step 5). If the SLA is met the execution schedule can be passed to the job manager (step 6), otherwise a new list of possible resources (step 4) should be requested. It is interesting to note that steps 4 and 5 may have several interactions in order to tune the list of resources where a SLA can be met. The collection of the final list of resources may imply the selection of an execution schedule that is cheaper to run, but that nevertheless meet the agreed SLA.

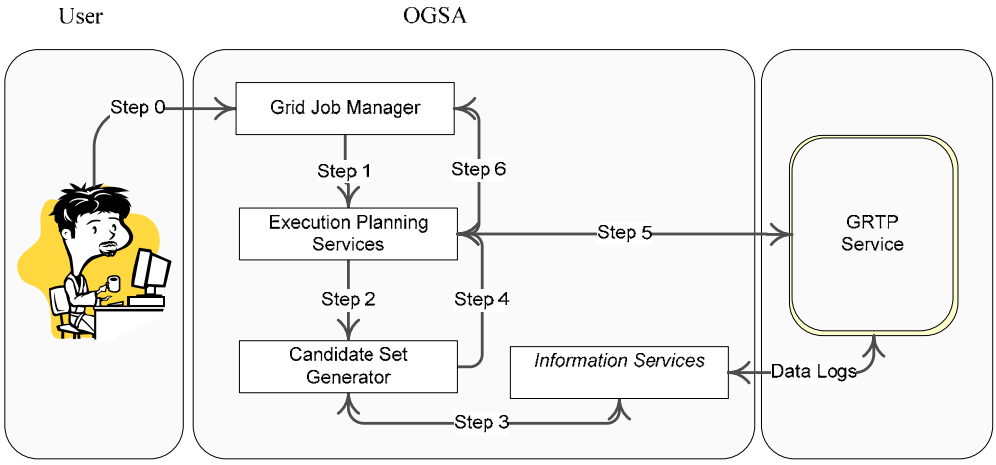


Figure 19 Interaction between the Response Time Prediction Service and the OGSA

The previously described scenario, shows that on the one hand the Execution Planning Service (EPS) is able to request a set of resources where a job can be submitted from the Candidate Set Generator (CSG), and on the other hand, the EPS should also meet a given Service Level Agreement (SLA).

To meet both requirements, a number of decisions about resources, services, and submission time has to be taken. In this scenario it is considered that the local scheduler accepts the EPS resource selection where the job has to run. Thus, the local scheduler acts on behalf of the EPS by reserving and submitting jobs with a given policy.

The EPS level is referred to as meta-scheduler since it interacts with the local schedulers to decide what resources should be used before dispatching a job. Meta-scheduling is an area which has been researched by many authors. In [106] Sabin presented the scheduling of parallel jobs in a heterogeneous multi-site environment. This research carries out a global scheduling within a set of different sites. Also, more centralised approaches have been proposed in the literature. For instance, the impact of geographical distribution of Grid resources on the machine utilisation and the average response time is analysed in [107], and in [108] Pinchak describes a meta-queue system which manage jobs with explicit workflow dependencies. In this case, a centralised scheduling system is presented.

The EPS is may interact with the GRTP service to obtain a predicted response time for a given job. The GRTP service may be queried several times with different combination of resources until the EPS understand that the predicted response time in the selected resources meet the requested SLA.

For instance, the first query may use the job identification, the user that is about to submit the job in one specific Grid queue. The same query may be repeated using other Grid queues. At

the end, the EPS can have a list of predicted times for different queues in the system. Both results can be compared to select the final resource list for a given job and SLA.

4.2. General Overview of the proposed GRTP Method

Figure 20 displays a graphical representation of GRTP life-cycle, where the following steps are labelled:

- **T1, Data Normalization**: Every time that a job completes its execution, the Grid site is able to produce a new workload trace record and make it available to the GRTP service. At T1, the new records are processed and transformed from a distributed semantically heterogeneous data set, into a collection of structured tables (Labelled *information* in the diagram) that the GRTP method can make use of. A new record is accepted if it fulfils the requirements of the workload trace definition (Chapter 3 Section 3.2.): It must belong to a Grid site and it has to have the job identification, the job execution number, and the job response time. Having met these requirements, any other field and data is accepted at T1.
- **T2, Meta Modelling**: In this step the analysis of the workload trace data fields introduced in Chapter 3 Section 3.3. is implemented. This process is constantly producing and adapting a workload meta-model that ensures that only relevant fields and data are put together for the GRTP method.
- **T3, Data Cataloguing**: The GRTP service is triggered by a response time prediction request as described in the service interaction with OGSA in Chapter 2 Section 4.1. This step uses a combination of the workload meta-model produced by T2 and the information produced by T1. The result is a subset of fields and historical data that is only related to the job that is going to be predicted. In other words, T3 uses the WHC and the prediction request to instantiate iWHCs (Chapter 3 Section 3.4.1.) that are soon after used to filter the historical information.
- **T4, Catalogue Similarity**: In this step, the iWHCs are ordered by their level of accuracy. A iWHC level of accuracy is determined by a previous prediction request. A request predicts the response time for a job using all possible iWHCs. This response time is stored together with the iWHC in a historical database. Once the job is has finish running, the actual response time is added to the iWHC historical record. Therefore, each entry in the historical database of iWHC ends up having the predicted response time and the actual response time. The percentage of the difference between these numbers is the iWHC level of accuracy for that specific instance.

- **T5, Response Time Prediction**: At this point, the most accurate iWHC (given by T4) is selected and instantiated. The next step is to understand what type of job is represented by the data. This is carried out by implementing the job response time data profile analysis introduced in Chapter 3 Section 3.4. In this step, depending on the job type, a data mining method is applied to produce the forecasted prediction response time that is used as the service output.
- **T6, Post Mortem Activities**: This final step validates the predicted response time against the real response time offered by the execution panning service. The actual response time is stored in the iWHC historical database to calculate its accuracy level to infer future results.

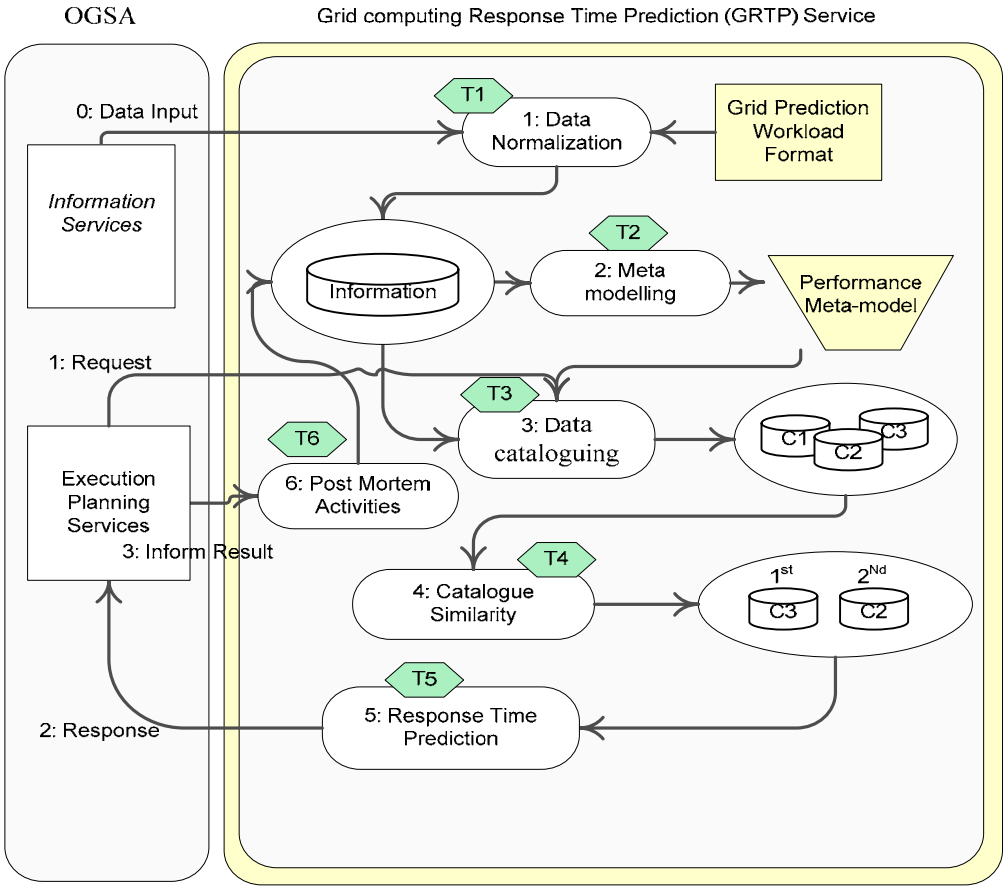


Figure 20 GRTP method general graphical overview

4.3. Usage Scenarios

As previously described, a request to the GRTP service triggers a number of internal steps and one of the steps is the selection of the most accurate iWHC. The iWHC produces historical information related to the job about to be predicted.

By definition, the GRTP service will be fully functional for a job response time prediction query only if:

- historical data can be selected for the requested job's iWHC and,
- the iWHC has been used before and its accuracy information is available.

The aim of this section is to outline the usage scenarios for the GRTP method. The requirements and implications considering each scenario are analysed from the consumer's point of view and also from the method's point of view.

The consumer point of view is focused on the implications that a client may expect when using the prediction service. The predictor point of view describes whether the application can produce a relevant result, or if the predictor should run in learning mode. It also describes what activities the prediction service should perform to reach the next scenario. The interaction between scenarios is depicted in Figure 21, where the amount and quantity of historical data play a key role.

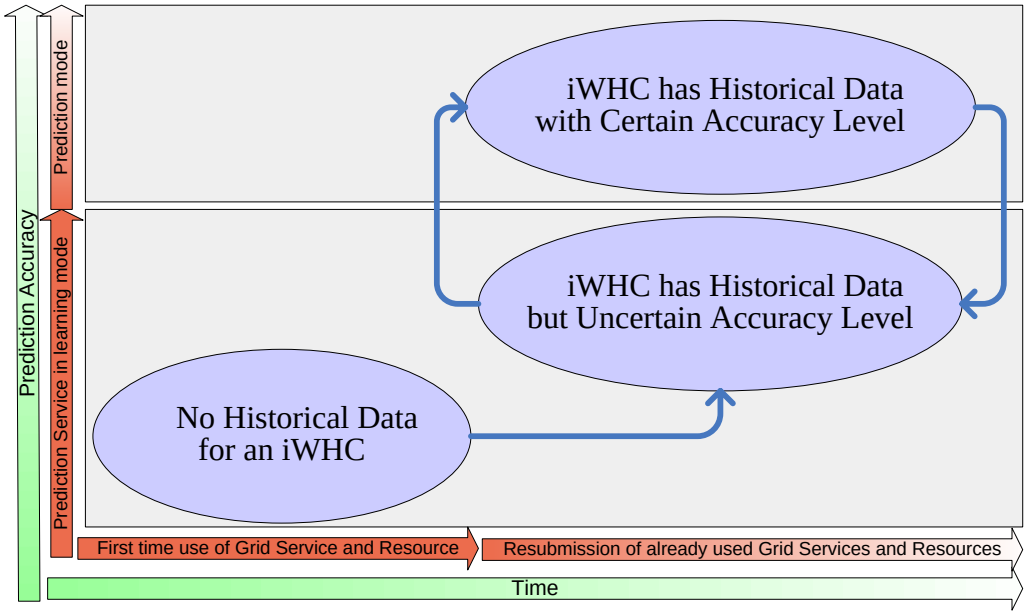


Figure 21 GRTP method usage scenarios for an iWHC

4.3.1. No Historical Data for an iWHC

The first case scenario takes place when the historical data for a given job's iWHC initially becomes available to the prediction service. This situation occurs either when a job is submitted in the Grid environment for the first time, or when a response time prediction solution has just been deployed and no historical information has been collected yet. The learning process starts at this point via collecting the workload information together with the accuracy level that describes the iWHC.

The implications of such scenario include:

- The GRTP service is available to process workload trace records. At this stage, no results are offered if a consumer queries the GRTP. The information retrieved from the different Grid sites is processed and stored at the same time that the meta-modelling is constantly built.
- GRTP starts building the knowledge database at this stage. The knowledge database is not only composed of analysed workload traces but it also contains how accurate a job response time has been predicted. As a consequence, even if the result is not provided to a consumer any requested prediction is carried out to populate the knowledge database. In future prediction requests, the historical information contained in the knowledge database will be used to move the GRTP service to the next two usage scenarios.

4.3.2. iWHC has Historical Data but Uncertain Accuracy Level

In this scenario, a job has been used several times and as a result the workload information has been loaded into the GRTP service. At the same time the prediction service has also populated the meta-model. However, the response time request affects an iWHC that still have a low accuracy level. The accuracy level is calculated comparing the predicted response time using a given iWHC against the real response time offered by the execution panning service. A low accuracy level for an iWHC is established when the historical response time prediction error suggests that the iWHC may be inadequate to use.

The implications of such scenario include:

- The GRTP service is available to a consumer for querying but the fact that the predicted result is not accurate should be taken into account by the service consumer.

- The prediction service continues predicting all requests, but none of the previously used catalogues have provided a high level of accuracy. Therefore, the result is provided to the consumer together with a warning about the accuracy of the solution.

Logically, this particular situation can be reached from a stage where no historical data is available for a meta-data model. However, this scenario can also be reached from a stage with assured level of accuracy given by changes in the Grid environment resources or jobs characteristics.

4.3.3. iWHC has Historical Data with Certain Accuracy Level

The final scenario describes a GRTP service that has been established and that has built a knowledge database. Also, in this scenario the iWHC used in a response time prediction request has an historical accuracy level that suggests that it can be used in the prediction method.

The implications of such a scenario include:

- This is the ideal situation for a prediction service consumer, given that the performance information about the job submissions is available.
- The prediction service continues to predict each submission while the post-mortem activities help tuning the different variables. In case of discrepancies between a predicted response time and the real value, the iWHC for a given job will be downgraded to the lower scenario.

4.4. Summary and Conclusions

In the first section of this research a number of prediction solutions were analysed and the special characteristics that production Grid environment workload traces offer were introduced. This chapter is a pivot point between the first section and the upcoming sections of this research where the proposed response time prediction method is presented and evaluated.

This chapter introduces the GRTP method via the use of a graphical explanation of its life-cycle, the description of the interaction between the GRTP service and the OGSA Execution Planning Service, and the presentation of the three possible scenarios that the GRTP service may face. During the presentation of these points, references to the first and next sections of this research are quoted to connect both parts together and guide the reader to the next section.

Chapter 5 – Workload Trace Data Fields Analysis for the GRTP Method

To facilitate the reader's understanding, the GRTP Method is presented in two consecutive parts. The initial part, explained in this chapter and graphically presented in Figure 22, takes Grid workload traces as input parameters, continues using information theory functions to qualify the workload trace's data fields, and finally produces a set of filters (WHCs) for the historical data associated to the response time prediction request.

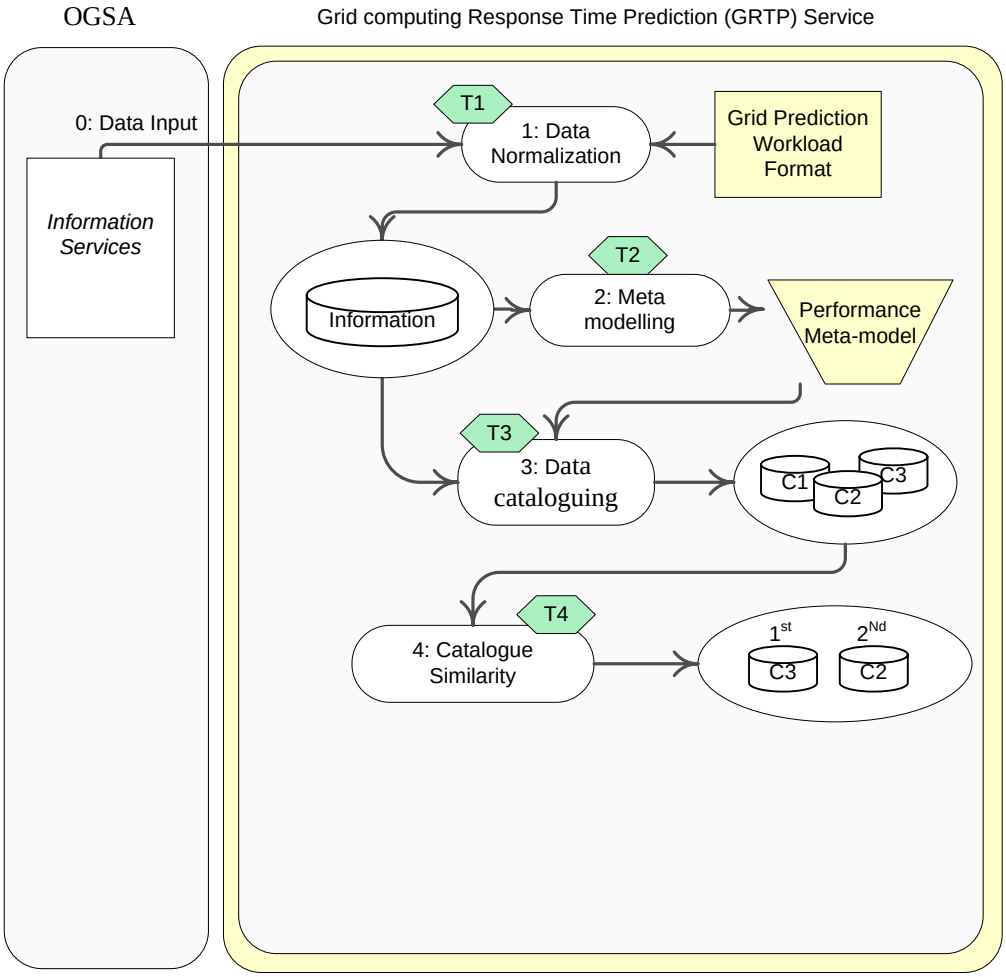


Figure 22 Workload Trace Data Fields Analysis Steps in the GRTP method

5.1. Grid Prediction Workload Format

This research follows the idea of presenting a Grid prediction workload format that accepts any information regardless its data type and composition. This suggestion was evaluated in [76] where all possible data fields coming from production Grid workload traces were analysed. The results of the experiment showed the importance of having the possibility of using all possible information to predict a job response time instead of just using what it is defined in a fixed and general workload format.

Furthermore, the data analysis presented in Chapter 3 Section 3.3. demonstrated that each data field can be categorised by its information in a particular Grid site, instead of trying to conclude the field's quality of information based on the field's name.

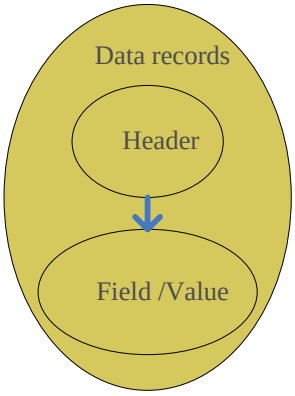


Figure 23 Grid workload prediction model structure

Moving the emphasis of each field from its name to its information quality guarantees that any Grid site can be explored to the maximum extent. Following this line of argument, the proposed Grid prediction workload format relaxes previous definitions by taking the form pictured in Figure 23, where each data record contains a header and a number of detail records composed of the field name and its instance value.

5.1.1. Proposed Grid Prediction Workload Format

Given a workload trace wT defined as $[gid, jid, jen, jrt, \{(f_0, v_0)..(f_k, v_k)\}]_{1..q}$, a Grid Prediction Workload Format (GPWF) is presented as a database entity relationship (Figure 24, left hand side) that can be described as a master-detail diagram with a:

- Header entity that takes the format < gid as Grid identification, jid as identification of the job, jen as job execution number, jrt as job response time>. A header example is pictured in Table 16.

Grid identification	Job Identification	Execution number	Response Time
Grid500	app968	4326	2577
Grid500	app968	4327	3668
Grid500	app97	4328	3607
Das	app902	4329	539
Das	app902	4330	6541

Table 16 Grid Prediction Workload Format header example.

- A detail entity that takes the format < Foreign Key to Header, $\{f_i : i = 1,..,k\}$ as Field name, $\{v_i : i = 1,..,k\}$ as Field value> where the field name is a name of any possible field provided by the Grid site (such as LastRunSiteID, Nproc, NumberAllocatedProc, QueueID) and the field value is the corresponding value for this field. A detail example is pictured in Table 17

Foreign Key to Header	Field Name	Field Value
Grid500-app968-4326	UserID	user134
Grid500-app968-4326	UsedNetwork	-1.0
Grid500-app968-4326	Status	1
Grid500-app968-4326	QueueID	queue0
Grid500-app968-4326	PartitionID	-1
Grid500-app968-4326	OrigSiteID	G1/site6
Grid500-app968-4326	NumberAllocatedProc	20
Grid500-app968-4326	Nproc	20
Grid500-app968-4326	LastRunSiteID	G1/site6/c1
Grid500-app968-4326	JobStructureParams	-1
Grid500-app968-4326	JobStructure	UNITARY
Grid500-app968-4326	GroupID	group6

Table 17 Grid Prediction Workload Format detail example.

In Figure 24 the GPWF and SWF entity relationship diagrams are presented for comparison. In this diagram the job response time (jrt) in the header entity is implemented using a non-mandatory field. This is because the header and details are stored when the job is submitted and the real response time is still unknown. While the job is running, the jrt field is kept as null.

This value is updated with the final response time after the job has finished. Only records that have a response time different than null are considered for a prediction request.

Figure 24 exemplifies the difference between both approaches. Even if the GPWF can hold all the information contained in the SWF, it is also open to accept any other field that a Grid site may offer for the performance analysis without the need of modifying the definition of the format.

The entity relationship diagram uses the following notation:

- **A shading name** represents the name of the table.
- **A data field in bold** represents a mandatory data field. Otherwise the data field is not mandatory.
- **PK**: Is a primary key for a data table.
- **FK1**: Is a foreign key between two tables.

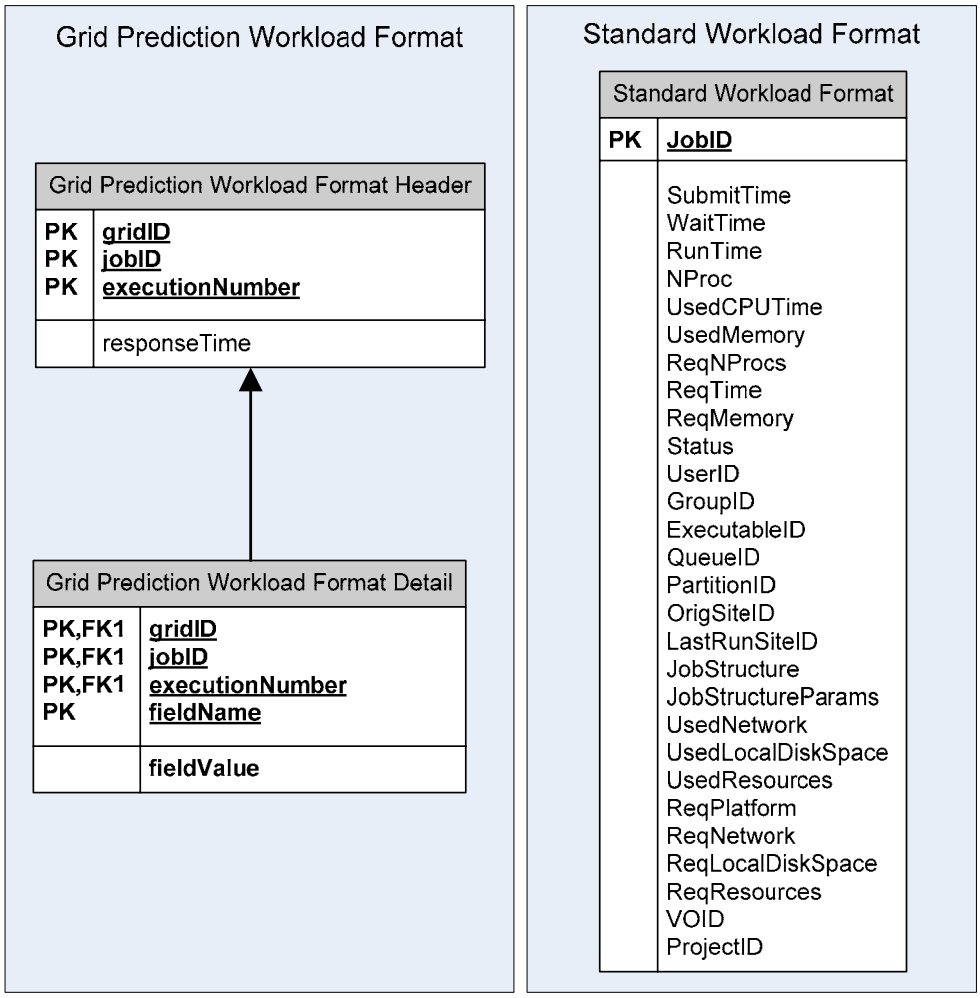


Figure 24 GPWF and SWF entity relationship diagrams

The advantages of the GPWF definition is that it relaxes other current workload formats by allowing the inclusion of any fields that may be beneficial in the process of understanding the past performance of a job to predict its future behaviour. Furthermore, instead of taking the traditional semantic view which restricts a data field's value to the field's name for a particular Grid instance, the GPWF disregards the field name and allows the GRTR method to qualify data fields based on the quality of the information that they contain.

5.2. Workload Meta-Model and the Creation of WHCs

In Chapter 3 Section 3.3., a workload trace data analysis demonstrated how the data fields in a workload trace can be quantified and related to another fields based on the field's information using formal information theory fundaments. In [76] it was established that the job identification must be the main binding field to retrieve historical data. In this research, both ideas are put together in a method that creates a workload meta-model.

As a result, starting from the definition of the GPWF the solution proposes the job identification as a binding field and, in contrast with any previous research in the area, the extended level would be created by any other field that holds relevant information. This step would guarantee the neutralisation of the heterogeneous data sensitivity and also the reduction of the complexity of the solution by reducing the number of data fields.

5.2.1. Proposed Workload Meta-Model Method

Definition: A workload meta-model is proposed as $wmM(gid)[\{f_1,..,f_h\}\{f_{h+1},..,f_n\}]$ where gid is the Grid identification, $\{f_i : i = 1,..,h\}$ are a list of fields composing the *binding set of fields*, and $\{f_i : i = h+1,..,n\}$ is a list of fields composing the *extended set of fields*.

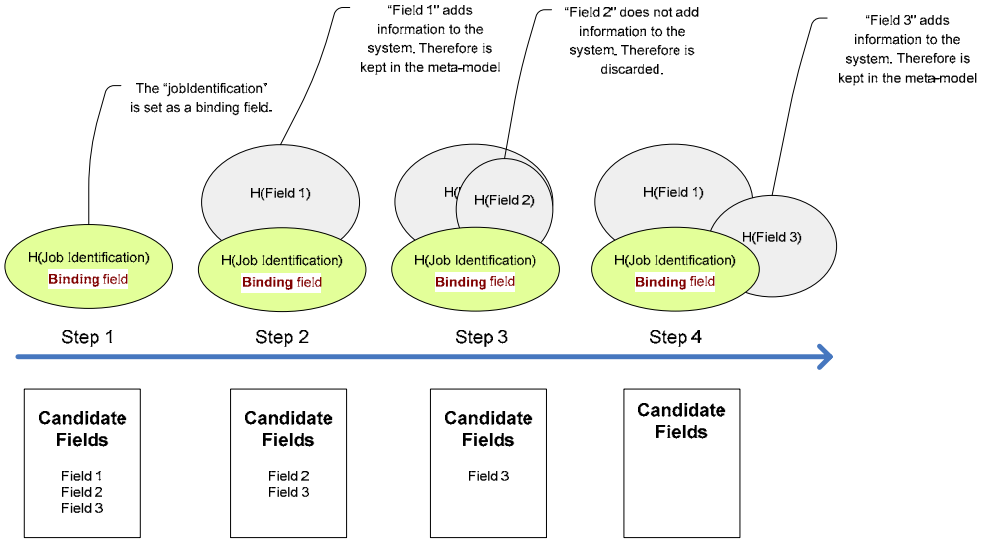


Figure 25 Workload meta-model construction algorithm graphical representation

A graphical representation of the process that created the workload meta-model is shown in Figure 25. During the first step the job identification composes the binding set of fields. This is followed by a number of steps that build the final workload meta-model. If the joint entropy of all selected fields, together with the candidate field increases in each of the consecutive steps, then the candidate field is added into the extended set of fields of the meta-model. In other cases, the field is discarded because it does not add any valid information to the system.

The workload meta-model algorithm works as follows: the main function accepts four inputs: the Grid identification, the binding set of fields that in this research is the job identification [76], the extended set of fields that should be empty when the function is first called, a candidate list of fields composed of the rest of the fields that can be obtained from the Grid workload prediction model, and the threshold that is a value of information that a new field should add to the system to be considered for the extended list of fields. The function starts calculating the joint entropy of the binding set together with each field in the candidate list. For each field, if the joint entropy does not pass the threshold, then the candidate field is discarded; otherwise it is kept in a potential list of fields to be included. A new field from the candidate set is added to the extended set of fields based on its ability to increase the entropy of the extended set. This field is subsequently deleted from the list of candidate fields. The process continues recursively calling the function with the new binding and extended sets. The process stops when the set of candidate fields is empty. The pseudo code for the algorithm is shown in Algorithm 2:

workloadMeta-Model (Grid Id gid , Binding Fields bF , Extended Fields eF , Candidate Fields cF , Threshold t)
If $(|cF| \leq 0)$ Then
    Return ($wmM(gid)[\{bF\}\{eF\}]$)
Else
    For each $f \in cF$ do
        Calculate the joint entropy, $H(bF_0..bF_n, eF...eF_m, f) = H_f$
            If $H_f < t$ then
                Set $cF = cF - f$
    Select the $f \in cF$ which joint entropy H_f is maximum
    Set $eF = eF \cup f$
    Call workloadMeta-Model(gid , bF , eF , cF ,t)

Algorithm 2. Building the workload meta-model algorithm.

The algorithm was applied to the Grid5000 testbed with a threshold of 1 percent and the resulting step sequence is printed in Table 18.

- The **step 0** starts with JobID as the binding field. The next action is to calculate the joint entropy of the JobID together with each field in the candidate list. The joint entropy evaluation shows that the UserID is the field that adds more information to the system. The entropy of the JobID alone was 3.6255 and together with the UserID the entropy of both fields is 4.8014 (an increase of 32.4358 percent). The UserID field is selected and added to the extended list of fields. Also, in this step nine other fields are discarded as they are not going to add any information to the system (Those fields add 0.0000 percent of information to the field JobID).
- Step 0 has considerable reduced the complexity of the system by discarding several fields that are not adding any information. As a conclusion, at **step 1** only five fields should be analysed. Step 0 calculates the joint entropy of the binding set (JobID) and the extended set (UserID) together with each pending field in the candidate list. Step 1 shows that in the workload log of Grid5000 the QueueId, GroupID and OrigSiteID are fields that are dependent from the JobID and the UserID. Even though it was shown that those fields added some information in step 0, they are discarded in step 1 because they are not longer adding any information. Step 1 selects the field LastRunSiteID to be included into the binding set.
- **Step 2** starts with the binding set (JobID) and the extended set composed of UserID and LastRunSiteID. Following the same methodology, Step 2 selects Status as the field to be

included into the extended set of fields and discards any of the other fields in the candidate set.

- Given that the candidate set of fields is empty, the algorithm finishes the creation of the workload meta-model. After four interactions, the procedure creates a meta-model that is compose of { *BF* {JobID} *EF* {UserID, LastRunSiteID, Status }}.

Data fields for entropy.	Entropy(Fields)	Entropy percentage added by new field
JobID	3.6255	
Step 0 Meta-Model = { *bF* {JobID} *eF* {}} *cF* = { UserID, NproC, LastRunSiteID, Status, GroupID, OrigSiteID, QueueId, JobStructure, JobStructureParams, UsedNetwork, UsedLocalDiskSpace, UsedResources, ReqPlatform, ReqNetwork, ReqLocalDiskSpace, ReqResources }		
JobID, UserID	4.8014	32.4358
JobID, NproC	4.4698	23.2878
JobID, LastRunSiteID	4.3687	20.4989
JobID, Status	4.3049	18.7398
JobID, GroupID	3.9206	8.1387
JobID, OrigSiteID	3.9206	8.1387
JobID, QueueId	3.7717	4.0320
JobID, JobStructure	3.6255	0.0000
JobID, JobStructureParams	3.6255	0.0000
JobID, UsedNetwork	3.6255	0.0000
JobID, UsedLocalDiskSpace	3.6255	0.0000
JobID, UsedResources	3.6255	0.0000
JobID, ReqPlatform	3.6255	0.0000
JobID, ReqNetwork	3.6255	0.0000
JobID, ReqLocalDiskSpace	3.6255	0.0000
JobID, ReqResources	3.6255	0.0000
Step 1 Meta-Model = { *bF* {JobID} *eF* {UserID}} *cF* = { NproC, LastRunSiteID, Status, GroupID, QueueId, OrigSiteID }		
UserID, JobID, LastRunSiteID	7.2811	51.6448
UserID, JobID, Nproc	5.4718	13.9624
UserID, JobID, Status	5.3199	10.7978
UserID, JobID, QueueId	4.8404	0.8119
UserID, JobID, GroupID	4.8014	0.0000
UserID, JobID, OrigSiteID	4.8014	0.0000
Step 2 Meta-Model = {{JobID}{UserID, LastRunSiteID }} T= { NproC, Status}		
UserID, JobID, LastRunSiteID, Status	7.4645	2.5188
UserID, JobID, LastRunSiteID, Nproc	7.2811	0.0000
Step 3 Meta-Model = { *bF* {JobID} *eF* {UserID, LastRunSiteID, Status }} *cF* = {}		

Table 18 Grid5000 workload meta-model construction example

5.2.2. Proposed WHC Creation Method

To create the list of WHCs (defined in Chapter 3 Section 3.4.1. as $whC(gid)[\{jid\}\{f_0,..,f_h\}]$) based on a workload meta-model for a given Grid site, the binding field is selected for the $\{jid\}$ section together with any combination of the extended set of fields for the section $\{f_0,..,f_h\}$).

Following the example printed in Table 18, where the Meta-Model { *BF* {JobID} *EF* {UserID, LastRunSiteID, Status }} was created, the list of WHCs is listed in Table 19. The list of WHCs is the power set of the fields in the extended set together with the field in the binding set.

List of WHCs
(gid)[{JobID}]
(gid)[{JobID}{UserID}] ,
(gid)[{JobID}{ LastRunSiteID}],
(gid)[{JobID}{Status}]
(gid)[{JobID}{LastRunSiteID, Status}],
(gid)[{JobID}{LastRunSiteID, UserID}],
(gid)[{JobID}{Status,UserID}]
(gid)[{JobID}{LastRunSiteID, Status, UserID}]

Table 19 Grid5000 example list of WHCs

This example shows how a complex system comprising seventeen fields is reduced down to a model of four fields. The result guarantees the neutralisation of the heterogeneous data sensitivity via the mandatory use of the binding set of fields, and reduces the complexity of the solution to the minimum by means of excluding data fields that carry low quality information.

The picture is completed when the WHCs are instantiated to filter the historical data. A response time prediction request triggers the instantiation of the list of WHCs by assigning values to the data fields. For instance, a prediction request for Grid5000, app968, user134 and status 1 end up producing the iWHC list printed in Table 20

List of iWHCs
(gid=Grid5000)[{JobID= app968}] (gid=Grid5000)[{JobID= app968}{UserID =user134}] , (gid=Grid5000)[{JobID=app968}{Status =1}] (gid=Grid5000)[{JobID= app968}{Status =1, UserID =user134}]

Table 20 Grid5000 example list of iWHCs for a prediction request

5.2.3. Proposed Method Running Cost Comparison

The solutions that use a powerset of data fields to create data catalogues [62][63] adds a significant extra running cost to the overall method. The comparison can be expressed as:

- The use of a power set of data fields uses a $\sum_{i=1}^{n}\binom{n}{i}$ number of catalogues for n fields.
- The proposed solution uses $\sum_{i=1}^{k}\binom{k}{i}$ catalogues when $k < n$ being k the number of fields in the extended set as catalogues.

The comparison is better illustrated in practical terms. The running cost is reduced by the exclusion of fields that do not provide information. This point is illustrated in Table 21 where the resulting lists of WHCs for each production testbeds are listed against Table 22 where the list of catalogues using any combination of fields is shown.

To understand the benefits of the workload meta-model algorithm, in Table 21 the output for Grid5000 is firstly presented. In this example, 8 different combinations of fields that can be used to filter the historical workload traces are produced. Base in the data entropy, those fields contain all the information that the historical data can provide. In comparison, Table 22 shows the outcome of selecting all data fields. In this case, the number of filters is 127. For a system that needs to analyse all possibilities for each job prediction, the significant reduction of data filters produced by the meta-model algorithm produce an important positive impact in the running cost of the overall method.

	Proposed workload meta-model method	
	Grid5000	DAS
List of WHCs	[{JobID}] [{JobID}{LastRunSiteID}] [{JobID}{Status}] [{JobID}{UserID}] [{JobID}{LastRunSiteID, Status}] [{JobID}{LastRunSiteID, UserID}] [{JobID}{Status, UserID}] [{JobID}{LastRunSiteID, Status, UserID}]	[{JobID}] [{JobID}{LastRunSiteID}] [{JobID}{Nproc}] [{JobID}{UserID}] [{JobID}{LastRunSiteID, Nproc}] [{JobID}{LastRunSiteID, UserID}] [{JobID}{Nproc, UserID}] [{JobID}{LastRunSiteID, Nproc, UserID}]
	AuverGrid	
List of WHCs	[{JobID}] [{JobID}{QueueId}] [{JobID}{Status}] [{JobID}{QueueId, Status}] [{JobID}{UserID}] [{JobID}{QueueId, UserID}] [{JobID}{Status, UserID}] [{JobID}{QueueId, Status, UserID}]	

Table 21 List of WHCs created from the workload meta-model method

	Non-restricted Method (using any combination of fields)
	Grid5000
List of catalogues	[{JobID}] [{JobID,JobStructureParams}] [{JobStructureParams}] [{JobID, LastRunSiteID}] [{JobID, JobStructureParams, LastRunSiteID}] [{JobStructureParams, LastRunSiteID}] [{LastRunSiteID}] [{JobID, Partitioned}] [{JobID, JobStructureParams, Partitioned}] … [{JobStructureParams, LastRunSiteID, Partitioned, QueueId, Status, UserID}] [{LastRunSiteID, Partitioned, QueueId,Status, UserID}] [{Partitioned, QueueId, Status, UserID}] [{QueueId, Status, UserID}] [{Status, UserID}] [{UserID}] (127 Catalogues for only six fields: JobID, LastRunSiteID, Status, UserID, JobStructureParams, PartitionID)

Table 22 List of catalogues using any combination of fields

5.3. Selecting the Most Accurate iWHC

WHC is composed of a set of data fields. An instantiated WHC (iWHC) is a WHC with values assigned to all data field. An iWHC can be applied to a workload to retrieve the list of records filtered by the values in the data fields. As shown before in Figure 15, more than one

WHC may be available for a Grid site and therefore more than one iWHC may be available for a given job prediction request. Consequently, prediction method should be able to identify the most appropriate iWHC.

In [109] different data mining methods were analyzed to select the most appropriated data filters. The suggestion was to use an approach that analyses how accurate an iWHC has performed for a given job in the past. The observations are based on the actual validation of iWHCs against the real response time value. This information is referred to as the *iWHC accuracy level.* As a result the selection of an iWHC uses values that indicate how precise a catalogue was in the past, and how accurate it can be expected to be in the future.

It is supposed that the history of an iWHC will have several accuracy levels. Different methods can be used to analyse them and define a unique iWHC accuracy level. The methods can take the nearest accuracy value into account, or use all the historical values. For example they can implement a standard average or median statistical function of the values. The aforementioned statistical solutions are simple to implement, but their disadvantages are: they cannot distinguish between current or dated values; they may be sensitive to outliers (average function) and there is no an option to adapt the iWHC selection function according to the Grid site.

In this situation, the proposed solution is a reinforcement learning method [110] that overcomes these disadvantages and finds an ideal implementation scenario. Reinforcement learning describes how an agent is able to learn successfully to control policies by experimenting in the environment. The goal of the agent is to define a reward function for each action that the agent may take based on each distinct state in the environment.

In this specific case, the states are given by the different iWHCs for a given job. The action that can be taken for each iWHC is to select it for the forthcoming prediction function. The reward or penalty of selecting an iWHC is given by its accuracy level. Therefore, the prediction method has the chance to create all possible states (represented by iWHCs) and also to check the reward or penalties that the selection of each of them may incur via validating the predicted job response time value against the actual job response time. These two steps set the ideal implementation scenario for the implementation of a reinforcement learning technique that selects the ideal iWHC. Furthermore, reinforcement learning does not need to be supervised, it is not sensitive to outliers, it deals with aged data, and it can be customised to ponder current or dated data.

5.3.1. Proposed iWHC Selection Method: Discounted Accumulative Reward Function.

Definition: To predict a job's response time the historical data is filtered using an $iwhC$. The *accuracy level* for an $iwhC$ is proposed as the percentage of error calculated by comparing the predicted response time and the real response time. Given that a job can have several $iwhC$ and that each $iwhC$ can have several accuracy levels (one accuracy level calculated each time that a job has been predicted in the past) the j^{th} accuracy level for the i^{th} instantiated WHC is noted as $iwhC_i al_j$.

In other words, the accuracy level effectively records how the data filter given by an iWHC has performed in a specific time.

Definition: Let us also define γ the delayed factor such that $0 \prec \gamma \leq 1$. The *reinforced learning discounted accumulative reward function* for an instantiated WHC is proposed as $V(iwhC_i al, \gamma) = \gamma^1 iwhC_i al_1 + \gamma^2 iwhC_i al_2 + .. + \gamma^m iwhC_i al_m ... = \sum_{j=1}^{m} \gamma^j iwhC_i r_j$.

The value γ is a constant that determines the relative value of delayed versus immediate rewards. In particular rewards received i time into the past are discounted exponentially by a factor of γ^i. Note that if γ gets close to 0 only the immediate reward is considered. As the γ gets close to 1, past rewards are given greater emphasis relative to the immediate reward (Figure 26). The use of a reinforced learning policy adds an important factor to the system. If the Grid environment has changed, the old rewards may perturb the cumulative value and γ should be set close to 0.

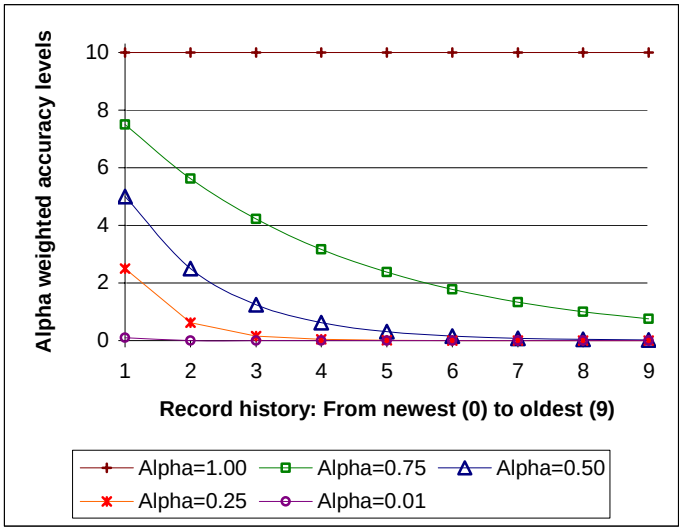


Figure 26 Weighted iWHC accuracy levels for different values of Gamma

Given that for each iWHC the reward or penalty is the prediction error percentage, the objective is to select the iWHC that has the minimum discounted cumulative reward.

Definition: Given a job that has n instantiated WHC, a method that chooses a iWHC that has the minimum value for the reinforce learning discounted accumulative reward function is proposed as: $iwhC_{s/1\le s\le n} / \neg\exists V(iwhC_s al,\gamma) < V(iwhC_i al,\gamma) \;\; \forall iwhC_i : i = 1,..,n$.

Both, the proposed reinforce learning method and discounted accumulative reward function can be adjusted to deal with aged iWHC accuracy levels. The GRTP uses γ to ponder the historical values, giving more priority to the newest data. From a practical point of view, the tuning of γ should follow the following principles:

- If a Grid site has the inclination to be dynamic where recourses and services are always changing, the practical suggestion is to give more priority to the latest iWHC accuracy levels assigning a value to γ tending to 0.
- Otherwise, in a less dynamic environment, a γ tending to 1 is able to make use of past experiences and deal better with outliers.

5.4. Chapter Summary and Conclusions

This chapter presented in detail the first steps of the proposed GRPT method that transforms a Grid workload trace into a set of classified iWHCs.

It starts by presenting the proposed Grid Prediction Workload Format that allows the inclusion of any data fields, instead of taking the traditional semantic view which restricts a data field's value to the actual field's name for a particular Grid instance.

Following on from this, a workload meta-model method which ensures that selected data fields are not losing information and also helps in the neutralisation of the heterogeneous data sensitivity via the implementation of data field restrictions is proposed. As an add-on, the meta-model reduces the overall complexity of the GRPT method.

Out of the meta-model, a list of related WHCs for a given job is produced. A WHC can be converted into an iWHC for any given response time prediction request. These iWHCs are classified via the use of a reinforcement learning method and a discounted accumulative reward function that uses the historical accuracy level of each WHC as a rewards or penalty. The proposed reinforce learning method can be adjusted to deal with aged data, giving more priority to the most recent accuracy levels.

Chapter 6 – Job Response Time Data Profile for the GRTP Method

This chapter explains the second part of the proposed GRTP Method. The GRTP Method uses the classification of the WHCs -previously explained- to produce a response time for a prediction request. Along with this, the post mortem activities where the validations of the predicted results are stored for future reference are also carried out. Figure 27 graphically represents the steps that this chapter develops. Figure 27 starts describing the method used to classify the iWHC for the type of job. This is carried out by implementing a job response time data profile analysis. Following the data profile analysis and depending on the iWHC classification, a data mining method is applied in order to generate a response time forecast (T5). The final step (T6) validates the predicted response time against the real response time offered by the execution planning service and to produce an accuracy level of each iWHC that is saved to infer future results.

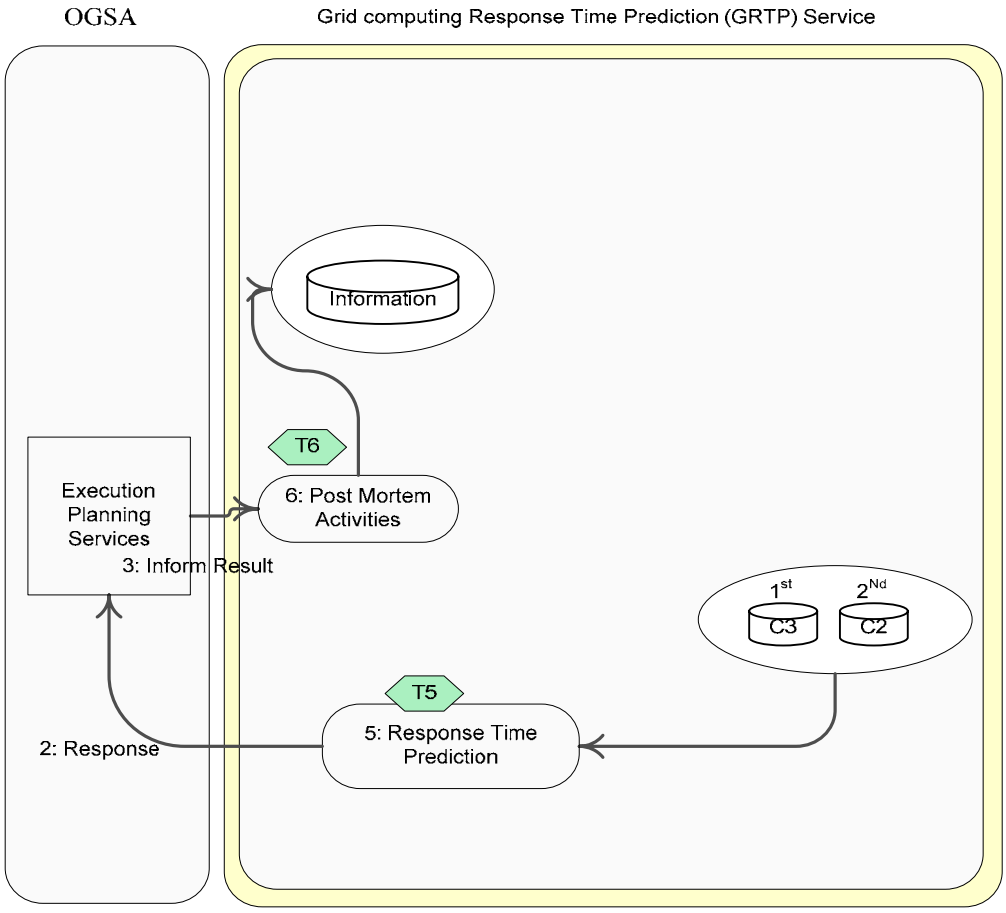


Figure 27 Workload Trace Data Fields Analysis Steps in the GRTP method

6.1. Proposed iWHC Data Clustering Solution

As discussed in section 3.4. of Chapter 3, given the nature of the data found in production Grid environments, it is difficult to find a function that fits the content of an iWHC. Therefore, it is proposed to use a method which tries to identify data clusters containing stable data, rather than using a method that fits a curved data trend, such as the statistical average, the response time linear regression, or the polynomial analysis.

The QT clustering method [103] has the advantage that it does not need the number of clusters specified a priori, and it also considers a level of the quality threshold that guarantees that all data is within that range. But, the QT requires an important computing power to run on large iWHC and also it does not cluster continuous data values.

In order to overcome these two problems, the QT method is adapted and presented in the proposed Time Based Quality Threshold (TBQT) clustering method. TBQT is developed with the response time prediction service in mind and, therefore, the method emphasises the desire to discover clusters that have a guaranteed quality threshold. Those clusters represent a period of time where the job has performed with similar response time in a clear stable state where no other factor has affected the job's response time.

The difference in the order that the data is analysed reduces the complexity of the algorithm and, as a result, the TBQT method requires less computer power. Consequently, TBQT can be used to analyse large iWHCs. In addition, the TBQT cluster threshold can be changed to increase or decrease the granularity of the data analysis. This is a valid option and should be tuned depending on the Grid environment. A Grid environment that runs large jobs (in response time terms) can implement a large clustering threshold in order to group together similar job executions. A small clustering threshold is better indicated for Grid environments that run short jobs at the cost of increasing future performance calculus.

6.1.1. Proposed Time Base Quality Threshold Clustering Algorithm

Definition: A *diameter* of a cluster is the absolute value of the difference between the maximum and the minimum response time data in the cluster.

The TBQT method is based on the QT solution. QT briefly works as follows: a candidate cluster is formed by starting with the first data and subsequently adding any other data that minimises the increase in the cluster diameter. The process continues until no data can be added without surpassing the diameter threshold. A second candidate cluster is formed by starting with

the second data and repeating the procedure described above. Note that all data are made available to the second candidate cluster. The process continues for all data being clustered. A set of candidate clusters are created at the conclusion of this stage. At this point, the largest candidate cluster is selected and retained. The selected data is removed from consideration and the entire procedure is repeated on the reduced set.

The TBQT method takes a different approach at the time of data gathering: The first step is to consider each record of data as a continuous cluster. The next step is to find two continuous clusters that when merged together meet the given threshold and also have the minimum diameter of all combination of cluster pairs. If there is a candidate pair, then both clusters are merged. If two clusters meet both requirements, then the number of clusters is reduced by one. Now the entire procedure is repeated on the new set of clusters. The procedure outputs the result if there are no two clusters that, when merged, meet the threshold. The pseudo code is shown in Algorithm 3.

TBQT_Clustering(instanciatedWHC $iwhC$, threshold t)
    For each $iwhC_i \in iwhC$ do
        Set $CiwhC_{j,0} = iwhC_i$
    Call TBQT_ClusteringRecursiveCall($CiwhC$, t)

TBQT_ClusteringRecursiveCall(clusteredInstanciatedWHC $CiwhC$, threshold t)
    If $|CiwhC| \leq 1$
        Return $CiwhC$
    Else
        Find $j / 0 \leq j < |CiwhC|$ such that
            the diameter $(CiwhC_j \cup CiwhC_{j+1})$ is minimum
        If $(diameter(CiwhC_j \cup CiwhC_{j+1}) > t)$ then
            Return $CiwhC$
        Else
            Set $CiwhC_j = CiwhC_j \cup CiwhC_{j+1}$
            Set $CiwhC = CiwhC - CiwhC_{j+1}$
            Call TBQT_ClusteringRecursiveCall(Set $CiwhC$, t)

Algorithm 3. Time Based Quality Threshold (TBQT) clustering.

As requested, the algorithm is sensitive to the time based order in which the data appears, and continuous data value is clustered together with this new method. Moreover, the total

number of clusters is not needed at the start of the algorithm, all of the clusters achieve a quality guarantee, and the running cost is reduce to $(n \times (n-1))$, order n^2 for n elements to cluster.

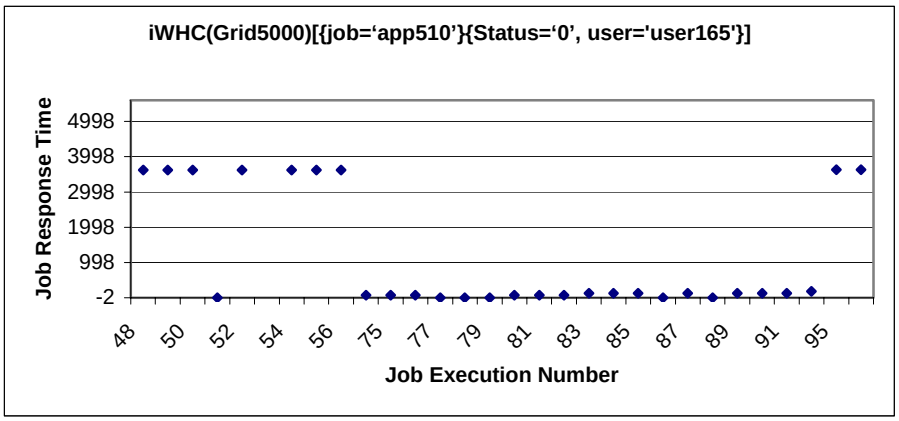


Figure 28 Response time data profile in an iWHC.

An example of the response time data of a random iWHC taken from Grid5000 is plotted in Figure 28. The results of the TBQT algorithm applied to this data can be seen in Table 23, where three different thresholds (600, 180, and 60 seconds) are offered as columns. For information purposes, each cluster also shows the average response time.

In these results, the smallest threshold, 60 seconds, divide the iWHC in several consecutive sections, each of them clearly representing a continuous stable historical list of job submissions. The -2.0 value in the response time column indicates the job has failed. The values generated by failed jobs are noticed by the TBQT algorithm and clustered together.

The largest threshold, 600 seconds, produces 7 clusters compared to the 13 created by the smallest option. This difference represents a reduction in the final running cost of the prediction solution working with a large iWHC. However, this threshold puts slightly different response time records together. This point may also have an implication in the prediction solution.

Data		Clusters		
eid	Response Time	Threshold 600 (Sec)	Threshold 180 (Sec)	Threshold 60 (Sec)
48	3618.0	Cluster 6, Avg. 3618	Cluster 7, Avg. 3618	Cluster 12 Avg. 3618
49	3619.0			
50	3619.0			
51	-2.0	Cluster 5 Avg. -2	Cluster 6 Avg. -2	Cluster 11 Avg. -2
52	3617.0	Cluster 4 Avg. 3617	Cluster 5 Avg. 3617	Cluster 10 Avg. 3617
53	7000.0	Cluster 3 Avg. 7000	Cluster 4 Avg. 7000	Cluster 9 Avg. 7000
54	3617.0	Cluster 2 Avg. 3618	Cluster 3 Avg. 3618	Cluster 8 Avg. 3618
55	3620.0			
56	3617.0			
74	74.0	Cluster 1 Avg. 78	Cluster 2 Avg. 62	Cluster 7 Avg. 73
75	73.0			
76	74.0			
77	-2.0			Cluster 6 Avg. -2
78	-2.0			
79	-2.0			
80	75.0			Cluster 5 Avg. 99
81	74.0			
82	74.0			
83	125.0			
84	125.0			
85	125.0			
86	-2.0			Cluster 4 Avg. -2
87	126.0			Cluster 3 Avg. 126
88	-2.0			Cluster 2 Avg. -2
89	124.0		Cluster 1 Avg. 137	Cluster 1 Avg. 137
90	124.0			
91	123.0			
92	179.0			
95	3629.0	Cluster 0 Avg. 3628	Cluster 0 Avg. 3628	Cluster 0 Avg. 3628
96	3627.0			

Table 23 TBQT for iWHC(Grid5000)[{job=app510}{Status=0, user=user165}]

6.1.2. Analysing the Clustered Results

In Figure 29 different data charts are showing the result of applying the TBQT clustering algorithm to an iWHC for the application App510 that last run on the site G1/site6/c1 on Grid5000. The algorithm used 300 seconds as a threshold diameter.

The first chart shows the full list of historical data where several changes can be seen. Next to the main chart in Figure 29, there are six small charts representing the results in each cluster. The clusters have been labelled from the current date to the past, starting with Cluster 0. Each of the clusters can be visually mapped to the general chart of data. To simplify the explanation, note that 3 clusters with 1 data record each and with a response time value of -2 (execution

error) have not been printed. These 3 clusters represent local outliers that can be seen in the job execution axis with identification numbers 278, 337 and 362, respectively.

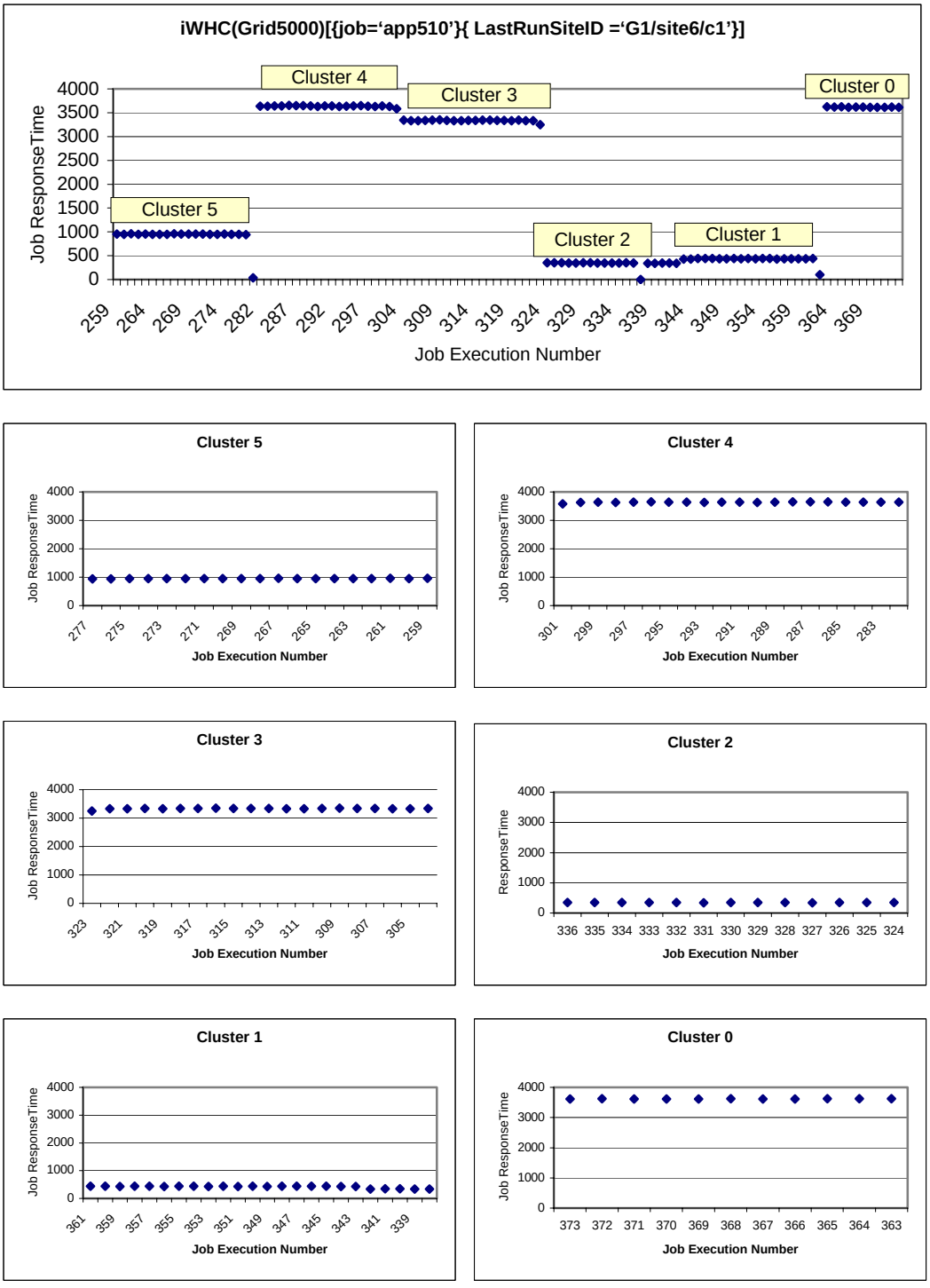


Figure 29 TBQT clustering method for an iWHC

This result shows how the algorithm understands the different changes in the iWHC historical data. A possible point to discuss would be that Cluster 2 and Cluster 1 could have ended up in the same cluster if one would have added a method that discards outliers to the clustering algorithm. However, as will be described in the next section, the response time prediction methods are taking these cases into account.

One of the key points that this research endeavours to provide is a solution that can be used in a real environment. Consequently, the solutions should answer to a prediction request within a realistic time. To put this expression into context, to perform the GRTP full life cycle for a prediction request, the chosen clustering method should be run for each iWHC to calculate the iWHC accuracy level. The average number of iWHC for the selected testbeds is eight (Chapter 5 Section 5.2.3.). For that reason, the time shown in the performance comparison chart should be multiplied by this number. Furthermore, after successfully running the GRTP service for 30,000 prediction requests it was determined that the average number of historical records in an iWHC is 339 for Grid5000, 597 for DAS and 2851 for AuverGrid

Figure 30 charts the TBQT and QT run time comparison that was analysed in the same resource environment. In this figure, the increase in run time for the TBQT method is lineal (making it suitable for a production Grid environment), while the QT algorithm has an exponential execution timeline.

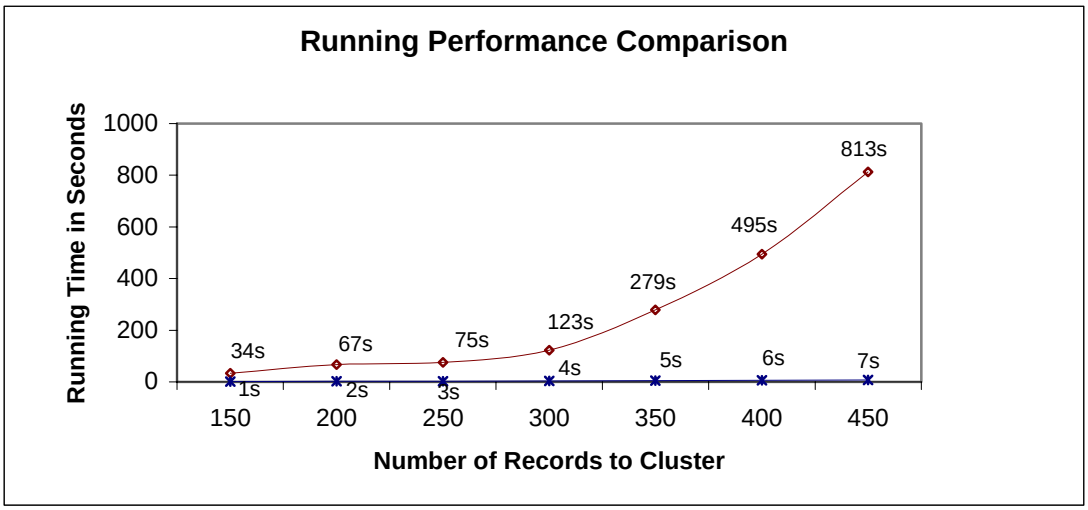


Figure 30 TBQT vs. QT: Clustering algorithm running performance comparison.

6.2. The Response Time Prediction Methods

At this point, the proposed solution has mined the historical data to select the set of fields that provides information (WHC), the most relevant set of historical records (iWHC classification using reinforcement learning), and the change in behaviour that a job has presented in continuous historical submission within each iWHC (Clustering using TBQT).

The next and final step uses this information to answer a response time prediction request. The first step divides the universe of jobs into two:

- The first job type has an iWHC where the data clusters tend to stabilise at a common point. In other words, the different clusters are likely to reach a common average point into the future. This type of behaviour is usually seen when a resource component or job parameter is tuned over the course of different submissions to reach a stable historical performance.
- The second job type has an iWHC with data clusters without a well-defined tendency. This job type is common in production Grid environments and they are generally produced by continuous submission of jobs with different parameters not usually published in the workload traces.

Both job types conform to the universe of possibilities that the GRTP method considers in production Grid environments. A job may have more than one iWHC and consequently it can have two different classifications. However, these classifications can change if new data is produced and added to the iWHC.

6.2.1. Job Classification: Stable or Unstable

The clusterWeight function is introduced before explaining the job classification method. This function weighs a cluster considering the amount and age of the data inside each cluster.

The clusterWeight function briefly works as follows: the method initially counts the number of records in the cluster (clusterSize functions). After that, the weight of the cluster is calculated adding up the arithmetic divisions of the clusterSize (dividend) and the age of each record (divisor). The age of each record is expressed by a number. The age starts from 1 (current record) and increases by 1 for each record towards the past. The pseudo code for the algorithm is shown in Algorithm 4.

The clusterWeight function produces a value that reflects that a similar cluster gets a superior weight if the records are new. On the other hand, this value can be compensated if the

number of records is bigger in an older cluster. These details are displayed in Table 24, where three clusters with the same size get weighed differently because of the age of the records.

clusterWeight (cluster C)
    Set cS = clusterSize(C)
    For each $d \in C$ do
        Set clusterWeight = clusterWeight + (cS / *Age*(d))
    Return clusterWeight

(*Age* is a function that returns a value that increases while the information gets older)

Algorithm 4. Weighting a data cluster.

Cluster 2	
Age	Local Weigh
12	0.3333
11	0.3636
10	0.4000
9	0.4444
Cluster Weigh:1.5414	

Cluster 1	
Age	Local Weigh
8	0.5000
7	0.5714
6	0.6667
5	0.8000
Cluster Weigh:2.5381	

Cluster 0	
Age	Local Weigh
4	1.0000
3	1.3333
2	2.0000
1	4.0000
Cluster Weigh:8.3333	

Table 24 ClusterWeight function for 3 clusters with different age

Definition: An iWHC is classified as *Unstable* if there is a previous cluster in the job history whose weight is bigger than the latest cluster, and also if the diameter of the previous cluster and the newest cluster together is bigger than a given threshold. Otherwise, the iWHC is classified as *Stable.*

Explained in a different way, an iWHC is unstable because a significant part of the job's history (given by a heavier clusterWeight function) is separated from the current response time tendency.

isStableiWHC (clusteredInstanciatedWHC $CiwhC$, threshold t)
    If $|CiwhC| > 0$
        Set vClusterWeight = clusterWeight($CiwhC_0$)
        For each $k = 1; k < |CiwhC|$ do
            If(vClusterWeight < clusterWeight($CiwhC_k$))
                If($diameter(CiwhC_0, CiwhC_k) > t$)
                    Return(false)
    Return(true)

Algorithm 5. Function to classify an iWHC as Stable or Unstable.

The method that implements the classification is described as follows: If the iWHC has more than one cluster, the clusterWeight function is calculated for the first cluster (cluster 0). For any of the following clusters, the clusterWeight functions are compared against the cluster 0. This comparison helps to identify a past cluster that is strong enough, in quantity and time, to still be relevant for the prediction method. If the comparison provides a heavier past cluster, the diameter of the cluster 0 together with the past cluster is calculated. If the resulting diameter is bigger than a given threshold, the iWHC is classified as unstable. The pseudo code for the algorithm is shown in Algorithm 5.

A data example of a stable iWHC(Grid5000)[{job=app510}{}] with a threshold of 60 seconds that has produced 34 data clusters is plotted in Figure 31. Each chart shows the values that are considered in the isStableiWHC function: the quantity of records in each data cluster (chart 1), the response time average in each data cluster (chart 2) and the final cluster weight of each data cluster (chart 3). The weight of the current cluster (cluster 0) is the biggest because of its age (new) and the number of records that contains (19 records as in chart 1) with a response time average of 386 seconds (as in chart 2). None of the previous clusters are heavier than cluster 0. Many of them contain only one record and can therefore be labelled as outliers. Analysing the charts shows that the outliers are discarded when the weight function is applied. As a result, this iWHC is classified as stable.

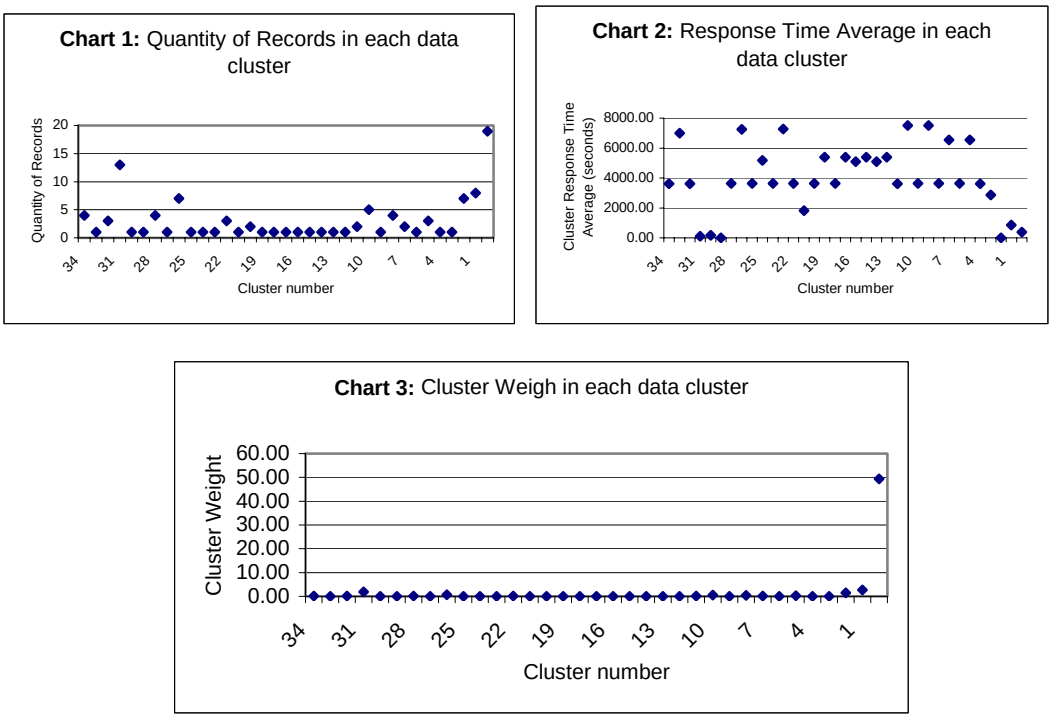


Figure 31 Data charts for an iWHC classified as Stable

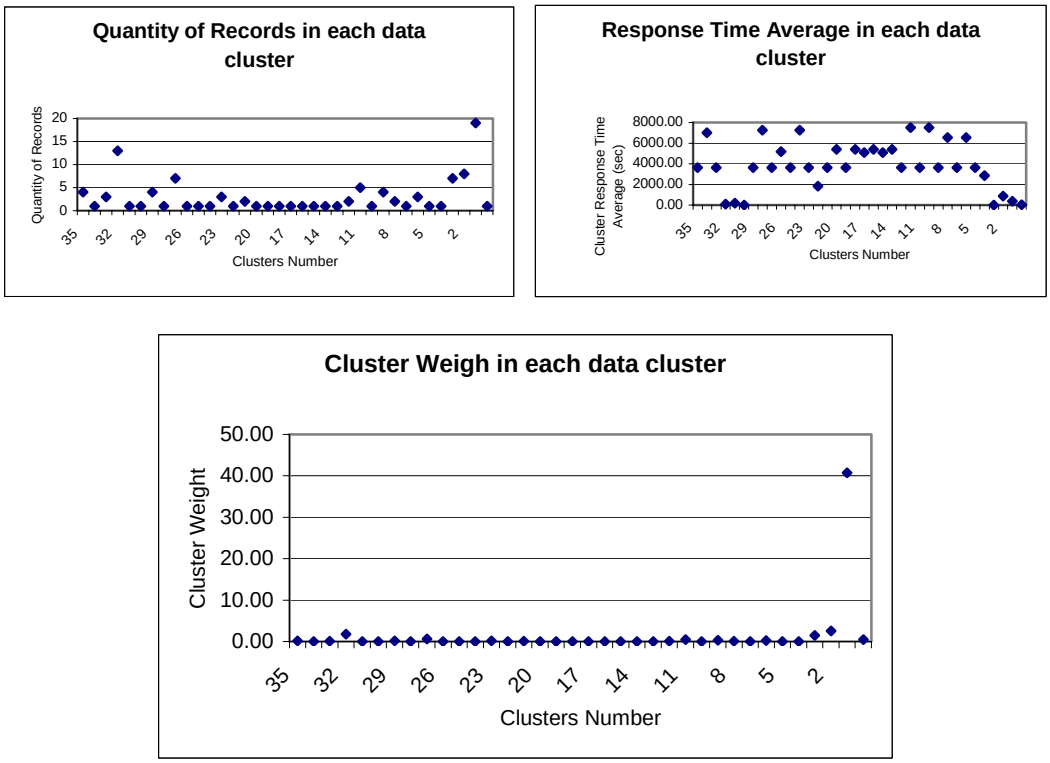


Figure 32 Data charts for an iWHC classified as Unstable.

The example in Figure 31 is followed by including new job executions (Figure 32). The last chart shows that the weight of the new cluster (new cluster 0) is less important than that of previous clusters. This is due to a change in the most recent job history and now the cluster 0 contains 1 record with an average response time of 38 seconds, where the previous cluster 0 had an average of 386 seconds. Since it is not known whether the job will continue with this trend, or it is just an outlier, the iWHC is classified as unstable.

6.2.2. Proposed Response Time Prediction for Stable iWHCs.

Once the historical data has been selected, current response time prediction solutions [53][54][61][62][63] generally apply an average or a linear regression function to obtain the predicted result. Bearing the data profile found in an iWHCs in mind (Chapter 3, Section 3.4.), an average function would be influenced by outliers and would not consider any trend in the current data. A weight average only improves the previous function by considering the age of the data. A linear regression approach is less influence by outliers and follows a trend in the data line, but it does not take advantage of the cluster of data produced by job submission

changes found in an iWHC. As a result, it is proposed to use a weighted mean of the clusters in an iWHC.

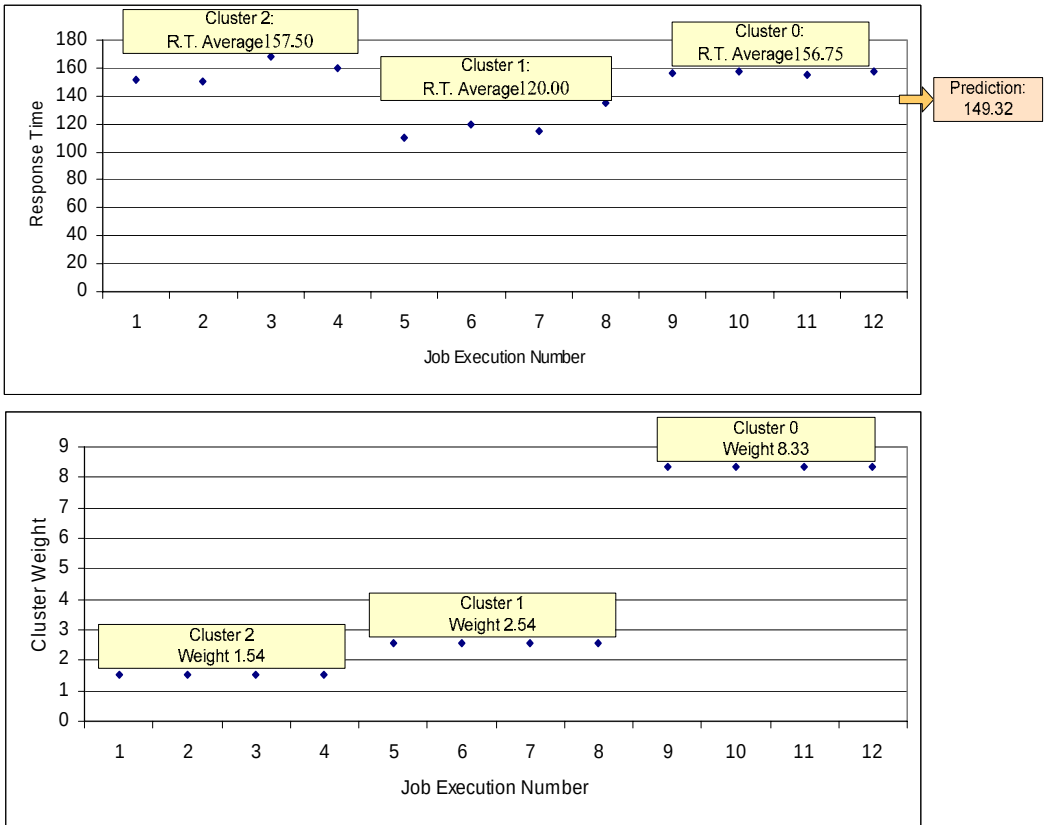


Figure 33 Data charts for a response time prediction method for a Stable iWHC.

This method is first introduced for a stable job's iWHC. The first chart in Figure 33 displays the different clusters found by the TBQT method. A tendency to stabilise the average of each cluster can be seen in this chart. The prediction method focused on stable jobs evaluates the clusters by their internal average and weight function. The prediction is finally composed by the weighted mean of all clusters in an iWHC. The weighted mean is similar to an arithmetic mean, where instead of each of the data points contributing equally to the final average, some data points contribute more than others. The relative importance of the internal mean of each parameter is given by the weight function.

Definition: w_i is the returned value for the weight function for the cluster i, Avg_i the average of response time for the cluster i, and n the number of clusters in an iWHC. The proposed response time prediction for a stable iWHC is defined as: $\frac{\sum_{i=0}^{n} w_i Avg_i}{\sum_{i=0}^{n} w_i}$

Summing up, the prediction method for stable iWHCs uses all clusters to predict the future response time. It evaluates the importance of each cluster based on its quantity and age. This means that a cluster containing newer data has more influence than other clusters containing older data. However, this influence may be balanced if the amount of data in an old cluster is more important than data in a new cluster.

```
ResponseTimePredictor_StableiWHC (clusteredInstanciatedWHC CiwhC )
        For each k = 1; k < |CiwhC| do
                Set cW = clusterWeight ( CiwhC_k )
                Set cA = clusterReponseTimeAverage ( CiwhC_k )
                Set divisor = divisor + cW * cA
                Set dividend = dividend + cW
        If (dividend > 0)
                Return divisor/ dividend
        Else
                Return -1
```

Algorithm 6. Response time prediction method for a stable iWHC.

Cluster 2: Weight 1.54		Cluster 1: Weight 2.54		Cluster 0: Weight 8.33	
Age	Response Time	Age	Response Time	Age	Response Time
12	152	8	110	4	156
11	150	7	120	3	158
10	168	6	115	2	155
9	160	5	135	1	158
R.Time Average: 157.50		R.Time Average: 120.00		R.Time Average: 156.75	
Response Time Prediction: 149.32					

Table 25 Response time prediction example for a stable iWHC

The pseudo code for the algorithm is shown in Algorithm 6 and an extended result is displayed in Table 25. From the results one can see that the iWHC has been classified as stable and consequently the entire list of clusters is used. The weight function gives more significance to the latest cluster compared to past clusters. Consequently, the predicted value receives a

decreasing influence from all data records in the iWHC from the present to the past. The predicted response time value is finally calculated selecting the most accurate iWHC (as defined in Chapter 5, Section 5.3.) for all the iWHCs in the job. The remaining predicted response time values are used in the post-mortem activities, outlined in the next section, to calculate the iWHC accuracy level.

6.2.3. Proposed Response Time Prediction for Unstable iWHCs

Unstable iWHCs are unlikely to reach a common response time point in the future. For that reason, the proposed method attempts to reduce the prediction error by focusing on the list of clusters which are within a certain specified diameter from the newest one. When the method encounters an unknown suggestion about the future tendency of the job, it tries to continue with the current job trend until the iWHCs get back to a stable situation.

```
ResponseTimePredictor_UnstableiWHC(clusteredInstanciatedWHC CiwhC , threshold t )
    If |CiwhC| < 1
        Return -1
    Else
        Set RCiwhC_0 = CiwhC_0
        Set i = 1
        For each k = 1; k < |CiwhC| do
            If( diameter(CiwhC_0, CiwhC_k) ≤ t )
                Set i = i + 1
                Set RCiwhC_i = CiwhC_k
        Return(Prediction_StableiWHC( RCiwhC ))
```

Algorithm 7. Response time prediction method for an Unstable iWHC.

Therefore, the method only considers the clusters that are within the same threshold of the newest cluster, rather than using all available clusters which may lead to an error in the response time prediction. The pseudo code for this response time predictor is shown in Algorithm 7. The algorithm only selects the clusters that once added to the latest cluster (tendency cluster) have a diameter that is below a certain specified threshold. Consequently, this method supposes that the trend shown by the latest cluster is the correct guide line to use to predict the future job trend. The response time prediction method for a stable iWHC is called once all available clusters have been evaluated and filtered.

This function also selects the response time value produced by the most accurate iWHC as an answer. The remaining response time values are used in the post-mortem activities, which are explained in the next section, to calculate the iWHC accuracy level.

Figure 34 and Table 26 shows an example of an unstable iWHC prediction with a threshold of 60 seconds. In this case, the latest cluster has only one previous cluster within the same threshold. As a result, the weighted mean is restricted to use cluster 0 and cluster 2.

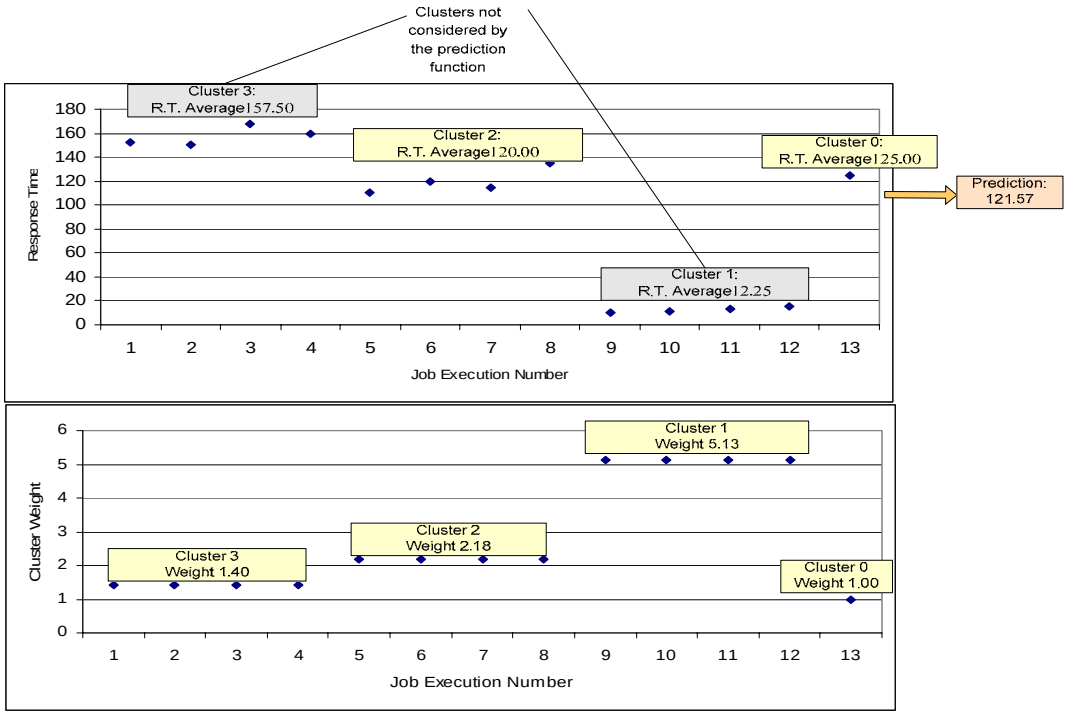


Figure 34 Data charts for a response time prediction method for an unstable iWHC.

Cluster 3: Weight 1.40	
Age	Response Time
13	152
12	150
11	168
10	160
R.Time Average: 157.50	

Cluster 2: Weight 2.18	
Age	Response Time
9	110
8	120
7	115
6	135
R.Time Average: 120.00	

Cluster 1: Weight 5.13	
Age	Response Time
5	10
4	11
3	13
2	15
R.Time Average: 12.25	

Cluster 0: Weight 1.00	
Age	Response Time
4	125
R.Time Average: 125.00	

Response Time Prediction: 121.57

Table 26 Response time prediction example for an unstable iWHC

6.3. Post-mortem Activities

A number of post-mortem activities are carried out in order to complete the full cycle of the GRTP method. These activities guarantee the tuning of the GRTP service and also provide information about its prediction accuracy. The post-mortem activities are included in the GRTP method are:

- As it was described in Chapter 5 Section 5.3., the iWHC selection uses its accuracy level. The task for the reinforcement learning agent is to learn which the most relevant iWHC is. The agent's policy uses the level of accuracy as a reward function. It converts each perceived reward into a single number that determines the iWHC selection. This value is generated by the GRTP method for each iWHC produced in a job prediction request. The iWHC accuracy level is calculated as a percentage by comparing the predicted result against the real job response time.
- The evaluation of the fields that compose the workload meta-model (Chapter 5 Section 5.2.) needs to run in the background depending on the number of new workload trace records. This step ensures that new fields are taken into account by the meta-model definition, and that fields which are no longer in use are excluded from future analysis.

Some of the post-mortem activities, such as the tuning of setting parameters, should be performed by an expert in the Grid computing response time prediction area. The GRTP method provides a number of parameters which can be adjusted to the particular Grid environment. These parameters are:

- A proposed iWHC selection method using a discounted accumulative reward function uses a value of γ that is a constant that determines the relative value of delayed versus immediate rewards. If γ gets close to 0, only the immediate reward is considered. As γ gets close to 1, past rewards are given greater emphasis relative to the immediate reward. γ should be tuned following the recommendations presented in Chapter 5, section 5.3.1.
- The workload meta-model threshold: a data field can be included in the prediction workload meta-model if the information that adds to the already selected set of fields is greater than a given threshold. This number can be tuned in order to allow more or less fields to be considered by the method. Using more fields means that the method will analyse more catalogues, and consequently possibly provide better accuracy in the response time results. However, more fields imply an increase in the overall running cost of the GRTP method.
- The clustering method threshold defines the granularity of the data clusters. Using a small threshold means that fewer data records will be clustered together, which in turn increases

the running cost of the GRTP method. A wise decision may be to consider the SLA to set the clustering threshold. The threshold of the clusters must be smaller than the maximum expected prediction error set by a SLA. Clusters with diameters smaller than the expected error allow the performance predictor to use historical data that belongs to a certain threshold. This threshold is smaller than the maximum expected prediction error agreed in the SLA. But, setting the threshold according to a SLA does not guarantee an acceptable prediction result. This is because historical data may not be available for the specific threshold.

- Most importantly, the clustering granularity must be set according to the SLA that the Execution Planning Service is set to meet. The clustering method threshold the allowed by a given SLA.
- Response time prediction threshold for an unstable iWHC: This setting parameter plays a similar role to the clustering method threshold, and for the purpose of this research both thresholds were set to the same value. Having said that, the GRTP method affords separate tuning of each setting parameter independently. In this case, a small threshold implies that fewer clusters in an iWHC are considered by the unstable prediction method.

6.4 Summary and conclusions

This chapter complements Chapter 5 by explaining the final section of the GRTP method. It presents a meticulous description and comparison of the data mining methods used to predict the response time in a Grid environment. The key contributions of this chapter are:

- A clustering algorithm called Time Base Quality Threshold focuses on finding patterns in unlabeled data, and which also reduces the complexity of the final response time prediction method.
- A method that classifies iWHC as stable or unstable, and as a consequence it also classifies the jobs represented by the iWHCs.
- A method for predicting a job response time for a stable iWHC. The prediction method focused on stable jobs evaluates the clusters by the internal average and weight function. The prediction is composed by the weighted mean of all clusters in the iWHC.
- A method for predicting a job response time for an unstable iWHC. This method tries to continue with the current job trend until a change occurs in the iWHC. The method only uses the clusters that are within the same threshold as the newest cluster.

- A description of the list of setting parameters for the prediction method. Those parameters can be tuned to adapt the solution to the Grid environment. A practical analysis of the impact of those parameters in the prediction results is analyzed in the next chapter.

Chapter 7 – GRTP Method Experimental Results and Analysis

This chapter presents a series of experiments carried out to test the GRTP Method response time prediction. Each of the experiments produces several results that can be used to judge the quality, reliability and consistency of the proposed solution.

The performed experiments implement a cross-validation technique [111]. Cross-validation is applied to assess how the results of a statistical analysis will generalise to an independent data set. The technique is mainly used in situations where one wants to estimate how accurately a predictive model performs in practice. One round of cross-validation involves partitioning a sample of data into complementary subsets. The analysis is performed on one subset referred to as the *training set*, and the analysis is subsequently validated on the second subset called the *validation set or testing set*. To reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are averaged over the rounds.

The cross-validation method used in this research is called *holdout*, and the data was separated into a disjoint training set and a testing set. The training set consisted of a minimum of 20,000 job submissions taken from DAS. The two testing sets were taken from the AuverGrid and Grid5000 Grid environments and were also composed of a minimum of 20,000 job submissions each.

7.1. Comparison Charts Explanation

Two different chart types are used regularly throughout this chapter. The main objective of make use of a consistent set of charts is to facilitate the assessment of the GRTP service when different running options are implemented.

The first chart is used to evaluate a specific response time prediction solution. The evaluation is focused on measuring the quality of the prediction result against the real response time result. The X axis shows a range of response time prediction errors. To have a clear picture of the method's prediction accuracy, the granularity of the ranges of errors increases over the axis, where the ranges that are close to zero have a small granularity. It is important to notice that the same range of the X axis is used in all charts for all experiments; therefore it is possible to run a visual comparison between the charts. The Y axis shows a percentage of occurrences that are fitted into each error range. An example of the chart described is printed in Figure 35.

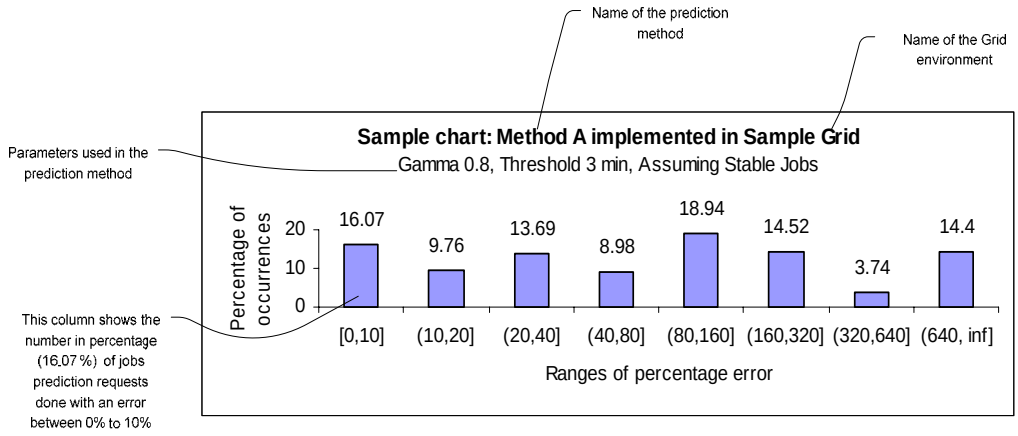


Figure 35 Sample chart: Results for a Method A in Sample Grid.

Figure 36 shows the second chart type commonly used in this chapter. This is a comparison chart that shows how one method has performed against another. The chart illustrates how well (in percentage terms) each response time prediction method has performed when comparing the response time predictions that one method has calculated more accurately than the other (more accurately, in this case, means that the value of the predicted response time for one method is closer to the actual job response time than that of the other method). Within this chapter, this chart is always presented twice when comparing two methods. One chart is to show how well a first method has performed against another; and the second one shows the same information from the opposite perspective, i.e., how well the second method has performed against the first one.

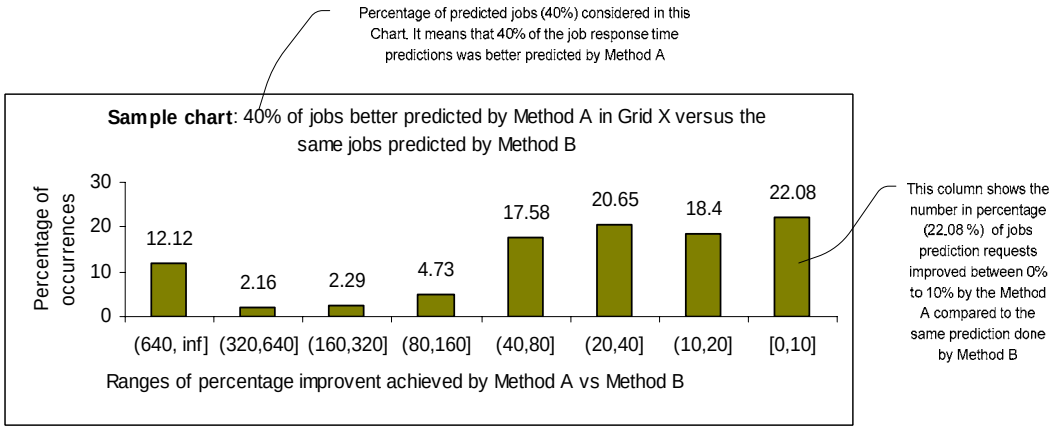


Figure 36 Sample chart: Shows how well a Method A has predicted against a Method B.

For example, Figure 36 shows that *Method A* has predicted 40 percent of the prediction requests better than *Method B*. Out of those 40 percent, 22 percent has been improved by 10

percent by *Method A* against *Method B*, 18 percent has been improved by 20 percent by *Method A* against *Method B*, etc. This chart offers the chance to visually compare two methods reading: the columns closer to the left side represent a better improvement compared to the columns closer to the right side of the chart**.**

7.2. Evaluation of Existing Solutions

Chapter 2 mentions that workload traces in response time prediction methods were used by Downey [53] , Gibbons [61] and Smith [62]. By comparison, it was demonstrated that [63] Smith's method was the most effective and accurate method. Smith was able to deal with distributed environments without much intrusion and without imposing restrictions on the input data fields. Downey, Gibbons and Smith's methods were further compared and evaluated in a production Grid environment in [76]. Again, Smith proved to be most accurate solution. However, it was also demonstrated that there are a number of key points where response time prediction solutions for Grid environments can be improved. For that reason the GRTP method was presented. The GRTP method is a dynamic and non-intrusive solution able to understand the input data and evolve as required.

DAS Grid environment was selected as the training set for the evaluation. DAS represented a difficult challenge for the prediction methods given that the historical data offered common variances of the response time for a given job. The causes of those response time variances were mainly not inform in any other data field of the historical data. This phenomenon was observed when the data was mined in detail to appreciate how difficult the challenge was before hand.

Taking this observation into account, 20,000 job submissions were predicted to test how the most accurate of the current existing solutions (Smith [62] using arithmetical average for the prediction function) perform in a production Grid computing environments (DAS). The results are shown in Figure 37 and Table 27.

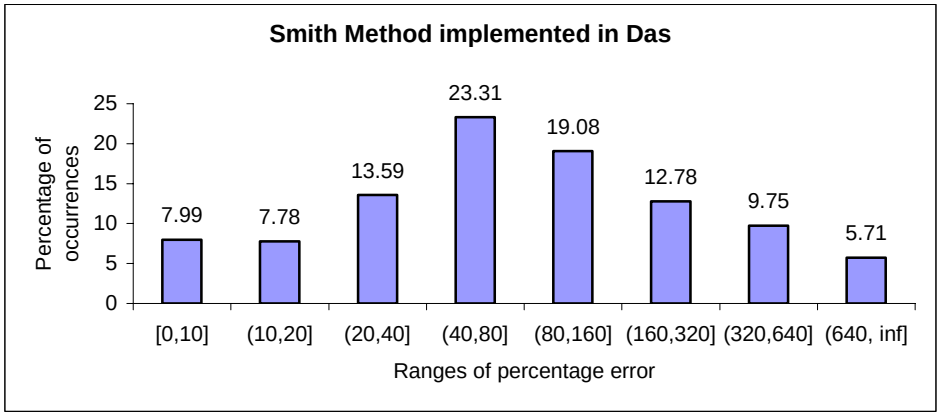


Figure 37 Smith method in DAS.

DAS	
Catalogues	% Use
{[JobID,LastRunSiteID,Nproc,UserID]}	13.01
{[JobID,Nproc, UserID]}	13.01
{[Nproc, UserID**]}**	11.65
{[LastRunSiteID,Nproc, UserID**]}**	11.65
{[JobID, UserID]}	8.67
{[JobID,LastRunSiteID, UserID]}	8.67
{[LastRunSiteID,UserID]}	8.09
{[UserID]}	8.09
{[JobID,Nproc]}	4.91
{[JobID,LastRunSiteID,Nproc]}	4.91
{[Nproc]}	2.31
{[LastRunSiteID,Nproc]}	2.31
{[JobID]}	1.34
{[JobID,LastRunSiteID]}	1.34
{[LastRunSiteID]}	0.02

Table 27 Smith method catalogues and its usability in DAS

It has been previously mentioned that the main cause for the prediction errors in Smith's method is, that the use of any possible combination of data fields to filter the historical information in heterogeneous workloads is sensitive to non-normalise data. As a result, the selected records are not related to the prediction request.

Smith's method proposes the selection of a catalogue containing the minimum confidence interval of the response time, and the average of the response time within the selected catalogue as a prediction function. In this experiment, the prediction requests details produced by Smith's method shows that most of the errors were produced because the selected catalogue did not contain data fields logically connected to the prediction request. This argument is appreciated in Table 27 where the utilisation of each catalogue is displayed. Table 27 shows that the

catalogues {[Nproc,UserID]}, {[LastRunSiteID,Nproc, UserID]}, {[LastRunSiteID, UserID]} were utilised in 31 percent of the cases in order to predict a given job response time.

As a result, the prediction median error of Smith's method in the DAS environment was 76 percent. Taking these numbers into account (Figure 37), the immediate consequence is that the OGSA Execution Planning Service should expect that any prediction request will have a high level of imprecision, almost as big as the real job response time itself (100 percent).

The same experiment was also conducted this time using the testing sets offered by Grid5000 and AuverGrid Grid environments. The result for Grid5000 is presented in Figure 38, Chart A and Table 28. In this case, the prediction figures have improved considerably compared to the previous example where half of the jobs were predicted with 20 percent of error. However, the results for AuverGrid are not encouraging and present a significant number of submissions with prediction errors of 640 percent or more (Figure 38, chart B and Table 28). No further comparison is needed in order to demonstrate that Smith's method cannot be implemented in AuverGrid.

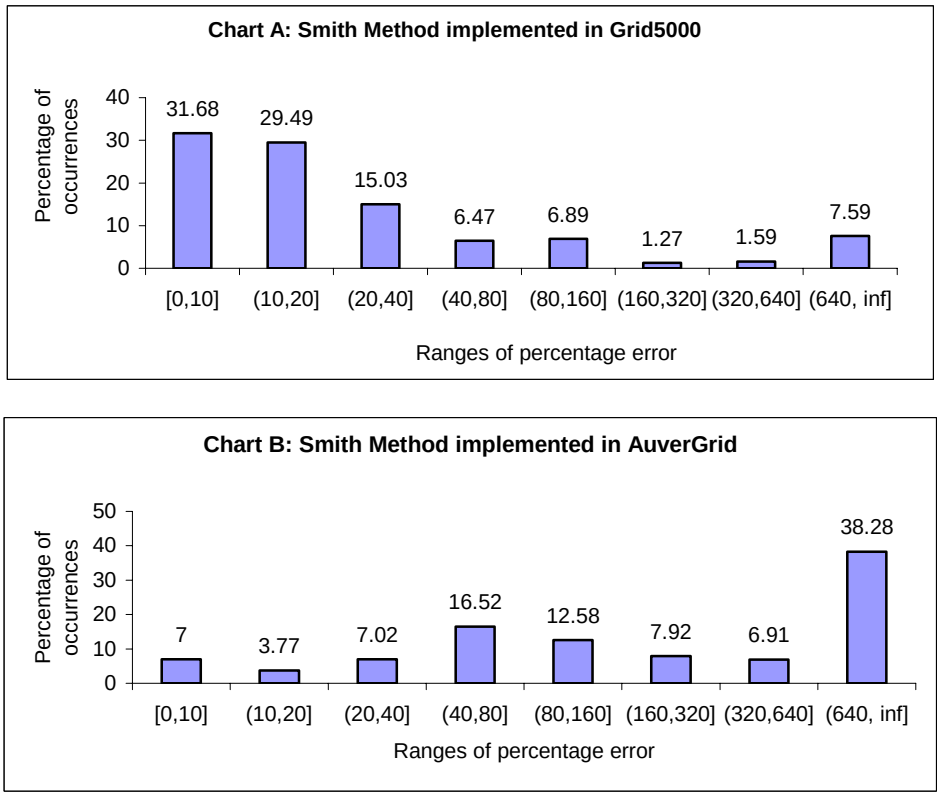


Figure 38 Smith method in Grid5000 (Chart A) and AuverGrid (Chart B).

Grid5000		AuverGrid	
Catalogues	% Use	Catalogues	% Use
{[JobID,Status,UserID]}	33.79	{[JobID,QueueId,Status,UserID]}	14.14
{[JobID,LastRunSiteID,Status,UserID]}	33.79	**{[QueueId,Status,UserID]}**	14.13
{[LastRunSiteID,Status,UserID]}	4.07	{[JobID,Status,UserID]}	12.49
{[Status,UserID]}	4.07	**{[Status,UserID]}**	12.48
{[UserID]}	3.72	{[JobID,QueueId,UserID]}	7.54
{[LastRunSiteID,UserID]}	3.72	**{[QueueId,UserID]}**	7.53
{[JobID,LastRunSiteID,Status]}	2.70	{[JobID,QueueId,Status]}	6.32
{[JobID,Status]}	2.70	{[JobID,UserID]}	5.87
{[JobID,LastRunSiteID,UserID]}	2.35	{[UserID]}	5.86
{[JobID,UserID]}	2.35	{[QueueId,Status]}	5.41
{[JobID]}	1.63	{[QueueId]}	3.29
{[JobID,LastRunSiteID]}	1.63	{[JobID,QueueId]}	1.88
{[LastRunSiteID]}	1.44	{[Status]}	1.52
{[LastRunSiteID,Status]}	1.02	{[JobID,Status]}	0.78
{[Status]}	1.02	{[JobID]}	0.78

Table 28 Catalogues for Smith method in Grid5000 and AuverGrid

In these two cases, it is possible to understand the performance of Smith's method by analysing the data used in each prediction. The explanation for the poor results seen in AuverGrid is similar to the one presented for DAS. In DAS the catalogues were selected using data fields that, while offering the minimum confidence interval of the response time within the catalogue, are selecting records that are not related to each other.

Contrarily, the Grid5000 catalogues utilisation displayed in Table 28 shows that the job identification is included in the catalogues most used for Grid5000. The first two catalogues in Grid5000 that include the job identification are used in 67 percent of the predictions, compared to AuverGrid where the most utilised catalogues are not linked to the target job (Table 28 highlighted in bold).

These experiments validate the proposed workload meta-model method presented in Chapter 5 Section 5.2., where the creation of a binding set of fields that links the catalogues to the target job limits the freedom of data fields. Furthermore, the solution's different performance results confirm that the selected testbeds represent a wide spectrum of production Grid computing environments.

7.3. GRTP Method Evaluation Assuming All Jobs as Stable

The first evaluation of the proposed method is carried out assuming that all jobs are stable (Chapter 6 Section 6.2.1.). For this job type all clusters within an iWHC are used to predict the response time. The method evaluates the clusters by their internal average and weight function.

The weight function considers the age and the number of records in a cluster. The prediction is finally composed by the weighted mean of all clusters in an iWHC

In this experiment, two setting parameters can be tuned. The first parameter is γ where $0 \leq \gamma < 1$. γ is part of the proposed iWHC selection method that uses a discounted accumulative reward function (Chapter 5 Section 5.3.1.) γ determines the relative value of delayed versus immediate rewards. From a practical point of view, the tuning of γ should follow the following principles:

- If a Grid site has the tendency to be dynamic with constantly changing recourses and services, the practical suggestion is to give more priority to the latest iWHC accuracy levels assigning a value to γ tending to 0.
- Otherwise, in a less dynamic environment, a γ tending to 1 is able to make better use of past experiences and deal better with outliers.

The second adjustable setting parameter is the threshold of the clusters in an iWHC. This parameter was presented in the proposed TBQT clustering algorithm in Chapter 6 Section 6.1.1. The threshold parameter affects the final GRTP running cost; the bigger the threshold, the fewer the number of clusters created from an iWHC, which in turn represents a reduction in running cost. On the other hand, having a small threshold means that few data records will be clustered together and that more running cost will be added to the GRTP method.

Fine-tuning of these setting parameters is part of the post-mortem activities discussed in Chapter 6 Section 6.3. For this initial example, γ is set to 0.8 and the clustering threshold to 3 minutes. However, both parameters are changed later in this chapter in order to analyse their influence on the accuracy of the GRTP prediction method. At the end of this chapter, an analysis of the sensitivity of the parameters is presented to better understand their impact on the GRTP method.

Figure 39 and Table 29 display the result of the GRTP method in the DAS Grid environment assuming that all jobs are stable. The result shows a distribution with a median prediction error value of 53 percent. Initially this number may seem unsatisfactory for a response time prediction method. However, comparing it to Figure 37, where the results for the Smith method are shown, it is clear that the GPTR method offers a significant improvement over Smith's method, which has a median prediction error of 86 percent. The fact that the number of prediction errors is greater than 160 percent has been considerably reduced implementing the GRTP method is a significant point.

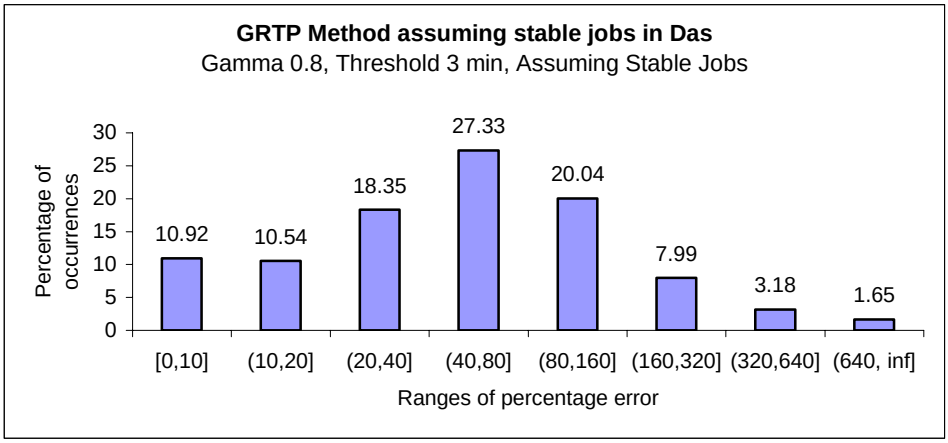


Figure 39 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) in DAS

DAS	
Catalogues	% Use
{[JobID]}	15.70
{[JobID,LastRunSiteID]}	15.70
{[JobID,LastRunSiteID,UserID]}	14.01
{[JobID,UserID]}	14.01
{[JobID,LastRunSiteID,Nproc]}	11.61
{[JobID,Nproc]}	11.61
{[JobID,LastRunSiteID,Nproc,UserID]}	8.69
{[JobID,Nproc,UserID]}	8.69

Table 29 WHC list for GRTP method in DAS (Gamma 0.8, threshold 3min).

A one-to-one evaluation of each response time prediction can be made in order to understand where the differences between both methods lie (Figure 40). The evaluation shows that the GRTP method assuming stable jobs predict better than Smith's method in 57 percent of the cases. The evaluation also shows that Smith's method predicts better than the GRTP method in 36 percent of the cases. Also, in Figure 40 can be seen that 7 percent of the predictions were calculated with the same precision using both methods. In these cases their scores were too close to be distinguished and therefore the results were not claimed for either method.

An interesting point is presented when the percentage range of improvements in the prediction errors is analysed. Chart A in Figure 40 shows that many job predictions have been considerably improved upon by the GRTP method. Comparing Chart A with Chart B in Figure 40 highlights the significance of the improvement; Chart B shows that when Smith's method is better the prediction improvements are much smaller than those of the GRTR method. These results suggest that the GRTP method not only predicts better than Smith's method in the majority of the prediction requests, but also that it affords a smaller prediction error.

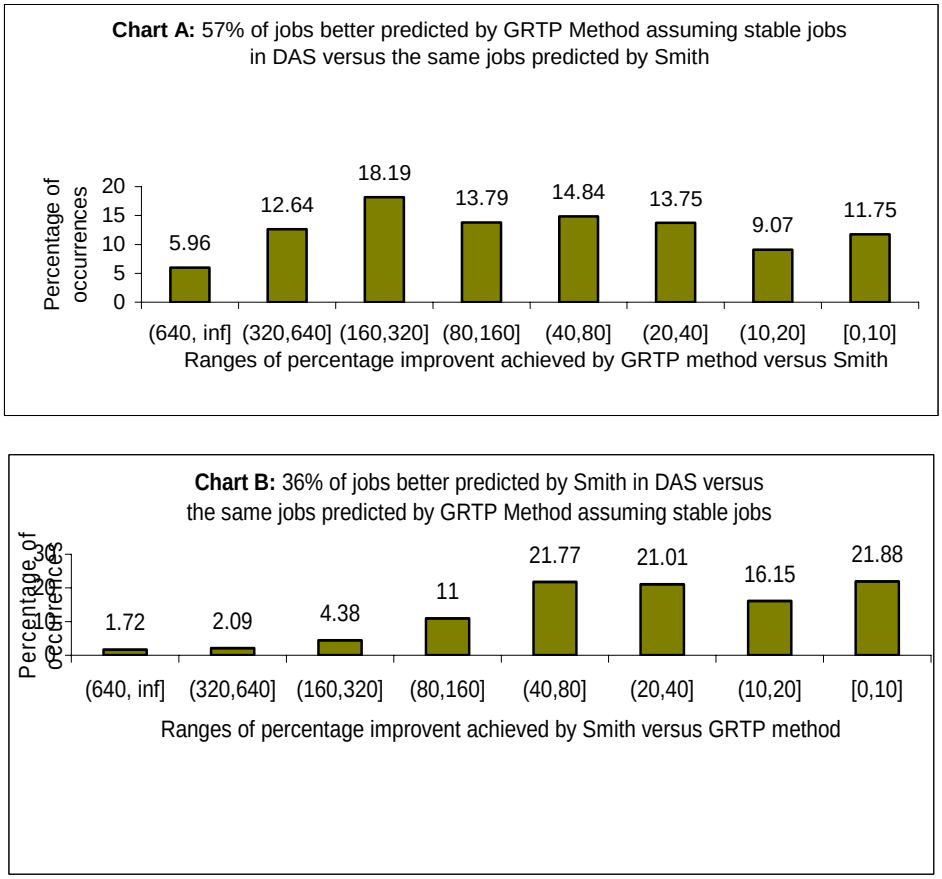


Figure 40 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) versus Smith method in DAS

The experiment was repeated in Grid5000 and AuverGrid in order to validate the results in different Grid environments. Figure 41 displays the result of the GRTP method in Grid5000. Comparing these results with Smith's for the same Grid environment (Figure 38 Chart A), shows that the median error for the GRTR method is 5 percent, compared to 16 percent for Smith's method. The comparison illustrates that the response time prediction has been improved by using the GRTP method. GRTP predicts 65 percent of the cases with a 10 percent error while Smith achieves the same prediction with a 20 percent error.

Figure 42 displays a one-to-one evaluation of the two methods in the Grid5000 Grid environment. The evaluation shows that the GRTP method predicts better than Smith's method in 67 percent of the cases. In most cases where the GRTP method predicts better than Smith's, the improvement is in the region of 20 percent. However, Smith's method offers a better response time prediction in 26 percent of the cases. Within the 26 percent, Smith improves the GRTP method by 10 percent or less. Following the experiment presented for DAS, the results in Grid5000 also suggest that the GRTP method not only predicts better in the majority of the

prediction requests but also that when it does it, the prediction error is smaller than Smiths results.

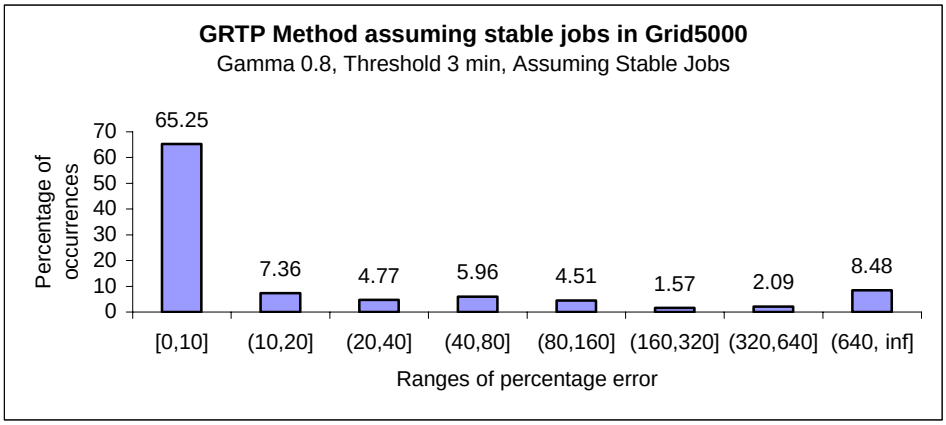
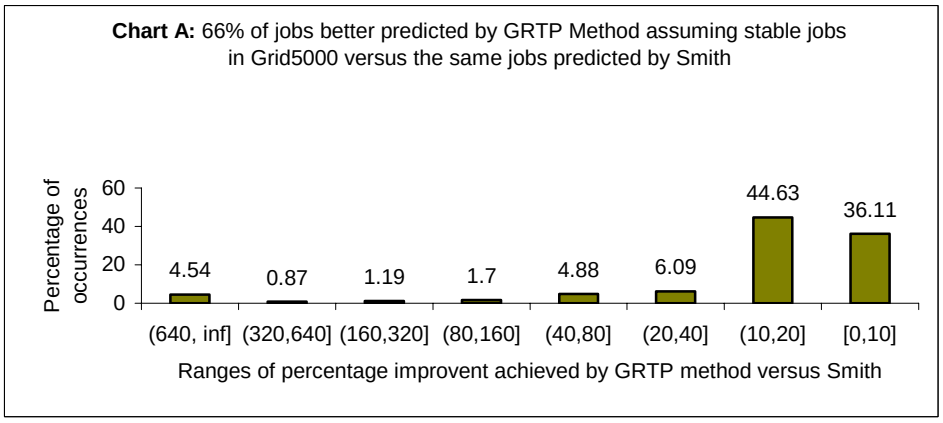


Figure 41 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) in Grid5000



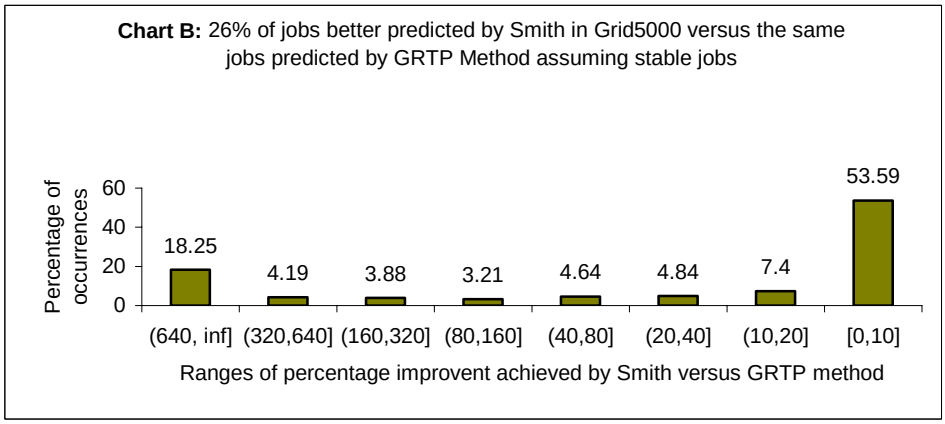


Figure 42 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) versus Smith method in Grid5000

The results of the comparison between the GRTR method and Smith's method in Grid5000 are similar to those found when the two methods were compared in the DAS Grid environment. The results suggest that the GRTP method offers a better response time prediction than Smith's method and that it affords a smaller prediction error.

The final experiment compares the GRTR method with Smith's method in the AuverGrid Grid environment. Figure 43 shows the result of the GRTP method. The median error for the GRTP method in AuverGrid is 64 percent, compared to 225 percent for Smith's method (Figure 38). From the AuverGrid results it is clear that the GRTR method offers a definite improvement in response time prediction over Smith's in this environment. Smith's method generates a significant number of predictions with errors greater than 640 percent. The GRTP method reduced the amount of errors and archived an increasing number of occurrences in other columns that represent a smaller prediction error.

Figure 44 illustrates the enhancement in response time prediction offered by the GRTR method from a different perspective. Figure 44 shows that more than 70 percent of the prediction requests are more accurate using the GRTP method compared to Smith's in AuverGrid. Furthermore, more than 50 percent of the predictions improved Smith's numbers by more than 640 percent.

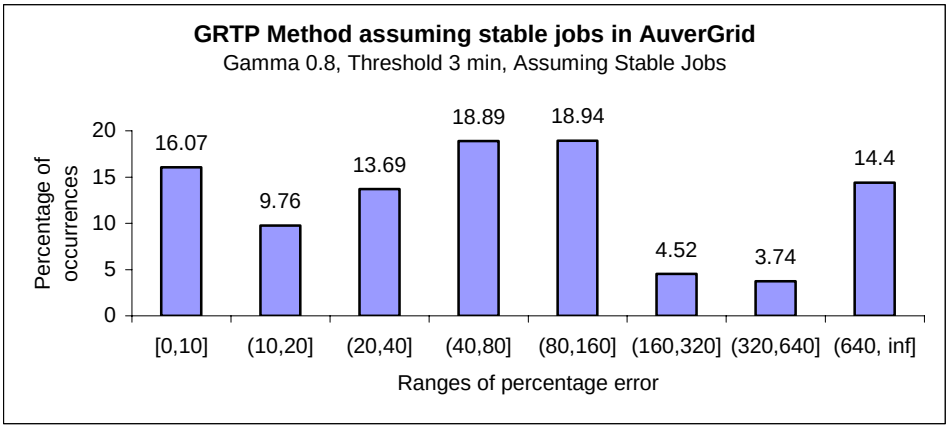


Figure 43 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) in AuverGrid

Figure 44 GRTP method assuming stable jobs (Gamma 0.8, thresholds 3min) versus Smith method in AuverGrid

7.4. GRTP Method Evaluation Assuming All Jobs as Unstable

The second evaluation of the GRTP method was implemented assuming that all jobs are unstable (Chapter 6 Section 6.2.1.). For this type of job all clusters within an iWHCs are used to predict the response time.

As previously explained in Chapter 6 Section 6.2.3., unstable iWHCs are unlikely to reach a similar response time at a point in the future. To prevent disturbances from clusters that may be created by outliers or important changes in the way that the job has performed in the past, this prediction method selects the clusters which are within a certain diameter of the latest one. The GRTR method for unstable jobs tries to continue with the current job trend when its future response time tendency is unknown. Therefore, instead of using all available clusters, which may produce a misrepresentation of the predicted result, it only considers the clusters that are within the same threshold as the newest cluster.

In this case an extra setting parameter can be tuned. In addition to γ and the clustering threshold, the threshold that the past clusters should meet has to be set (Chapter 6 Section 6.2.3.). With the purpose of facilitating facilitate the comparison between the implementation of the GRTP method when using stable and unstable jobs, the tuneable parameters were set the same in both cases (γ is set to 0.8 and the threshold is set to 3 minutes, and the threshold for past clusters was also set to 3 minutes).

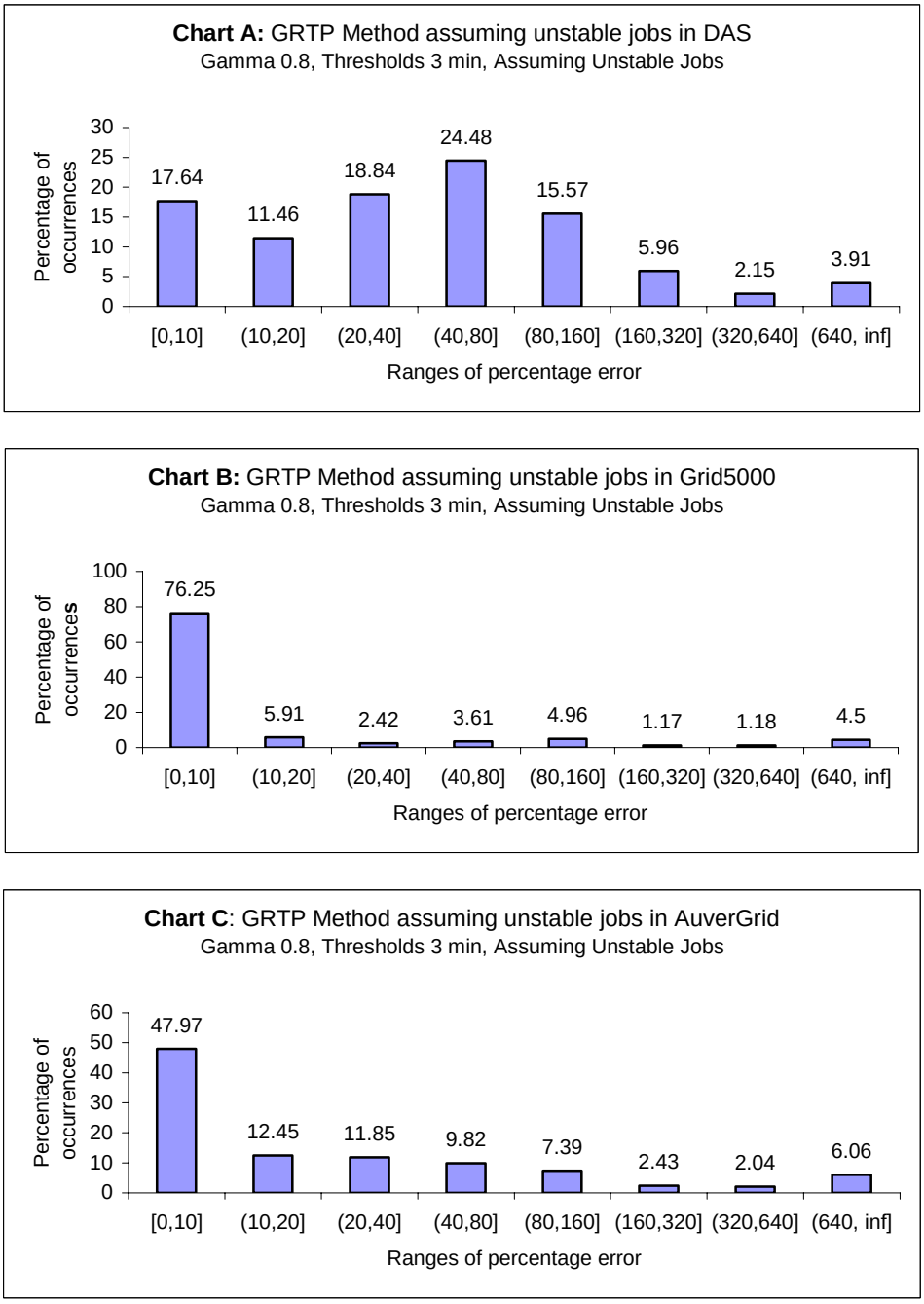


Figure 45 GRTP method assuming unstable jobs (Gamma 0.8, both thresholds 3min) in DAS (Chart A), Grid5000 (Chart B) and AuverGrid (Chart C)

Figure 45 displays the results of the analysis of the GRTR method in the three testbeds assuming all jobs are unstable. The results show that response time predictions have been markedly improved for both Grid5000 and AuverGrid. In Grid5000, the GRTR method is able to predict 76 percent of the cases within a 10 percent margin of error. Additionally, almost 50 percent of the cases are predicted within the same error gap in AuverGrid. This result is a remarkable achievement in its own right, but even more so when compared to other solutions, such as those of Figure 37 and Figure 38.

The GRTR method shows an improvement in response time prediction over previous approaches in DAS too, but it is not as remarkable as the improvements seen in Grid5000 and AuverGrid. Detailed data mining was performed in order to understand where the errors were generated. The data mining showed that DAS workload traces contain a significant number of jobs that exhibit inexplicable variances in their historical response time. For these jobs, the historical workload trace exhibits a significant change in the response time while the rest of the fields retain the same values. The reason for this may be known for the user that has submitted the job but it is not available via the workload trace. A typical example of this situation is when the parameters of the jobs are changes but not published. As a consequence, the response time prediction of the GRTR method is not as good in DAS as in the other two Grid environments used in this experiment.

The differences between the GRTP method and Smith's method are displayed in three one-to-one evaluations in Figure 46, Figure 47, and Figure 48. Comparing all three pairs of charts one sees a common trend; when assuming all jobs as unstable, the GRTP method in most cases offer a better response time prediction compared to other approaches.

The three result sets share another common trend; the response time of the jobs that have been improved by the GRTP method fit in greater ranges than the ones that were better predicted by Smith's method. For example, half of the 21 percent that Smith predicts better than the GRTP method in Grid5000, falls into the column of a 10 percent of improvement. In other words, the improvement achieved by Smith is less significant than the improvement achieved by the GRTP method.

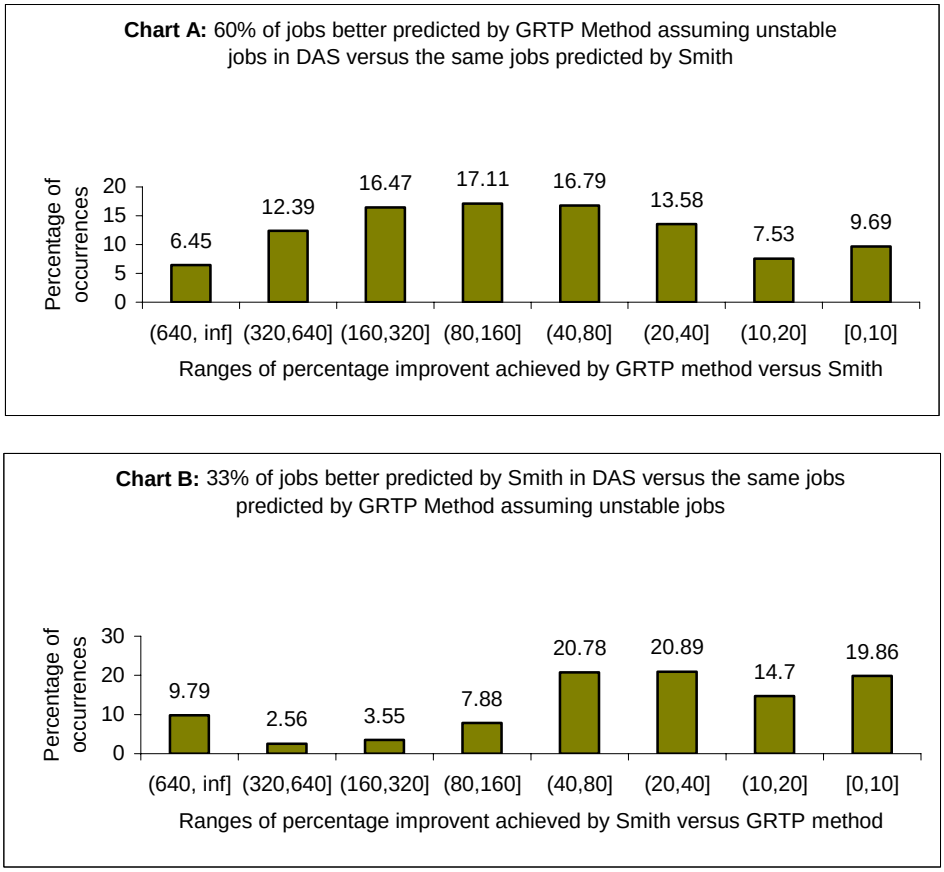
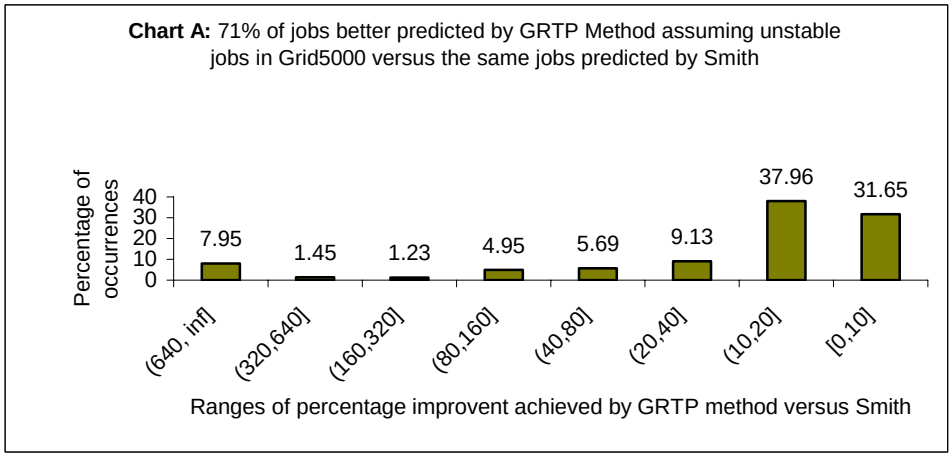


Figure 46 GRTP method assuming unstable jobs (Gamma 0.8, thresholds 3min) versus Smith method in DAS



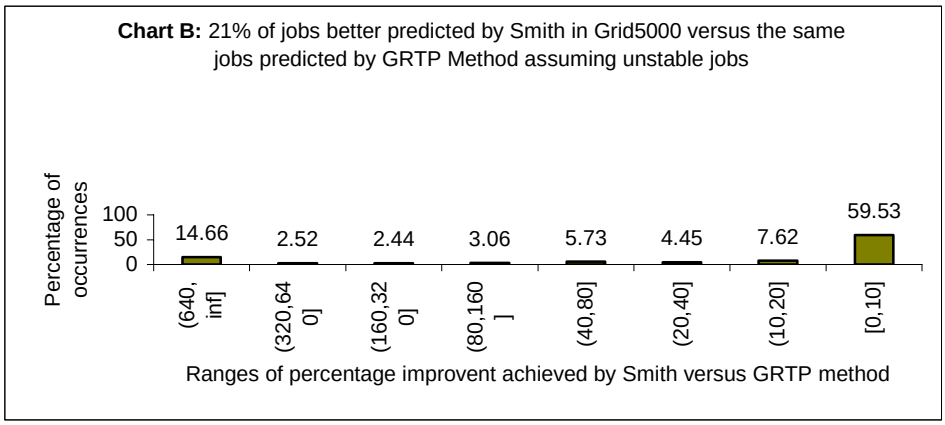


Figure 47 GRTP method assuming unstable jobs (Gamma 0.8, thresholds 3min) versus Smith method in Grid5000

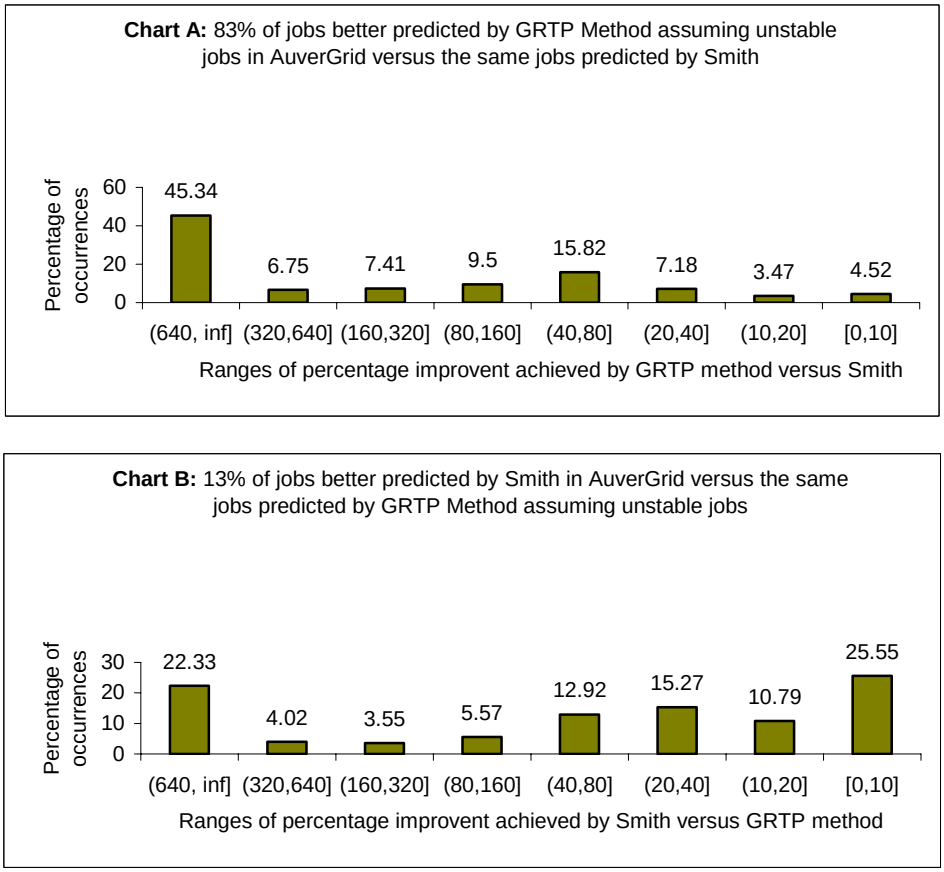


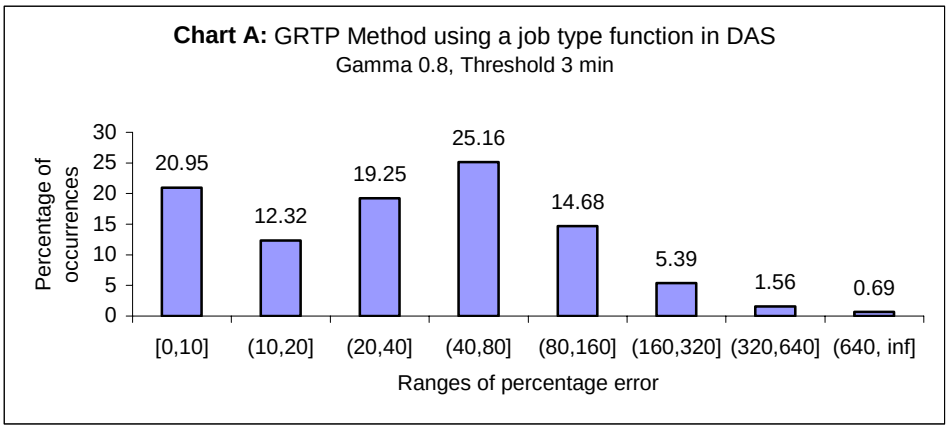
Figure 48 GRTP method assuming unstable jobs (Gamma 0.8, thresholds 3min) versus Smith method in AuverGrid

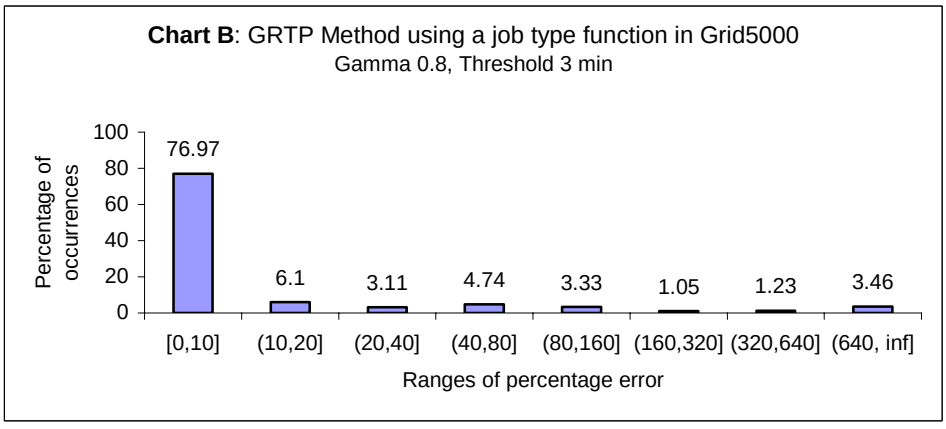
7.5. GRTP Method Evaluation Using a Job Type Method Function

This experiment was conducted by implementing the function that identifies the type of iWHC. This function, introduced in Chapter 6 Section 6.2.1., decides which prediction function the GRTP method should used based on the iWHC classification.

The results of this evaluation can be seen in Figure 49 and the response time prediction for DAS can be viewed in Chart A. The results show that the GRTP method is better at predicting the job response time than Smith's method (Figure 37). The median error for the GRTR method is 37 percent, compared to 86 percent in Smith's case.

The performance of the GRTP method in Grid5000 (Figure 49 Chart B) offered as a result that 76 percent of the cases have a prediction error smaller than 10 percent. Thus, the performance is substantially improved by the GRTR method compared to Smith's method (Figure 38 Chart A). The prediction performance was improved compared to the two previous experiments, especially in the exercise when all jobs were considered stable (Figure 41). The reason for the improvement is that the job's response time varies frequently in production Grid environments, and these changes are not published in the workload traces.





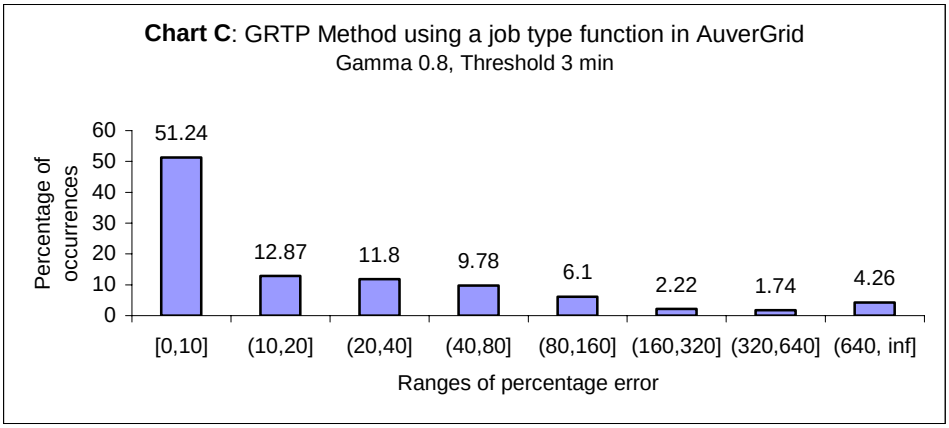


Figure 49 GRTP method using a job type function (Gamma 0.8, thresholds 3min) in DAS (Chart A), Grid5000 (Chart B) and AuverGrid (Chart C)

Finally, AuverGrid also presents a good quality response time prediction with 64 percent of the cases predicted with errors smaller than 20 percent. The results can be emphasised by comparing them to Smith's results in Figure 38 Chart B, where the prediction errors are practically impossible to handle in production Grid environments (38 percent of Smith predictions have an error greater than 650 percent). This is a key point as the GRTP method is able to perform well in Grid environments where other methods fail to achieve good results.

The previous performance improvements are confirmed by analysing the one-to-one evaluations in Figure 50, Figure 51, and Figure 52. The common trend in all the compared charts is that the response time of the jobs that have been improved by the GRTP method are mostly found in error ranges that are greater than those better predicted by Smith.

The GRTP method using a job type function offers better results than those predicted for GRTP considering the jobs as unstable (Figure 49 vs. Figure 45). This latest improvement is not as significant as what it was observed when comparing GRTP considering the jobs as unstable against GRTP considering the jobs as stable (Figure 45 vs. Figure 43,Figure 41, Figure 39).The reason is due to the natural variability of the workload trace data presented in the different Grid testbed. In these testbeds, most of the jobs are classified as unstables during its lifetime.

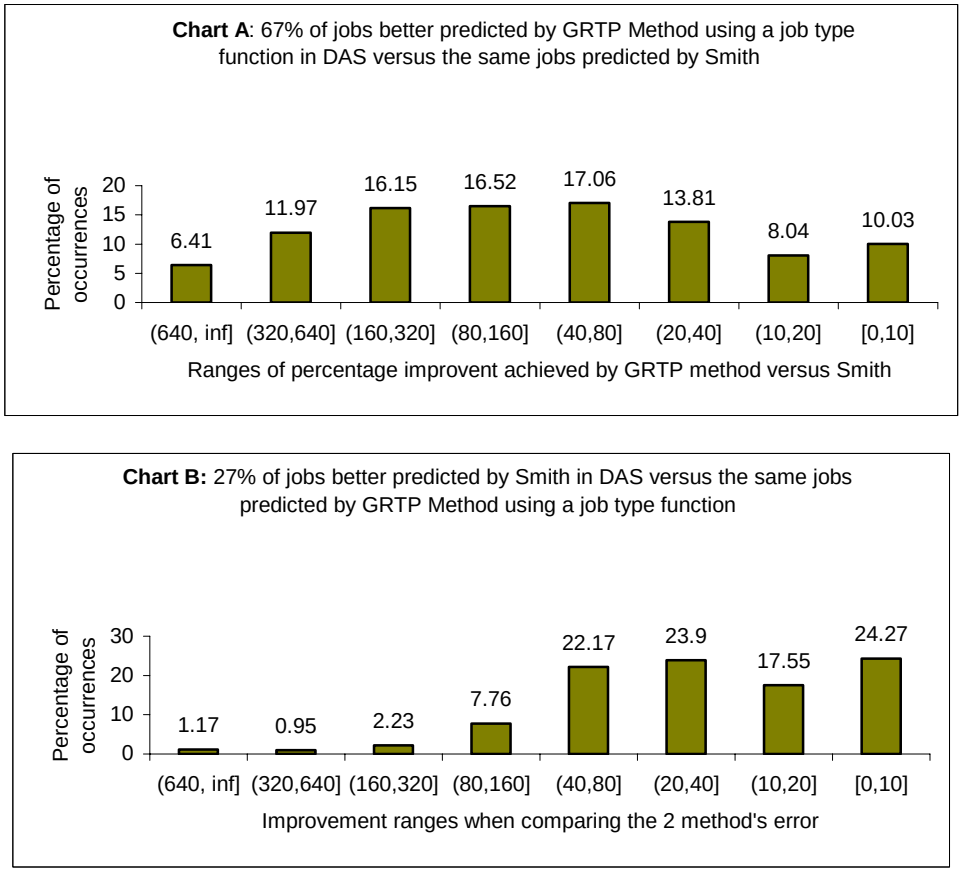
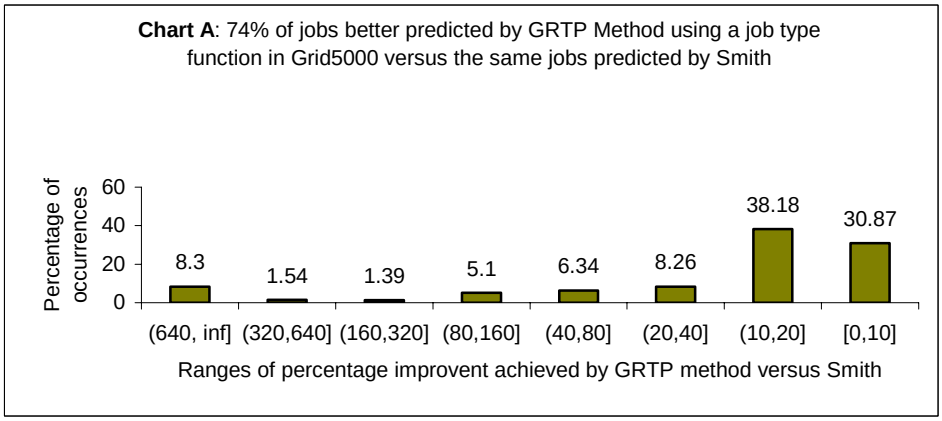


Figure 50 GRTP method using a job type function (Gamma 0.8, thresholds 3min) versus Smith method in DAS



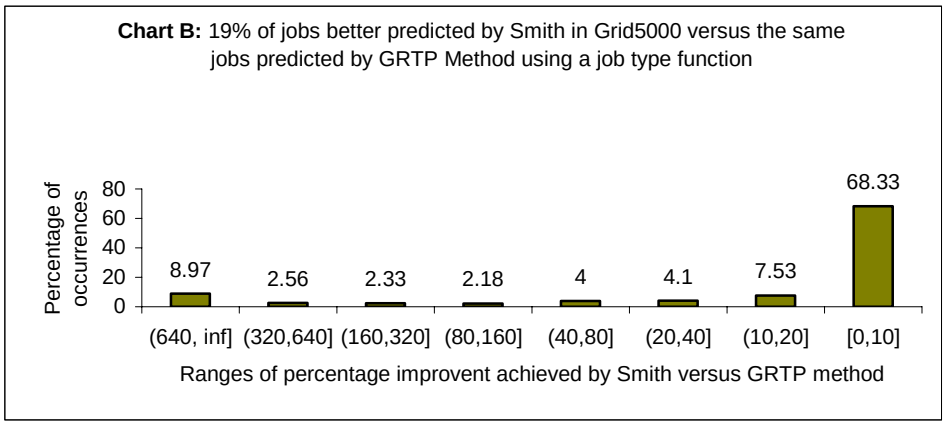


Figure 51 GRTP method using a job type function (Gamma 0.8, thresholds 3min) versus Smith method in Grid5000

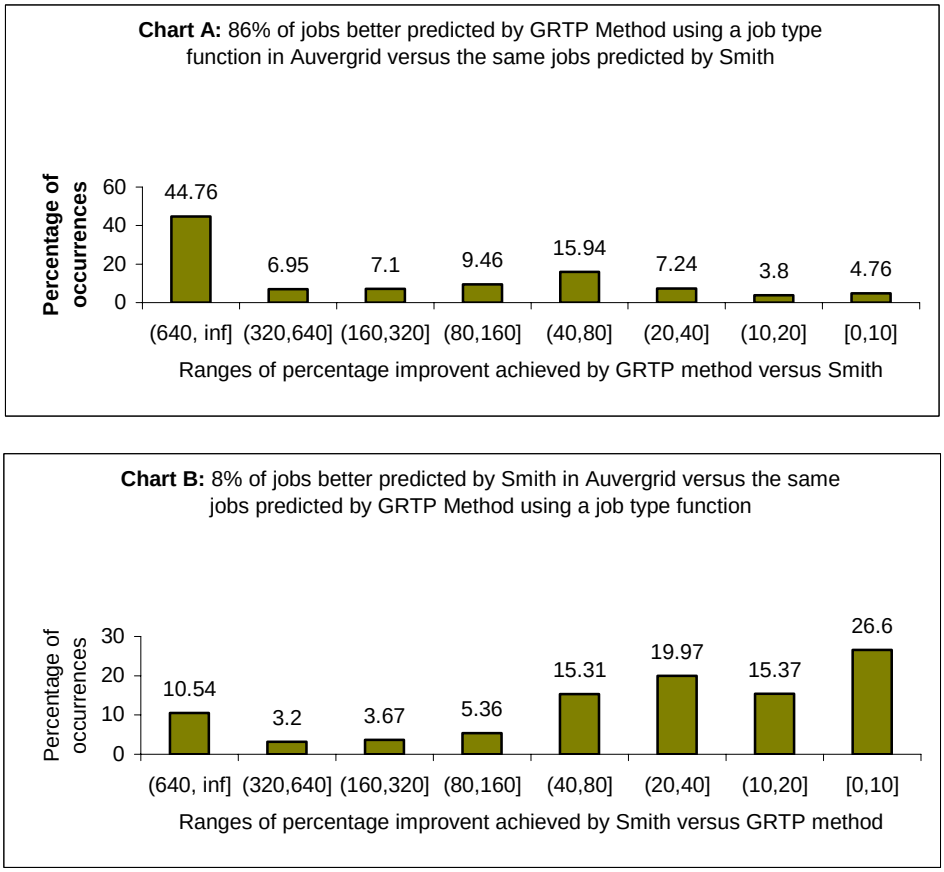


Figure 52 GRTP method using a job type function (Gamma 0.8, thresholds 3min) versus Smith method in AuverGrid

7.6. GRTP Method Evaluation Using Different Setting Parameters

The previous section presented the prediction results of the GRTP method when the job type function was used. The results suggested that the GRTP method is able to produce accurate prediction results in production Grid environments.

A pending point expressed in the previous experiments was to evaluate the GRTP method using different setting parameters. The parameters are γ, which is used in the discounted accumulative reward function, the threshold of the clusters in an iWHC, and the threshold of the historical clusters for unstable jobs.

For this experiment, γ was set to two different values: 0.8 and 0.3. In order to understand the reason behind this decision one has to understand the importance of the γ parameter;

by choosing a value for γ that is close to 1 one is able to make use of past experiences and deal better with outliers, and in choosing a value for γ that is closer to 0 one give more priority to the latest iWHC accuracy levels. The thresholds of the clusters and the historical clusters were set to 3 and 10 minutes: 3 minutes give a fine granularity of clusters but discard many historical clusters in the prediction function for unstable job types; 10 minutes group historical records with internal response time changes, but include more historical clusters in the prediction for unstable type of jobs.

The different results from the evaluation of the GRTP method using different setting parameters are displayed in Table 30, where an interesting result can be seen; the prediction accuracy in all three Grid environments increases when the cluster thresholds are smaller regardless the value of γ.

	Ranges of percentage error							
Grid(γ, Thresholds)	[0,10]	(10,20]	(20,40]	(40,80]	(80,160]	(160,320]	(320,640]	(640, inf]
DAS (0.3, 3)	**21.68**	**12.09**	**18.32**	**24.32**	**14.49**	**5.88**	**1.97**	**1.25**
DAS (0.8, 3)	**20.95**	**12.32**	**19.25**	**25.16**	**14.68**	**5.39**	**1.56**	**0.69**
DAS (0.3,10)	16.02	9.84	16.78	23.33	18.20	10.51	3.77	1.55
DAS (0.8, 10)	15.33	10.30	17.65	23.96	18.25	10.24	3.30	0.96
Grid5000 (0.3, 3)	**77.36**	**6.03**	**3.00**	**4.36**	**3.15**	**1.11**	**1.22**	**3.76**
Grid5000 (0.8, 3)	**76.97**	**6.10**	**3.11**	**4.74**	**3.33**	**1.05**	**1.23**	**3.46**
Grid5000 (0.3, 10)	73.84	7.87	3.35	4.83	3.41	1.25	1.37	4.08
Grid5000 (0.8, 10)	73.80	7.81	3.47	5.19	3.48	1.19	1.29	3.77
AuverGrid (0.3, 3)	**51.96**	**12.65**	**11.82**	**9.93**	**5.66**	**2.23**	**1.50**	**4.24**
AuverGrid (0.8, 3)	**51.24**	**12.87**	**11.80**	**9.78**	**6.10**	**2.22**	**1.74**	**4.26**
AuverGrid (0.3, 10)	48.74	12.63	11.22	10.25	6.53	2.77	2.16	5.70
AuverGrid (0.8, 10)	48.15	13.01	11.15	10.37	6.97	2.74	2.20	5.40

Table 30 GRTP method evaluation using different setting parameters

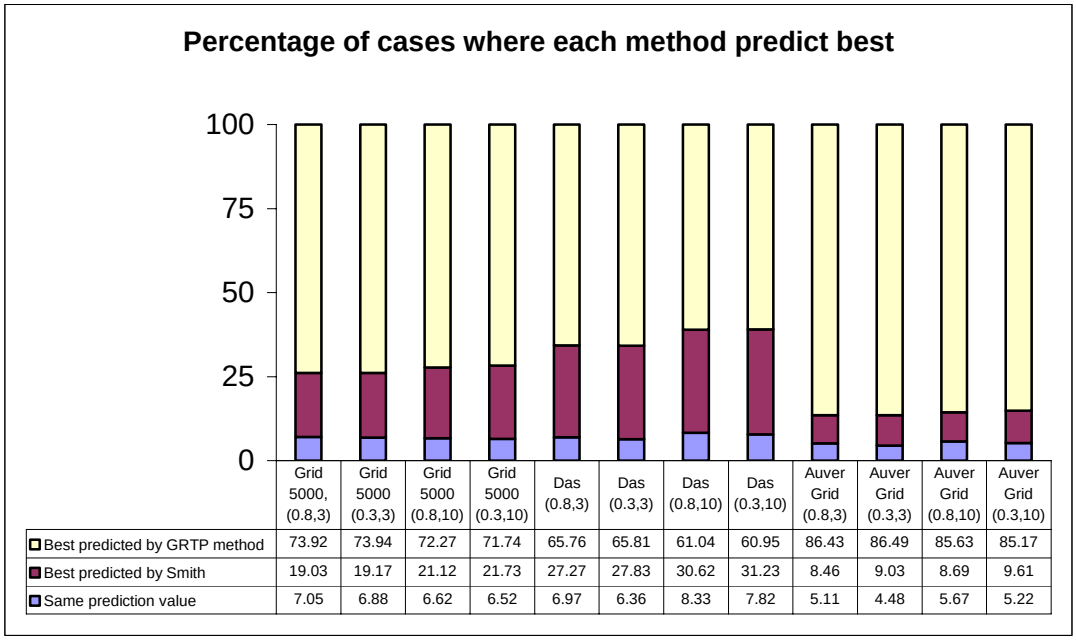
By analysing the same results, one can make a suggestion about the impact of γ on the GRTP method. The common tendency is that the smaller the Gamma values, the better the prediction accuracy. In other words, the response time trend given by the latest cluster should be given priority over old clusters. Furthermore, the results show that there is no need to consider the entire jobs response time history; the latest information should be enough to suggest a future response time.

The above two arguments can be combined to claim that the most accurate results are obtained when small gamma and threshold values are used. This affirmation is tested by noticing that the most accurate results in Table 30 (highlighted in bold) were obtained when Gamma was set to 0.3 and the threshold to 3.

But, the key conclusion from the results shown in Table 30 is that by reducing the thresholds and γ, the performance prediction accuracy improves significantly more in DAS rather than in the other two Grid environments. For instance, AuverGrid(0.8, 10) has 48 percentage of job submission within the [0,10] range of error while this is slightly improved by reducing the setting parameter (see AuverGrid(0.3, 3) with 52 percent of job submission). A similar trend can be seen in Grid5000. But DAS(0.8, 10) has 15 percent of job submissions in the [0,10] range of error while DAS(0.3, 3) has 22 percent. This is, in numbers, a 46 percentage increase in the number of job submissions included into the lower range of error.

The reason for the better improvement in DAS is found in the different workload trace characteristics that DAS contains if compared with GRID5000 or AuverGrid. DAS was represented by a workload trace that offered common variances of the response time for a given job. The causes of those response time variances were mainly not informed in any other data field of the historical data. This phenomenon was observed when the data was mined in detail. On the other hand, GRID5000 or AuverGrid had response time traces for a job which are informed in other data fields. This implied that the GRTP method was able to use this information to retrieve similar records from the past. For this reason, the use of bigger thresholds and γ close to 1 produced similar prediction results than those with smaller thresholds and γ close to 0.

Figure 53 displays which of the GRTP and Smith's methods generates the most accurate response time predictions. GRTP is using different Gamma and threshold setting parameters as specified above. The conclusion is that regardless of the values of the parameters used by the GRTP method, this method performs consistently better than Smith's.



	Grid 5000, (0.8,3)	Grid 5000 (0.3,3)	Grid 5000 (0.8,10)	Grid 5000 (0.3,10)	Das (0.8,3)	Das (0.3,3)	Das (0.8,10)	Das (0.3,10)	Auver Grid (0.8,3)	Auver Grid (0.3,3)	Auver Grid (0.8,10)	Auver Grid (0.3,10)
Best predicted by GRTP method	73.92	73.94	72.27	71.74	65.76	65.81	61.04	60.95	86.43	86.49	85.63	85.17
Best predicted by Smith	19.03	19.17	21.12	21.73	27.27	27.83	30.62	31.23	8.46	9.03	8.69	9.61
Same prediction value	7.05	6.88	6.62	6.52	6.97	6.36	8.33	7.82	5.11	4.48	5.67	5.22

Figure 53 Most accurate method utilisation share picture

7.7. Setting Parameters Sensitivity of the Method

The behaviour of the proposed method depends, in part, on: how the parameters that determine how much historical information is considered for selecting the right iWHC are set; the threshold for clustering the catalogues; and the threshold that the past clusters should meet to be considered by the prediction calculus.

One of the important aspects taken into account when designing the proposed GRTP method was to produce a solution that was not strongly dependent of its settings parameters. A method with high sensitivity to parameter settings invariably results in important changes in the method output if those settings are not well tuned.

High parameter sensitivity means that a slight change of the parameter value may affect the GRTP method performance considerably. Consequently, a solution with high parameter sensitivity is not robust enough to provide consistent performance across different Grid sites and over a long period of time.

Table 31 displays a combination of different γ and threshold values, which can be used to understand the sensitivity of the proposed method. This example uses a medium γ value (0.5) as well as the values used in the previous examples. The thresholds for each cluster and the past clusters were set to the same values as outlined in paragraph 7.5.

	Thresholds (Minutes)	γ	Error Median (Percentage)	Average GRTP Method Running Time (Miliseconds)
Grid5000	3	0.8	3.72	4109
		0.5	3.70	3438
		0.3	3.69	2566
	10	0.8	4.25	3844
		0.5	4.24	3875
		0.3	4.22	3949
DAS	3	0.8	37.10	4347
		0.5	37.15	3852
		0.3	37.34	3579
	10	0.8	49.76	4187
		0.5	50.62	4209
		0.3	51.00	4212
AuverGrid	3	0.8	9.36	4716
		0.5	9.25	3905
		0.3	9.00	3269
	10	0.8	11.12	3918
		0.5	11.08	4105
		0.3	10.79	4173

Table 31 Setting parameters sensitivity of the GRTP method

The median of the prediction error and the average GRTP method running time is shown together with possible combinations of γ and the values of the cluster thresholds. The results suggest the proposed GRTP method can be tuned to gain some response time prediction accuracy without affecting the quality of the prediction service. The common trend is to affirm that the smaller the thresholds the better the prediction accuracy. At the same time, reducing the thresholds size affects the running time of the method.

7.8. GRTP Method Analysis and Experiment Conclusions

The results of the experiments outlined in this chapter show that, in comparison to the most effective and accurate response time prediction method using workload traces (Smith [63]), the GRTP method improves the job response time prediction in different Grid environments.

The first experiment using the GRTP method demonstrated that the GRTP method, by assuming all jobs as stable, performs better than Smith in different testbeds. Additionally, and from the same results, it can be concluded that when Smith's method predicts better than GRTP the prediction improvement is not significant. Going a step further, the particular response time prediction cases when Smith's method predicts better can be explained via detailed data mining analysis. This analysis hints that every time that an occasional sudden change happens in a job response time trend the GRTP method needed to rebuild the knowledge database for this new scenario. In the experiment, because the GRTP method considered all jobs as stable, any historical data cluster would play a role in the prediction function. Therefore, it takes a number of job submissions to adjust the predictor to the new response time trend. When a response time change happens and Smith's method is not affected by non-normalised data fields, Smith is able to adapt more quickly than the GRTP method and produce better results. If after a sudden change, the later job submissions tend to be stable, Smith method may still predict better until the adjusting of historical clusters is completed by the GRTP method.

The second experiment testing the GRTP method assumed all jobs as unstable. The results produced a better prediction performance than GRTP assuming all jobs as stable, and also offered better results than Smith's method in different testbeds. The improvement is explained because the unstable prediction method discards clusters of data that are not following the latest job's response time trend. As a consequence, a better prediction was calculated

In particular, in the unstable experiment, the GRTP method was able to obtain better prediction results in Grid5000 and AuverGrid compared to DAS. The reason was that, even if there are many changes in the response time within an iWHC Grid5000 and AuverGrid, the response time keeps a steady line in between changes and therefore the use of the latest cluster helps in the prediction. On the other hand, DAS contains many jobs with sudden and inexplicable changes happening quite often.

The final suggestion from this experiment is that sudden changes in job response time are commonly found in current production Grid computing. This situation makes assumptions on taking all jobs as unstable more suitable for Grid computing prediction tasks. The reason behind this phenomenon can be found in the way that current users behave in production Grid environments. In many occasions the jobs are internally manipulated to perform different tasks. These changes are not published to the Grid middleware where the information found in the workload traces is stored. In other situations, the Grid site administrators are not publishing relevant information where the job was actually running. If the user and administrators conduct changes, the GRTP method will see an increasing number of jobs predicted using the stable type of job function.

The third GRTP method's experiment a job type function was added to select the most appropriate predictor for a stable or unstable job. The results of this experiment demonstrate that the prediction performance was improved compared to any of the previous experiments. For unstable jobs some of the historical data clusters are avoided in the prediction function. For stable jobs all the historical data clusters are considered. Looking at the Grid5000 Grid environment, the GRTP method offers significant performance for a data mining based prediction method, where 83 percent of the cases have a prediction error of less than 20 percent. In the AuverGrid environment, the GRTP method exhibited a good prediction performance with 64 percent of the cases predicted with errors of less than 20 percent; a result that becomes more relevant by comparing it against Smith's predictions.

The only situation that the GRTP method is not predicting well is when an unexpected change in response time trend takes place. The GRTP method is able to understand this situation and adapt to the new scenario. But, until this happens, the response time prediction is not accurate.

The final conclusion of this chapter is drawn from the experiment outlined in section 7.7, which tested the performance of the GRTP method when using different values for the gamma and threshold parameters. The outcome of this experiment indicates that the proposed method is not sensitive to the different settings of the parameters. The GRTP method can be tuned to achieve better response time prediction accuracy without affecting the quality of the prediction service. The common trend says that the smaller the clustering thresholds the better the prediction accuracy. At the same time, reducing the thresholds size affects the running time of the method.

Chapter 8 – Research Conclusions

This objective of this research is to predict job response time in production Grid environments. The objective was selected with the aim of improving the Grid resource usability and the administration of Grid environments.

Having discarded the traditional distributed computing solutions that take advantage of homogeneous and open environments, a non-invasive approach was presented that uses a Grid environment data workload traces generated from each Grid site as its source of input data.

The major challenge and the most significant restriction faced in this approach is the lack of control with regards to the quantity, type, and quality of the input data. The fact is that many requirements that a prediction method would like to demand cannot be always translated to the production Grid computing sites. Grid computing sites are, by definition, loosely coupled, heterogeneous, and geographically dispersed without any central management. The data traces generated by them are heterogeneous and dynamic.

The proposed response time prediction method was realised in a solution called GRTP. The GRTP method implemented several internal stages where the input data was data mined and prepared for the prediction functions.

The first GRTP internal stage allowed the inclusion of any data fields produced by the Grid sites. The GRTP uses an internal workload format that relaxes the definition of other current formats. Instead of taking the traditional semantic view which restricts a data field's value to the field's name for a particular Grid instance, the GRTP allows the qualification of the data fields based on the quality of the information that they contain

Another GRTP internal stage analysed the data fields of the workload traces using information theory methods. The results demonstrated that the quality of information varies from Grid site to Grid site and from data field to data field. This leads to the conclusion that prediction models which use data fields in an ad hoc manner will not always perform well in production Grid environments. The information theory methods also suggested that there are pairs of data fields that are strongly tied together and that taking into account only the largest data field (in term of quality of information) is enough to get the same information as provided by both fields together. The outcome of the information theory analysis is a method able to select a relevant set of data fields and present them in a meta-model. This meta-model represents a structured hierarchy based on the quality of the information of the data fields.

The meta-model is used to create filters of historical data. These filters are classified via the use of a reinforcement learning method and a discounted accumulative reward function. The discounted accumulative reward function uses the historical accuracy level of each filter as a reward or a penalty. The proposed reinforce learning method can be adjusted to deal with aged data, giving more priority to the most recent accuracy levels.

The historical data retrieved by the filters is also analysed by GRTP method. This exercise discovered a multi-pattern response time phenomenon commonly found in production Grid computing environments. This multi-pattern phenomenon follows the response time changes whose cause is hidden or non-shown in the workload data logs. As a result, the GRTP method implements a further data analysis to expose those changes. The data analysis identifies clusters of stable data instead of finding a function that fits the response time line.

The proposed clustering algorithm, called Time Base Quality Threshold (TBQT), is focused on finding patterns in continuous data records. TBQT is designed to analyse large sets of data, and the cluster threshold can be changed to increase or decrease the granularity of the data analysis

The data clusters are used to classify the job as stable or unstable based on its historical behaviour. A function that predicts job's response time is proposed for each job type. The stable job function evaluates the clusters by their internal average and weight function. The prediction is composed of the weighted mean of all available clusters. The unstable prediction function tries to continue with the latest job trend because the method has encountered an unknown suggestion about the future tendency of the job, until the job gets back to a stable situation. Therefore, the method only considers the clusters that are within the same threshold of newest cluster, rather than using all available clusters, which may lead to an error in the response time prediction.

8.1. Experimental result conclusions

The GRTP method was tested using a cross-validation technique using a training set taken from the Grid environment DAS and two testing sets taken from AuverGrid and Grid5000 Grid environments.

Three consecutive tests assuming stable and unstable jobs as well as using a job type method to select the most appropriate prediction function were carried out. The tests offered a significant increase in prediction performance for data mining based methods applied in Grid computing environments. In Grid5000 the GRTP method answered 77 percent of job prediction requests with an error of less than 10 percent while in the same environment, the current state of

the art solution [63] was able to predict 32 percent of the cases within the same range of error. In AuverGrid the GRTP method predicted 51 percent of the job's response time with an error of less than 10 percent, while the current state of the art solution was able to predict only 7 percent of the cases within the same range of error. The worse performance for the GRTP method was in DAS predicting 21 percent of the jobs with an error of less than 10 percent. But, the most effective and accurate response time prediction method using workload traces (Smith [63]) was only able to predict 8 percent of the cases within the same range of error.

One of the most important aspects taken into account when designing the GRTP method was producing a solution that would not be sensitive to its setting parameters. High parameter sensitivity means that a slight change may affect the method prediction performance. The proposed method proved to be robust to provide consistent performance across different Grid sites and over a long period of time using a different combination of setting parameters.

In general, the GRTP method was able to handle unexpected changes in resources and services, which affect the job response time trends, and it was able to adapt to new scenarios. The number of submission that GRTP takes to adapt to the new scenario depended on several factors.

One factor is the clustering method threshold that sets the granularity of the data clusters. This value influences how many new scenarios the GRTP discovers. In the experimental results, a bigger clustering threshold produced that fewer data clusters were found, i.e., more data records was clustered together. Unfortunately, in some cases this situation put dissimilar records together in the same cluster. A cluster with dissimilar records produced errors in the response time prediction. On the other hand, smaller clustering thresholds discovered many more changes in each job response time trend. But discovering and taking into account any new small change lead to prediction errors when the new discovered scenario was not consistently followed by future job submissions.

Another parameter setting that influenced the adaptation of the GRTP method is the response time prediction threshold for an unstable iWHC. The bigger the threshold the more data clusters are considered together to predict an unstable iWHC. As concluded in the previous parameter setting, increasing the threshold produced that unrelated records were considered for a job response time prediction. Contrarily, if the threshold is smaller, no many historical records data clusters are considered for a response time prediction, missing the chance of using a bigger set of data that can provide more information.

Finally, the γ parameter used by the discounted accumulative reward function also influences the GRTP adaptation to new scenarios. When past rewards are given greater emphasis relative to the immediate reward (γ has a value close to 1), new and accurate iWHC

were not taking into account immediately. This is generally a good approach for Grid environments with fewer variances. If γ gets close to 0, only the immediate reward is considered and new and accurate iWHC can be selected quicker. But, a new and accurate iWHC can also become obsolete and inaccurate if another sudden change happens. Therefore, moving quickly to use a new iWHC produced errors in the response time predictions.

In general, the proposed GRTP method can be tuned to adapt to different situations. If the Grid environment is stable where the services are running without many variances and the resources are normally loaded or if Grid environment informs the reason of the job response time changes in a workload data fields that the GRTP method can user, then the use of bigger thresholds and close to 1 discounted accumulative reward value is recommended. As it was shown in the previous chapter, GRID5000 and AuverGrid were two different Grid environments that presented this characteristic. Bigger thresholds and close to 1 discounted accumulative reward settings reduce GRTP method running cost and make use of the entire historical data to predict a job response time cancelling any noise that may confuse the method.

But, if the Grid environment has many sudden changes (such as data noise or jobs that ended the submission with errors and therefore it had a response time smaller than usual) the thresholds should be reduced and the γ value moved towards the 0 value to create a responsive solution. As espoused in the previous chapter, DAS was a Grid environment that presented these characteristics. The downside of using this kind of setting parameters is that the GRTP method can quickly start following a new response time tendency that is not in continued in future submissions. This situation produces errors in the predicted results.

The final conclusion from the experimental results is that the proposed GRTP method is capable of predicting job response time requests and it also improves the prediction quality when compared to other current solutions.

8.2. Future work

The GRTP method uses a combination of data mining techniques to predict a job response time. Because the proposed method is modular, any section can be extended or changed. Out of many future tasks that can implemented based on this research, it would be important to mention the following:

- The proposed Workload Meta-Model method successfully selects the input data fields that are providing relevant information for the predictor. The method used the joint entropy function to decide the fields to be kept or discarded. This method can be extended to create

a more elaborated Meta-Model that also includes the information dependency among data fields. The extended Meta-Model can be used to qualify the Grid environment and retrieve further information for the prediction methods.

- Another interesting idea is to produce Workload Meta-Models for lower granularity level. The proposed method presents a dynamic model at the level of the Grid site. But, lowering the models to the level of the jobs or resources may open many other alternatives to understand the historical data and how relevant the information is for each job or resource.
- The GRTP method has four setting parameters: γ for the accumulative reward function for the iWHC selection; the threshold for a field to be included into the workload meta-model; the GRTP method clustering method threshold that provides the granularity of the data clusters; and the response time prediction threshold for an unstable iWHC. Those four parameters that can be tuned to adjust the predicted results. The proposed method has been evaluated with different constant setting parameters. The on-the-fly adjusting of those settings depending on the Grid environment is a pending task that can be done automatically. This activity can open a new alternative for auto adjustable response time prediction solution.
- Finally, this method can be used not only to predict a job response time, but also to analyse the type and quality of information that a production Grid environment is producing. It was demonstrated in this research that many data fields in a workload trace are providing little information and that many jobs have unstable behaviours. There reasons are the Grid administrators that are not configuring the middlewares to log in a trace file the information in each grid site, and also the Grid users that change the internal settings or parameters of the jobs without publishing the information. A solution based on the GRTP method can be used to change and monitor the conduct of Gird administrator and users in a production Grid infrastructure like the NGS. The final result is to obtain a meaningful workload trace file that can be used as input of many supporting solutions for the Grid infrastructure.

References

[1] “Enabling Grids for E-sciencE (EGEE) Project” http://www.eu-egee.org/, 20-Nov-2009.

[2] “The Globus Alliance Project” http://www.globus.org/, 20-Nov-2009.

[3] “China National Grid (CNGrid) Project” http://i.cs.hku.hk/~clwang/grid/CNGrid.html, 20-Nov-2009.

[4] “The National Grid Service (NGS) Project”, http://www.grid-support.ac.uk/, 20-Nov-2009.

[5] “TeraGrid” http://www.teragrid.org, 20-Nov-2009.

[6] “SAS Grid Computing Project” http://www.sas.com/technologies/architecture/grid/, 20-Nov-2009.

[7] “Oracle Grid Computing Project” http://www.oracle.com/technologies/grid/index.html, 20-Nov-2009.

[8] “IBM® Grid computing Project” http://www-03.ibm.com/grid/, 20-Nov-2009.

[9] “Sun® Grid Engine Project” http://www.sun.com/software/sge/, 20-Nov-2009.

[10] Foster I: “Globus Toolkit Version 4: Software for Service-Oriented Systems” IFIP International Conference on Network and Parallel Computing, Springer-Verlag volume 3779, pages 2-13, 2006.

[11] “TORQUE Resource Manager”, http://www.clusterresources.com/products/torque-resource-manager.php, 20-Nov-2009.

[12] Anderson DP “Boinc: A system for public-resource computing and storage” 5th IEEE/ACM International Workshop on Grid Computing, 2004

[13] Menasce DA, Casalicchio E “QoS in grid computing”, Internet Computing, IEEE, volume 8, issue 4, pages 85-87, 2004.

[14] Malloy BA “Trace-driven and program-driven simulation: a comparison” Modelling, Analysis, and Simulation of Computer and Telecommunication Systems, MASCOTS '94, pages 395-396, 1994.

[15] Meyer BH. Pieper JJ. Paul JM “Power-Performance Simulation and Design Strategies for Single-Chip Heterogeneous Multiprocessors”, IEEE Transactions On Computers, volume 54, number 6, pages 684-697, 2005.

[16] Cassidy A.S., Paul J.M., Thomas D.E., “Layered, Multi-Threaded, High-Level Performance Design,” Procedures Design, Automation and Test in Europe Conference and Exhibition, pages 954-959, 2003.

[17] Poulsen DK, Yew PC “Execution-driven tools for parallel simulation of parallel architectures and applications”, Supercomputing '93. Proceedings, pages 860-869, 1993.

[18] Kerbyson DJ, Harper JS, Craig A, Nudd GR “PACE: A Toolset to Investigate and Predict Performance in Parallel Systems”. In European Parallel Tools Meeting, ONERA, Paris, 1996.

[19] “Grid Physic Network (GriPhyN) Project”, http://www.griphyn.org/, 2009.

[20] Jain RK “The art of Computer System Performance Analysis. Techniques for experimental design, measurement, simulation and modelling”. John Wiley & Sons, Inc. ISBN: 978-0-471-50336-1, 1991.

[21] Xiong H, Pandey G, Steinbach M, Kumar V “Enhancing Data Analysis with Noise Removal”, IEEE Transactions On Knowledge And Data Engineering, volume 18, number 3, 2006.

[22] Feitelson G “Workload Modeling for Computer Systems Performance Evaluation”, School of Computer Science and Engineering, The Hebrew University of Jerusalem, 91904 Jerusalem, Israel, http://www.cs.huji.ac.il/~feit/wlmod/wlmod.pdf, 20-Nov-2009

[23] Feitelson G “Packing schemes for gang scheduling”. Scheduling Strategies for Parallel Processing, Lectures Notes Computing Science, volume 1162, pages 89-110, 1996.

[24] Feitelson G, Nitzberg B “Job characteristics of a production parallel scientific workload on the nasa ames ipsc/860”. In Job Scheduling Strategies for Parallel Processing, Lecture notes in computing science, volume 949, pages 337-360, 1995.

[25] Tsafrir D, Feitelson G “Instability in parallel job scheduling simulation: the role of workload flurries”. Parallel and Distributed Processing Symposium, IPDPS. 20th International, 2006.

[26] Sunderam VS “A Framework for Parallel Distributed Computing” Journal of Concurrency: Practice and Experience, pages 315-339, 1990.

[27] “MPI A Message Passing Interface Standard”, Message Passing Interface Forum http://www.mpi-forum.org, 20-Nov-2009

[28] Foster I; Kesselman C “The Grid: Blueprint for a new computing infrastructure”, Chapter 2, Morgan Kaufman, ISBN: 1-558660-475-8, 1999

[29] Kreger H, “Web Services: Conceptual Architecture” (WSCA 1.0) By IBM Software Group, May, 2001.

[30] “Web Services Description Language (WSDL) Project”: http://www.w3.org/TR/wsdl, 20-Nov-2009.

[31] “W3C Simple Object Access Protocol (SOAP) Project”, http://www.w3.org/TR/soap/, 20-Nov-2009.

[32] “Open Grid Services Architecture Project” http://forge.gridforum.org/projects/ogsa-wg, 20-Nov-2009.

[33] Gruber R, Volgers P, De Vita A, Stengel M, Tran T “Parameterisation to tailor commodity clusters to applications” Future Generation Computing Systems. 19(1): pages 111-120, 2003

[34] Gruber R, Keller V, Kuonen P, Sawley MC, Schaeli B, Tolou A, Torruella M, Tran T “Towards an Intelligent Grid Scheduling System” PPAM, pages 751-757, 2005.

[35] Keller V, Cristiano K, Gruber R, Puonen P, Maffioletti S, “Integration of ISS into the VIOLA Meta-scheduling Environment”, Integrated Research in Grid Computing, Springer, CoreGRID Integration Workshop, Pisa, 2005

[36] Gruber R, Keller V “ISS concept” CoreGRID Integration Workshop, Cracow, 2006

[37] Steffenel LA “Modelling Network Contention Effects on AlltoAll Operations” IEEE Conference on Cluster Computing (CLUSTER 2006). Barcelona, Spain, 2006

[38] Barchet-Steffenel LA, Mounié G "Scheduling Heuristics for Efficient Broadcast Operations on Grid Environments". International Workshop on Performance Modelling, Evaluation, and Optimisation of Parallel and Distributed Systems (PMEO-PDS'06), in conjunction with IPDPS'06. Rhodes Island, Greece, 2006.

[39] Steffenel LA ”LaPIe - Communications Collectives Adaptées aux Grilles de Calcul”, PhD Thesis, INPG, Grenoble, France. 2005.

[40] Barchet-Steffenel LA, Mounié G "Total Exchange Performance Modelling under Network Contention". In: Proceedings of the 6th International Conference on Parallel Processing and Applied Mathematics, volume 3911, Springer-Verlag. Poznan, Poland. 2005.

[41] Barchet-Estefanel LA, Mounié G "Prédiction de Performances pour les Communications Collectives". In: Proceedings of the 16ème Rencontre Francophone du Parallélisme (RenPar'16), Le Croisic, France, pages 101-112, 2005.

[42] Barchet-Estefanel LA, Mounié G "Performance Characterisation of Intra-Cluster Collective Communications". In: Proceedings of the SBAC-PAD 2004 16th Symposium on Computer Architecture and High Performance Computing, Foz-do-Iguacu, Brazil, IEEE Press, pages 254-261, 2004.

[43] Barchet-Estefanel LA, Mounié G "Identifying Logical Homogeneous Clusters for Efficient Wide-area Communications". In: Proceedings of the EuroPVM/MPI 2004 11th European PVM/MPI Users Group Meeting, Budapest, Hungary, volume 3241, Springer-Verlag, pages 319-326, 2004.

[44] Barchet-Estefanel LA, Mounié G "Fast Tuning of Intra-Cluster Collective Communications". In: Proceedings of the EuroPVM/MPI 2004 11th European PVM/MPI Users Group Meeting (2004), Budapest, Hungary, Springer-Verlag , volume 3241, pages 28-35, 2004.

[45] Badia RM, Escale F, Gabriel E, Gimenez J, Keller R, Labarta J, Müller MS, “Performance Prediction in a Grid Environment”, 1st European Across Grids Conference, Santiago de Compostela, 2003

[46] Badia RM, Labarta J, Gimenez J, Escale F “DIMEMAS: Predicting MPI applications behaviour in Grid environments”, Workshop on Grid Applications and Programming Tools (GGF8), 2003.

[47] Song B, Ernemann C, Yahyapour R "Parallel Computer Workload Modeling with Markov Chains", Job Scheduling Strategies for Parallel Processing, ISBN 978-3-540-25330-3, pages 47-62, 2005.

[48] Song B, Ernemann C, Yahyapour R "User Group-based Workload Analysis and Modelling", In: IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2005), IEEE Press, volume 2, pages 953-961, 2005.

[49] Song B, Ernemann C, Yahyapour R, "Parallel Computer Workload Modelling with Markov Chains in job Scheduling Strategies for Parallel Processing”: 10th International Workshop, JSSP New York, USA, pages 47-62, Springer, 2005.

[50] Song B, Ernemann C, Yahyapour R, “Modelling of Parameters in Supercomputer Workloads", in Proceedings of the Workshop of Parallel Systems and Algorithms (PASA), in conjunction with ARCS 2004: Organic and Pervasive Computing, Augsburg; pages 400-409; LNI 41 GI, 2004.

[51] Nadeem F, Yousaf FF, Prodan R, Fahringer T “Soft Benchmarks-based Application Performance Prediction using a Minimum Training Set”, Second International Conference on e-Science and Grid computing, Amsterdam, Netherlands, 2006.

[52] Wieczorek M, Prodan R, Fahringer T, “Comparison of Workflow Scheduling Strategies on the Grid”, Springer-Verlag, LNCS, Proceedings of PPAM05 Conference, Poznan, Poland, 2005

[53] Downey AB “Using queue time predictions for processor allocation”. 3rd Workshop on job Scheduling Strategies for Parallel Processing, Lecture Notes in Computer Science; volume 1291, pages 35-57, 1997.

[54] Downey AB. “A parallel workload model and its implications for processor allocation” 6th Intl. Symposium, High Performance Distributed Computing, 1997.

[55] Dinda PA, O'Hallaron DR "Host load prediction using linear models". Cluster Computing, volume 3, number 4, pages 265-280, 2000.

[56] Dinda PA, Kallivokas LF, Lowekamp B, O'Hallaron DR "The Case for Prediction-Based Best-Effort Real-Time Systems" IPPS/SPDP Workshops, pages 309-318, 1999.

[57] Dinda PA, O'Hallaron DR "An extensible toolkit for resource prediction". In Distributed Systems, Technical Report CMU-CS-99-138, School of Computer Science, Carnegie Mellon University, 1999.

[58] Dinda PA "The statistical properties of host load" Scientific Programming, volume 7 , issue 3-4, pages: 211 - 229, ISSN:1058-9244, 1999

[59] Dinda PA, Lowekamp B, Kallivokas L, O'Hallaron DR "The case for prediction-based best-effort real-time systems". Proceedings of the 11 IPPS/SPDP'99 Workshops Held in Conjunction with the 13th International Parallel Processing Symposium and 10th, Lecture Notes In Computer Science; volume 1586, pages309 - 318, 1999.

[60] Dinda PA "Online prediction of the running time of tasks". Cluster Computing SIGMETRICS/Performance, pages 225-236, 2002.

[61] Gibbons R. "A historical application profiler for use by parallel schedulers", Job Scheduling Strategies for Parallel Processing, ISBN 978-3-540-63574-1, pages 58-77, 1997.

[62] Smith W, Foster I, Taylor VE "Predicting Application Run Times Using historical information" Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing table of contents, volume 1459, pages 122 - 142, ISBN:3-540-64825-9, 1998.

[63] Smith W, Taylor VE, Foster I "Using run-time predictions to estimate queue wait times and improve scheduler performance". Proceedings of the job Scheduling Strategies for Parallel Processing, Lecture Notes in Computer Science; volume. 1659, pages 202-219, 1999.

[64] Hickey R J, Black M "Refined Time Stamps for Concept Drift Detection During Mining for Classification Rules", Temporal, Spatial, and Spatio-Temporal Data Mining, Lecture Notes in Computer Science, Volume 2007/2001, pages 20-30, 2001

[65] Quinlan J R "Induction of Decision Trees" Machine Learning. volume 1, issue 1, pages 81-106, ISSN:0885-6125, 1986.

[66] Rodero I, Guim F, Corbalán J, Labarta J "Enanos: Parallel Computing: Current & Future Issues of High-End Computing, Proceedings of the International Conference ParCo 2005, volume 33, pages 81-88, ISBN 3-00-017352-8, 2006.

[67] Guim F, Corbalan J, Labarta J "Analysing loadleveler historical information for performance prediction". Jornadas de Paralelismo, Barcelona, 2005.

[68] Guim F, Rodero I, Levillain O, Corbalán J, Labarta J “Monitoring Grid entities through the palantir meta-information system”. Technical report, 2005.

[69] Guim F, Corbalan J, Labarta J “Impact of qualitative and quantitative errors of the job runtime estimation in backfilling based scheduling policies”. Submitted to SIGMETRICS Performance Journal, 2006.

[70] Wolski R “Experiences with Predicting Resource Performance On-line in Computational Grid Settings”. ACM SIGMETRICS Performance Evaluation Review, volume 30, number 4, pages 41-49, 2003.

[71] Swany M, Wolski R “Multivariate resource performance forecasting in the network weather service” In Proceedings of SC02, Baltimore, MD, November, 2002.

[72] Wolski R, Spring N, Hayes J “The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing”. Journal of Future Generation Computing Systems, volume 15, pages 757-768, 1999.

[73] Wolski R “Dynamically Forecasting Network Performance Using the Network Weather Service”. Journal of Cluster Computing, volume 1, pages 119-132, 1998.

[74] Wolski R, Spring N, Peterson C “Implementing a Performance Forecasting System for Metacomputing: The Network Weather Service”. In Proceedings of SC97, November, 1997.

[75] Berman F, Wolski R, Figueira S, Schopf J, Shao G “Application level scheduling on distributed heterogeneous networks”. Proceedings of the 1996 ACM/IEEE conference on Supercomputing, article number 39, 1996.

[76] Goyeneche A, Terstyanszky G, Delaitre T, Winter S “Improving Grid computing performance prediction using weighted templates”, Conf. Proc. of the UK e-Science 2007 All Hands Meeting, Nottingham, 2007.

[77] “The UK National Grid Service”, www.ngs.ac.uk, 20-Nov-2009

[78] “The Grid Workloads Archive”, http://gwa.ewi.tudelft.nl/pmwiki/, 20-Nov-2009

[79] Feitelson G “Workload modelling for performance evaluation”. In Performance Evaluation of Complex Systems: Techniques and Tools, Lecture Notes Computing Science volume 2459, pages 114–141, 2002

[80] Iosup A, Li H, Jan M, Anoep S, Dumitrescu C, Wolters L, Epema D "The Grid Workloads Archive", Future Generation Computer Systems, volume 24, issue 7, pages 672-686, ISSN:0167-739X, 2008

[81] Codd E. F. "A relational model of data for large shared data banks", Communications of the ACM archive, Volume 13 , Issue 6, Pages: 377-387, 1970.

[82] Shannon C.E. “A Mathematical Theory of Communication” Bell System Technical Journal, volume. 27, pages 379-423, 623-656, July, October, 1948.

[83] Reza F.M. “An Introduction to Information Theory”, ISBN: 9780486682105, Dover Publications Inc, 1994

[84] Berger AL, Della Pietra SA, Della Pietra VJ “A Maximum Entropy Approach to Natural Language Processing”, Journal of Computational Linguistics, volume 22, pages 39-71, 1996

[85] Box GEP, Jenkins GM “Time series analysis: forecasting and control” Holden-Day Series. ISBN: 0816211043, Holden-Day, Incorporated, 1990

[86] Anderberg M “Cluster analysis for applications”, Academic Press, New York. 1973

[87] Hartigan J “Clustering Algorithms”. Wiley, New York, NY. 1975

[88] Jain AK, Dubes RC “Algorithms for Clustering Data”, Prentice Hall, 1988.

[89] Jardine N., Sibson R., “Mathematical Taxonomy”. Wiley, London. 1971

[90] Sneath PHA, Sokal RR “Numerical Taxonomy”. Freeman, San Francisco, CA. 1973

[91] Tryon RC, Bailey DE “Cluster Analysis”. McGraw-Hill, New York, NY. 1973

[92] Everitt BS, Landau S, Leese M “Cluster analysis”. Halsted Press, New York, 1993

[93] Kaufman L, Rousseeuw PJ. “Finding groups in data: an introduction to cluster analysis”. SBN: 978-0-471-73578-6, John Wiley & Sons, 2005.

[94] Jain AK, Murty MN, Flynn PJ “Data clustering: a review” ACM Computing Surveys, 31(3):264–323, 1999.

[95] Defays D “An Efficient Algorithm for a Complete Link Method”. The Computer Journal, volume 20, number 4, pp. 364-366, 1977.

[96] Day WH, Edelsbrunner H “Efficient Algorithms for Agglomerative Hierarchical Clustering Methods”. Journal of Classification, volume 1, pages 1-24, 1984.

[97] MacQueen J “Some methods for classification and analysis of multivariate observations”. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, volume I, pages 281-297. University of California Press, Berkeley and Los Angeles, CA.

[98] Zahn CT “Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters”, IEEE Transactions on Computers, volume C-20(1), pages 68-86, 1971

[99] Banfield JD, Raftery AE “Model-based gaussian and non-gaussian clustering”. Biometrics, volume 49, pages 803–821, 1993

[100] Celeux G, Govaert G “Gaussian parsimonious clustering models”. Pattern Recognition, pages 781–793, 1995.

[101] Cheng Y “Mean Shift, Mode Seeking, and Clustering” IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 17, number. 8, pages 790-799, doi:10.1109/34.400568, 1995

[102] Frigui H, Krishnapuram R "Clustering by competitive agglomeration". Journal in Pattern recognition, volume. 30, number 7, pages 1109-1119, ISSN 0031-3203, 1997

[103] Heyer LJ, et al. "Exploring expression data: identification and analysis of co-expressed Genes". Genome Res. volume 9, pages 1106–1115, 1999

[104] Young AN, Amin MB, Moreno CS, Lim SD, Cohen C, Petros JA, Marshall FF, Neish AS, "Expression profiling of renal epithelial neoplasms: a method for tumor classification and discovery of diagnostic molecular markers", American Journal of Pathology, pages 1639-1651. 2001

[105] Goyeneche A, Terstyanszky G, Winter S "An approach to managing Quality of Services and Performance in Grid Computing", MicroCAD 2004 International Scientific Conference, Miskolc, Hungary, 2004

[106] Gerald S, Rajkumar K, Arun R, Ponnuswamy R. "Scheduling of parallel jobs in a heterogeneous multi-site environment". JSSPP, Lecture notes in computer science ISSN 0302-9743, volume 2862, pages 87-104 2003.

[107] Ernemann C, Hamscher V, Yahyapour R "Benefits of global Grid computing for jobscheduling" 5th IEEE/ACM International Workshop on Grid Computing, 2004.

[108] Pinchak C, Lu P, Goldenberg M "Practical heterogeneous placeholder scheduling in overlay meta-computers: Early experiences. job Scheduling Strategies for Parallel Processing", pages 205–228, Lecture Notes Computing Science vol. 2537, 2002

[109] Goyeneche A, Guim F, Rodero I, Terstyansky G, Corbalan J "Extracting performance hints for Grid users using data mining techniques: a case study in the NGS". The Mediterranean Journal of Computers and Networks SPECIAL ISSUE on Data Mining Applications on Supercomputing and Grid Environments, volume 3, number. 2, ISSN: 1744-2397, pages 52-61, 2007

[110] Mitchell TM "Machine Learning", McGraw-Hill Science Engineering, ISBN-10: 0-07-042807-7, 1997

[111] Kohavi R. "A study of cross-validation and bootstrap for accuracy estimation and model selection". Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, volume 2 (12), pages 1137–1143. Morgan Kaufmann, San Mateo, 1995.

Glossary of Acronyms

- ANL: Argone National Laboratory.
- CSG: Candidate Set Generator.
- CTC: Cornell Theory Center.
- DAS: Distributed ASCI Supercomputer.
- EPS: OGSA Execution Planning Service.
- GroupID: User group identification.
- GRTP: Grid Computing Response Time Prediction.
- GT: Globus Toolkit.
- GWF: Grid Workload Format.
- ISS: Grid Scheduling System.
- iWHC: Instanciated Workload Header Catalogue.
- JobID: Job Identification.
- JobStructure: Job Structure.
- LastRunSiteID: Last site in which the job run.
- MPI: Message Passing Interface.
- Nproc: Number of processors used by a job.
- NWS: Network Weather Service.
- OGSA: Open Grid Services Architecture.
- OrigSiteID: Site from which the job originated.
- PACE: Performance Analysis and Characterisation Environment.
- PartitionID: Partitions in the systems.
- ProjectID: Identification of the project of the job.
- PVM: Parallel Virtual Machine.
- QoS: Quality of Service.
- QT: Quality Threshold .
- QueueId: Job queue identification.
- ReqLocalDiskSpace: Job requirements for disk space.
- ReqNetwork: Job requirements for the network.

- ReqPlatform: Job requirements for the platform.
- ReqResources: Job requirements for other resources.
- SDSC: San Diego Supercomputer Centre.
- SLA: Service Level Agreement.
- Status: Status of the job.
- SWF: Standard Workload Format.
- TBQT: Time Based Quality Threshold.
- URI: Uniform Resource Identifier.
- UsedDiskSpace: Used disk space.
- UsedNetwork: Used network.
- UsedResources: Used resources.
- UserID: User identification.
- W3C: Word Wide Web Consortium.
- WHC: Workload Header Catalogue Definition.
- WSDL: Web Service Description Language.
- WSRF: Web Services Resources Framework.

Appendix A: Dissemination of the research findings

A.1. Journals

- A.Goyeneche, F. Guim, I. Rodero, G. Terstyansky, J. Corbalan, Extracting performance hints for Grid users using data mining techniques: a case study in the NGS, The Mediterranean Journal of Computers and Networks, SPECIAL ISSUE on Data Mining Applications on Supercomputing and Grid Environments, Volume 3, No. 2, April 2007, ISSN: 1744-2397, p 52-61
- T. Delaittre, T. Kiss, A. Goyeneche, G. Terstyanszky, S.Winter, P. Kacsuk, GEMLCA: Running Legacy Code Applications as Grid Services, Journal of Grid Computing Vol. 3. No. 1-2. June 2005, Springer Science + Business Media B.V., Formerly Kluwer Academic Publishers B.V. ISSN: 1570-7873, (Paper), pp 75-90, 1572-9814 (Online),http://dx.doi.org/10.1007/s10723-005-9002-8

A.2. Conferences

- A. Goyeneche, G. Terstyanszky, T. Delaitre, S. Winter, Improving Grid computing performance prediction using weighted templates, Conf. Proc. of the UK e-Science 2007 All Hands Meeting, Nottingham, September 10-13, 2007
- F. Guim, I. Rodero, A. Goyeneche, J. Corbalan, The Grid Backfilling: a Multi-Site Scheduling Architecture with Data Mining Prediction Techniques, Conf. Proc. of CoreGRID Workshop on Grid Middleware, Workshop in conjunction with ISC'07 conference, Dresden, June 25-26, 2007
- G. Sipos, A. Goyeneche, T. Kiss, P. Kacsuk, Executing Parameter, Study Workflows in the P-GRADE Portal, Conf. Proc. of the CoreGRID, Workshop on Grid Systems, Tools and Environments, December 1st, 2006, Sophia Antipolis, France, within the Framework of the GRIDs@Work week.
- Francesc Guim, Ariel Goyeneche , Julita Corbalan, Jesus Labarta1, Gabor Terstyansky, Grid computing performance prediction based in historical information., Proceedings of the CoreGRID Integration Workshop, Pisa, Italy, November 2005.

- A.Goyeneche, T.Kiss, G.Terstyanszky, G.Kecskemeti, T.Delaitre, P.Kacsuk, S.C. Winter, Experiences with Deploying Legacy Code Applications as Grid Services using GEMLCA, Conf. Proc. of the European Grid Conference, February 14 -16, 2005, Science Park Amsterdam, The Netherlands, Volume editors: P.M.A. Sloot, A.G. Hoekstra, T. Priol, A.Reinefeld, M. Bubak, pp 851-860, ISBN: 3-540-26918-5
- G.Terstyansky, T. Delaitre, A. Goyeneche, T, Kiss, K. Sajadah, S.C.Winter, P.Kacsuk, Security Mechanisms for Legacy Code Applications in GT3 Environment, Conf. Proc. of the 13th Euromicro Conference on Parallel, Distributed and Network-based Processing, Lugano, Switzerland, February 9-11, 2005
- P. Kacsuk, A. Goyeneche, T. Delaitre, T. Kiss, Z. Farkas, T.Boczko, High-level Grid Application Environment to Use Legacy Codes as OGSA Grid Services, Conf. Proc. of the 5th IEEE/ACM International Workshop on Grid Computing, pp. 428-435, November 8, 2004, Pittsburgh, USA
- T. Delaitre, A.Goyeneche, T.Kiss, G.Z. Terstyanszky, N.Weingarten, P. Maselino, A. Gourgoulis, S.C. Winter, Traffic Simulation in P-Grade as a Grid Service, Conf. Proc. of the DAPSYS 2004 Conference, pp 129-136, ISBN 0-387-23094-7, September 19-22, 2004, Budapest, Hungary
- T. Delaitre, A.Goyeneche, T.Kiss, G.Z. Terstyanszky, S.C. Winter, P. Kacsuk, D. Igbe, P. Maselino, K. Sajadah, N. Weingarten, Experiences with Publishing and Executing Parallel Legacy Code using an OGSI Grid Service, Conf. Proc. of the UK E-Science All Hands Meeting, pp. 999-1002, ISBN 1-904425-21, 31st August - 3rd September 2004, Nottingham, UK
- T. Delaitre, A.Goyeneche, P. Kacsuk, T.Kiss, G.Z. Terstyanszky, S.C. Winter, GEMLCA: Grid Execution Management for Legacy Code Architecture Design, Conf. Proc. of the 30th EUROMICRO Conference, Special Session on Advances in Web Computing, pp. 477-483, ISBN 0-7695-2199-1, August 31st - September 3rd 2004, Rennes, France.
- T. Delaitre, A.Goyeneche, T.Kiss, S.C. Winter, Publishing and Executing Parallel Legacy Code using an OGSI Grid Service, Conf. Proc. of the 2004 International Conference on Computational Science and its Applications, Technical Session on Grid Computing, ISBN 3-540-22056-9, pp 30-36, May 2004, Assisi, Italy.
- Ariel Goyeneche, Dr Gabor Terstyanszky, Prof Stephen Winter, An approach to managing Quality of Services and Performance in Grid Computing, MicroCAD 2004 International Scientific Conference, 6-7 March 2004, Miskolc, Hungary.

A.3. Chapters in books

- P. Kacsuk, T. Kiss, T. Delaitre, A. Goyeneche, S. C. Winter, G. Terstyanszky, Z. Farkas, C. S. Nemeth, T. Boczko, User Friendly Environment to Grid Enabled Legacy Codes, Engineering The Grid: Status and Perspective., Section 4: Programming Languages and Environments, January 2006, Hardcoverm, ISBN: 1-58883-038-1, http://www.aspbs.com/grid.html, 20-Nov-2009

A.4. News, dissemination and seminars

- Web Services Session for The National Grid Service's Grid Operations Support Centre (GOSC) meeting, Edinburgh, 29th October 2004.Hong Ong, from the Distributed Systems Group, University of Portsmouth and Ariel Goyeneche, Centre for Parallel Computers, University of Westminster.
- P. Kacsuk, T. Kiss, A. Goyeneche, T. Delaitre, Z. Farkas, T.Boczko, A High-Level Grid Application Environment to Grid-Enable Legacy Code, ERCIM News, Special: Grids: The Next Generation, No. 59, October, 2004, pp 44-45

Appendix B: GRTP Method Implementation

The GRTP method has been implemented and run in Grid5000, DAS, and AuverGrid to test its feasibility as a response time prediction service in production Grid computing environments. The presented implementation is the final step in the interaction through several software prototypes that helped in setting the system architecture design, selecting the programming language, and choosing the testing experiments to verify the correctness of the implementation.

The aim of this section is to present and explain the decisions made in the selection of the system architecture, the implementation of the different modules, the design of the supported database, as well as in the mechanism used for testing the entire system.

B.1. GRTP Service Architecture

To design the method's system architecture different prototypes were tested. The prototypes manually connected available software, such as MATLAB, Weka, and SPSS. But, the more options the proposed method was adding, the most difficult the executions of the prototypes were. The combination of existing software created a substantial overload in the testing work to the extent that the inclusion of new ideas was difficult to achieve without making mistakes and spending an important amount of time.

Therefore, the redesign of the final Service using a single architecture was considered as a final solution. Redesigning and implementing a new piece of software had the risk of introducing errors, but the amount of time and work that the running of prototypes were taking made this decision unavoidable.

The final architecture (Figure 54) uses as a centre point the response time prediction engine package. The engine uses other packages depending on the parameters that the prediction service would like to use. The advantages of using this architecture are:

- All main sections in the architecture are independent from each other. A job object and its related information is passed by one module to the other where the data transformation is implemented without affecting the rest of the architecture.
- The package Similarity, Clustering and Prediction can handle several classes of implementations within them. This gave the freedom of selecting any combination of Similarity, Clustering and Prediction methods.

– The modular approach makes possible the extension of the service in future releases by facilitating the inclusion of new packages and new classes within the current architecture.

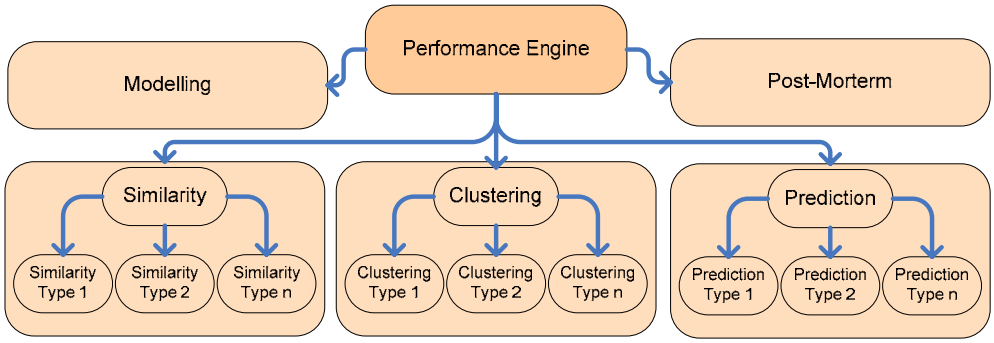


Figure 54 General implementation design of the response time prediction method.

B.2. Response Time Prediction Service Packages

In this section the description of the key packages of the GRTP Service are presented. A UML class diagram describes the structure of the system by showing the system's packages, classes, their attributes, and the relationships between the classes. Derivative packages that only complement this implementation are not presented.

B.2.1. Response Time Prediction Engine Package

The response time prediction engine package concentrates the core logic of the GRTP Service implementation (Figure 55). This package can be explained by describing the three main tasks of the EngineCrankshaft class:

– If a GRTP Service is started, a new EngineCrankshaft with all its details is created. Those details are composed of as Grid identification, workload meta-model, and the values for the reinforcement learning gamma and clustering threshold diameters. In the event that an already created process wants to be continued (a process that for some reason was cancelled while running), those values are recovered and set while the EngineCrankshaft is positioned into the last historical job that was analysed.

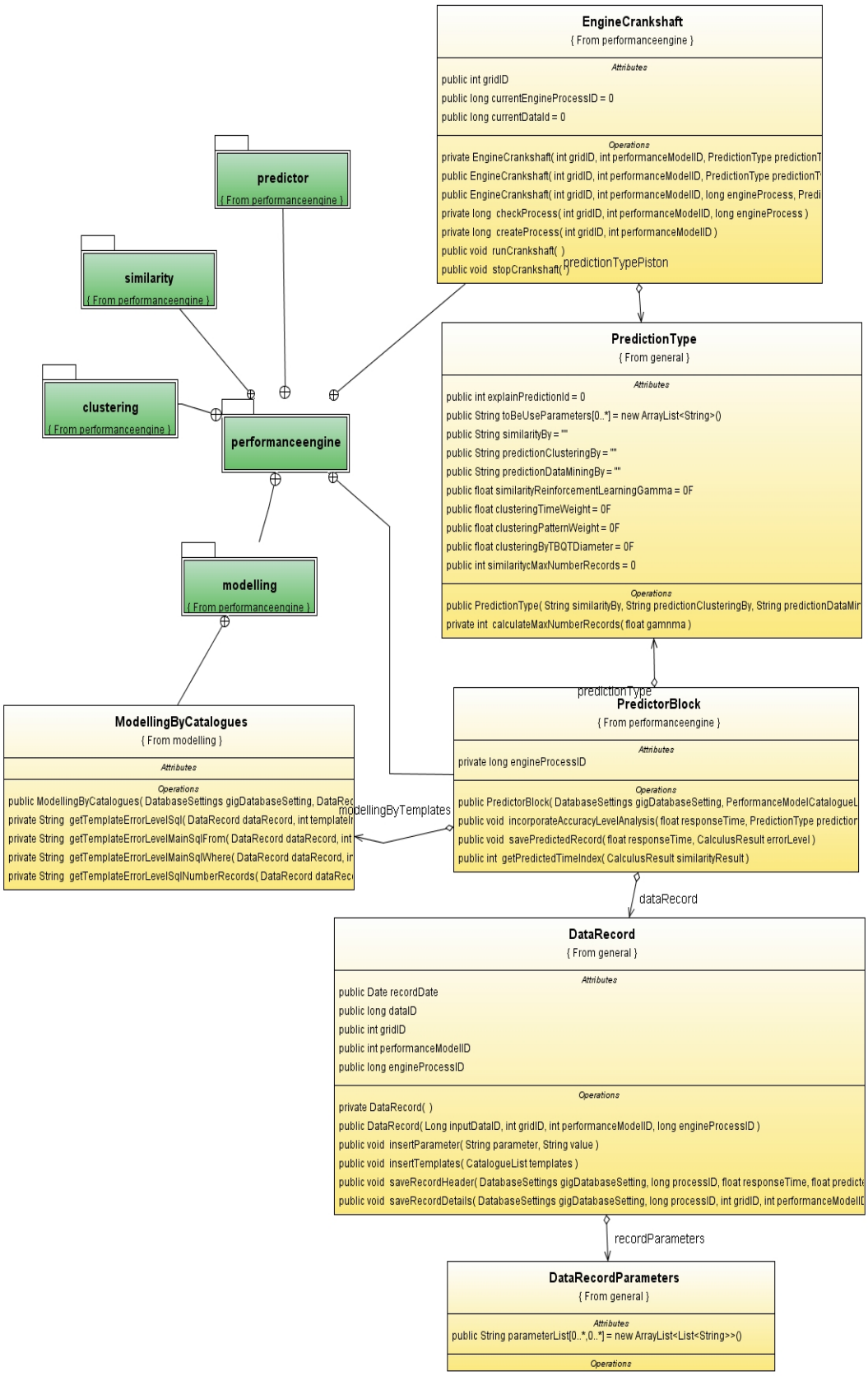


Figure 55 Response time prediction engine UML class diagram (relevant classes)

- Once the EngineCrankshaft is created (or recovered), the next task is to accept response time prediction requests. For each new request the EngineCrankshaft creates a prediction type (PredictionType) where the requested prediction parameters are stored (such as clustering and data mining parameters) and the following classes are called:
    - Workload Meta-model and Cataloguing Package (See Section B.2.2.)
    - Similarity package (See Section B.2.3.)
    - Clustering package (See Section B.2.4.)
    - Predictor package (See Section B.2.5.).
- Once the response time answer and the environment where this value has been predicted are returned to the Client, the final task is to expect for the real response time to be informed. With that value, the EngineCrankshaft can run all post-mortem activities such as the incorporation of the level of accuracy in the prediction. Once this process is completed, the saved information can be used in future predictions.

B.2.2. Workload Meta-model and Cataloguing Package

The modelling package (Figure 56) concentrates the functionalities that use the workload meta-model definition in order to create the list of WHC. The main tasks are:

- Based on the target Grid environment a workload meta-model is loaded (the workload meta-model is refreshed in background for the post-mortem activities). With that structure, the iWHC are created for the request.

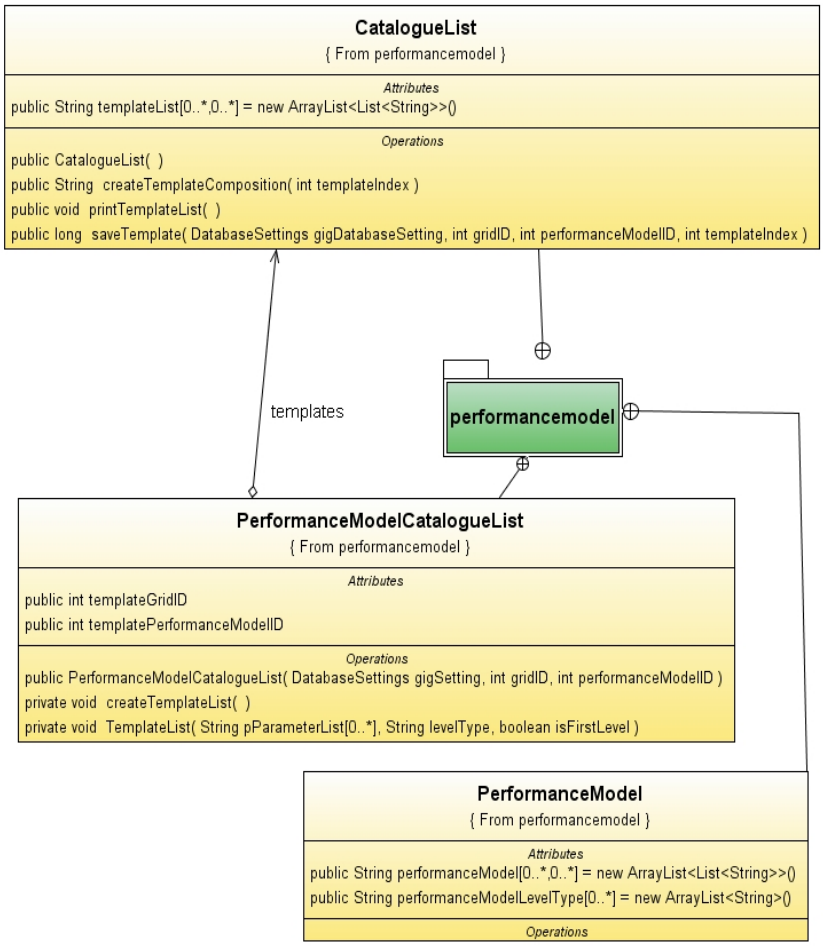


Figure 56 Workload meta-model and Cataloguing UML class diagram (relevant classes)

B.2.3. Similarity package

The modelling package (Figure 57) implements the procedure that helps to select the most appropriate iWHC from the list created in the previous package. Given that different options have been analysed and compared, the Similarity class decides which is the method selected in the specific process. The methods implemented are:

- SimilarityByConfidenceInterval: This is the method proposed in [62][63]. This option has been implemented in order to demonstrate its performance quality and also it is used to compare the new methods
- SimilarityByMeanOfErrorLevel: This method has been implemented to analyse how good the simple mean of the error perform.

– SimilarityByReinforcementLearningMeanOfErrorLevel: This is the proposed method where the reinforce learning discounted accumulative reward function is implemented.

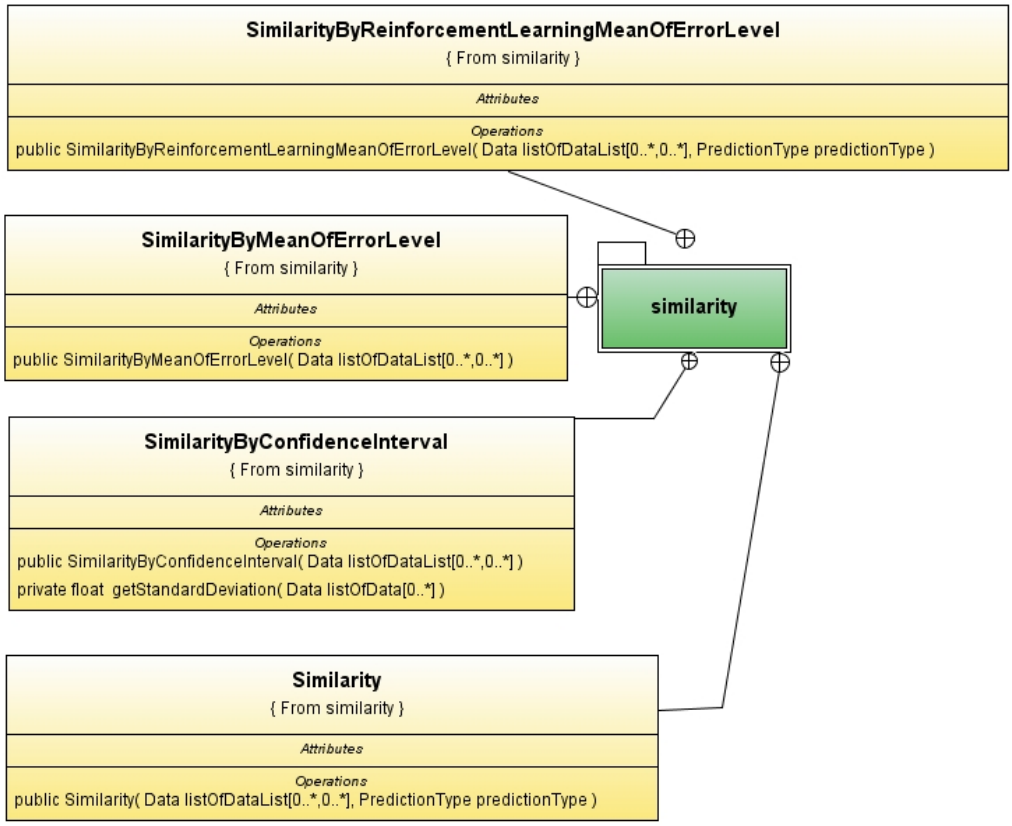


Figure 57 Ssimilarity UML class diagram (relevant classes)

B.2.4. Clustering package

Following the idea described in the previous package, the clustering package (Figure 58) implements different types of clustering methods that have been tested and compared in this research. The Clustering class decides which is the method selected in the specific process. The different implemented methods are:

– ClusteringByGroupingAll: When the data is treated all together without mining any other characteristics, this method should be used. The final result is a unique cluster with all records in an iWHC.

– ClusteringByQualityThreshold: The QT has been implemented to analyse its performance and accuracy against other methods.

– TimeBasedClusteringByQualityThreshold: This is the method proposed in this research based in the QT technique.

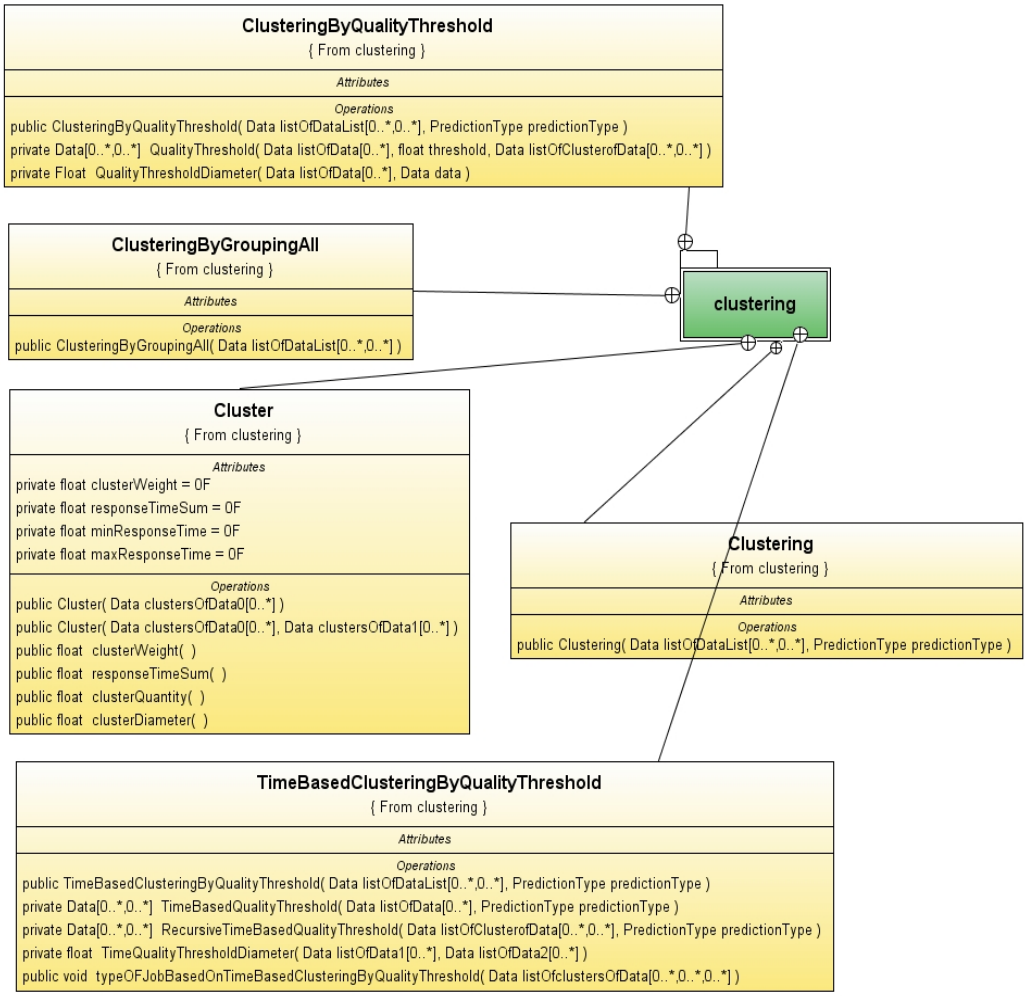


Figure 58 Clustering UML class diagram (relevant classes)

B.2.5. Predictor package

Finally, the predictor package (Figure 59) selects between the different prediction methods. The Predictor class may call using a specific prediction method. Otherwise, by understanding the type of job's iWHC, the Predictor class automatically selects between the "Stable" or "Unstable" prediction method. The implemented methods are:

– PredictionByMean: This is a prediction method used in several solutions, such as [61][62]. It has been implemented to be compared with the new methods.

- PredictionByNonStable: Implements the “Unstable” prediction method for “Unstable” iWHCs proposed in this research. This solution should be chosen when the job’s iWHC is classified as unstable.
- PredictionByStable: Implements the “Stable” method.

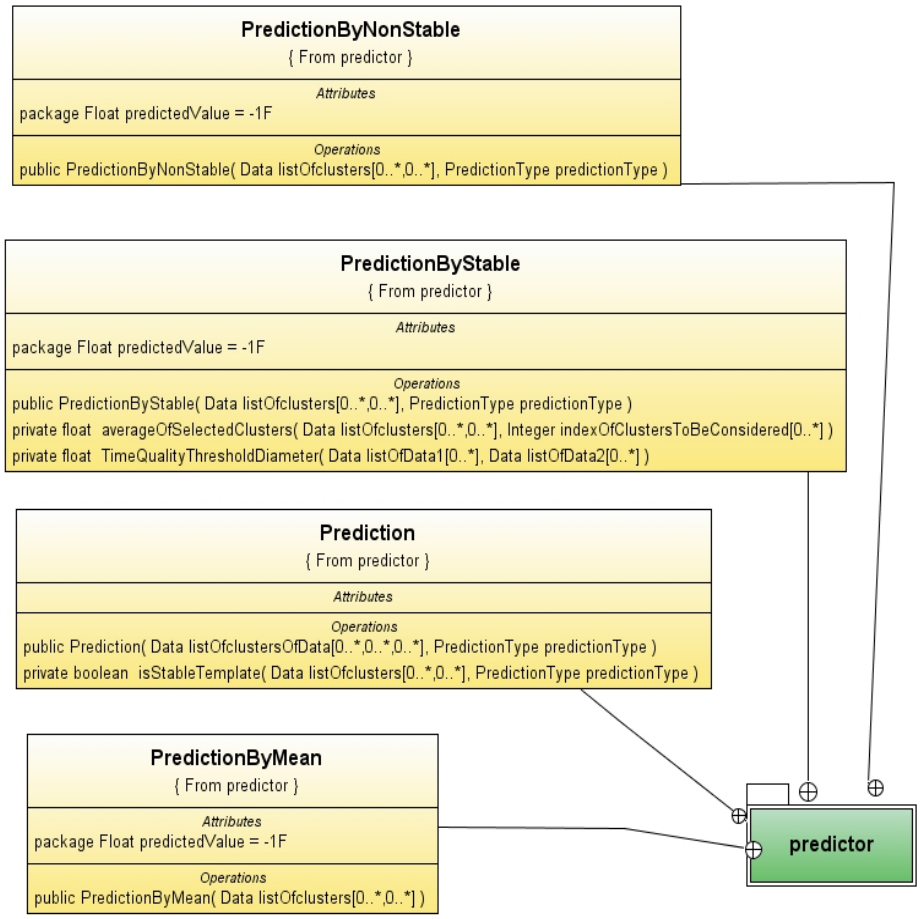


Figure 59 Predictor UML class diagram (relevant classes)

B.3. GRTP Database Entity-relationship Model

The conceptual representation of the structured data is presented in Figure 60. The diagram has been divided into four different sections:

- Workload Data: This section contains the Grid workload prediction model that is related to a Grid environment. All historical data that is used by the response time prediction engine is primary stored in this section of the model.
- Meta-model: This part of the model is used by the methodology that selects the relevant fields and creates a Grid related meta-model with them.

- Catalogues: From the meta-model the different iWHC are created. Those which are used by the different queries are stored in this section.
- Processed historical data: For each workload data entry, the prediction engine generates a number of values that help in future submissions. In this section the different methods selected by the engine are stored, together with the predicted results and accuracy level. It is interesting to see that for each job to be predicted a detailed result is kept for each iWHC, even for those that have not been marked as similar. This is the information used in future queries.

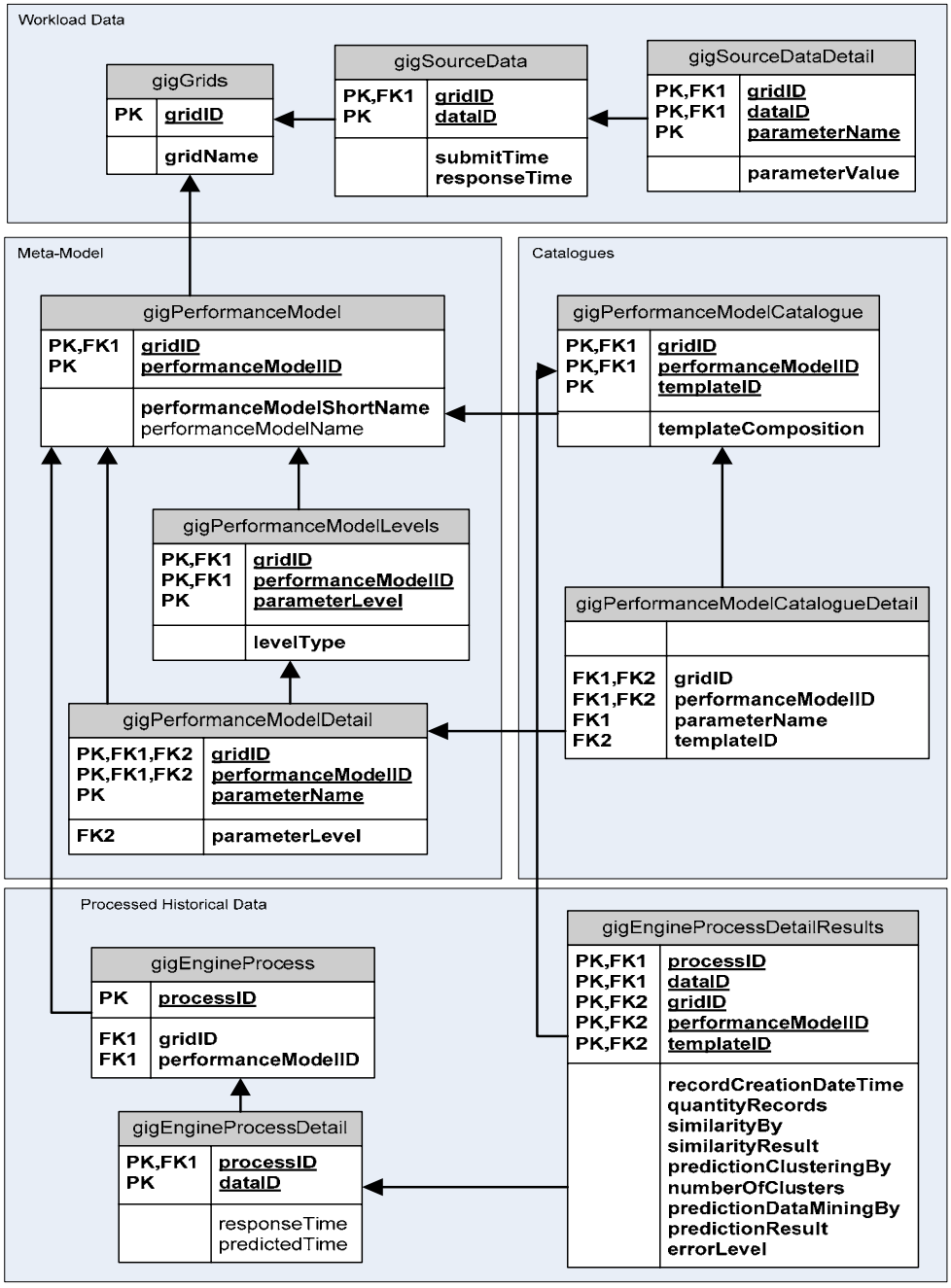


Figure 60 GRTP Database entity-relationship model

B.4. Testing of the GRTP Service Implementation

The GRTP Service implementation was tested using a *unit testing* approach. Unit testing is the method of making sure that the smallest units of software are working properly in isolation from the remainder of the code.

Getting the system architecture right is crucial to the success of a project and it also has a strong influence in the getting the testing approach right. The modular design of GRTP offered the chance to implement unit testing in each internal module. Without a modular approach, testing of the different parts of the system would have been a challenging task. Even more, a modular approach gave the opportunity to offer early design reviews avoiding the risk of quickly locked the system implementation into one particular approach and be blinkered from better solutions.

From the implementation point of view, the different independent modules evolved from prototypes to production. Prototyping was used as the process of gradually building isolated modules that finally composed the GRTP service implementation. A prototype was initially implemented using an external tool, such as MATLAB, WEKA, or SPSS. Once the different possibilities were analyzed and a decision was made, the final module was implemented in Java. The implementation was composed of designing, developing and testing phases.

From the testing point of view, the general approach was to take each module as a black box. The first set of tests used different parameters to ensure that large or small ranges of values and different combinations of inputs were not breaking the code. The idea of this initial test was to widen how much of the code was actually executed during testing. Once this initial test was passed, the correctness test was executed. In this step modules were tested using different sets of input-output results. If possible, the sets of input-output results were produced from proved sources. Otherwise, those sets were manually calculated.

For instance, the prototype and test for the entropy functions started with the implementation of a Java program that only used the entropy utility functions created in WEKA. WEKA has some predefined methods (entropyOverColumns and entropyConditionedOnColumns) for computing entropy and conditional entropy respectively. Once the approach of using the entropy functions was proved to work for GRTP, a final Java implementation was done. The main reason of this decision was that GRTP needs to work over millions of records stored in a database. WEKA functions require loading all the input records into memory. This aspect was slowing down the GRTP module and producing out of memory failures. Therefore, a new set of functions were implemented to work directly with a database system. In this case, the unit test was carried out using input-output results produced by WEKA. This was carried out by

selecting a random set of results and comparing the different outputs. Finally, the workloadMeta-Model creation was tested by setting a number of known inputs and expecting the corresponding outputs. These sets were worked out before the test was executed.

Another example was to prototype and test the clustering approach taken by GRTP. The first prototype used MATLAB and SPSS. The Fuzzy Clustering and Data Analysis Toolbox of MATLAB groups an input data set into clusters by different approaches, such as Kmeans and Kmedoid as hard partitioning methods and FCMclust, GKclust, GGclust as fuzzy partitioning methods. SPSS has three different procedures that can be used to cluster data: hierarchical cluster analysis, k-means cluster, and two-step cluster.

The clustering methods experiments helped to decide that a different approach was needed for Grid environments. Unfortunately QT was not implemented in any of these statistical analysis tools. Therefore, the option was to implement a prototype using R. R provide a function called qtclust that implements a generalization of the QT clustering algorithm. The only difference is that in each iteration not all possible cluster start points are considered, but only a random sample of a smaller size. This limitation is imposed because of the high running cost that this method have and as it was demonstrated in this thesis. Nevertheless, if the start point number is set to the size of the data set, the original algorithm as proposed. When QT was implemented in GRTP, the implemented class was unit tested against the results provided by R using random set of examples from the different testbeds. But, as it was explained in the thesis, QT was replaced by another module called TBQT. TBQT is a novel approach and therefore no implementation from an external source was available. In this case, the unit test was implemented using set of random inputs and manually created expected results.