UNIVERSIDAD DE EXTREMADURA

TESIS DOCTORAL

Optimización Multiobjetivo para la Planificación de Trabajos en Entornos de Computación Distribuida.

Multiobjective Optimization for Job Scheduling in Distributed Computing Environments.

Autor: María Arsuaga-Ríos

Departamento: Tecnología de los Computadores y de las Comunicaciones

Conformidad del director:

Fdo. Dr Miguel A. Vega-Rodríguez

2014

Director de la tesis: Miguel A. Vega-Rodríguez María Arsuaga-Ríos

Optimización Multiobjetivo para la Planificación de Trabajos en Entornos de Computación Distribuida.

Resumen

El paradigma de la computación Grid es definido como un sistema paralelo y distribuido que permite compartir, seleccionar y recolectar recursos autónomos distribuidos geográficamente de forma dinámica [41]. Todas estas acciones son llevadas a cabo en tiempo de ejecución dependiendo de la disponibilidad, capacidad, coste y calidad de los recursos requeridos por los usuarios. Este enfoque emerge de la sinergia entre la cooperación de los recursos computacionales con su control descentralizado y presentándolos como servicios.

Por tanto, uno de los más importantes actores, que participa en este control descentralizado, son los meta-planificadores, también conocidos como gestores de recursos. Este servicio es implementado en el middleware, lo que facilita el manejo del entorno Grid. La principal función de un meta-planificador es asignar los trabajos adecuadamente a los recursos siguiendo los requisitos computacionales y de calidad de servicio demandados por los usuarios. La computación Grid es ampliamente usada en el mundo científico para resolver experimentos complejos (conjunto de trabajos interdependientes) que requieren un alto rendimiento y productividad.

Por una parte, los científicos a menudo deben de cumplir plazos y presupuestos para experimentos enrolados en importantes proyectos. A causa de eso, la optimización del tiempo de ejecución y el coste económico asociado es un factor clave

a considerar en el proceso de planificación de trabajos llevado a cabo por los metaplanificadores. Sin embargo, este tipo de objetivos son conflictivos entre ellos, debido a que los recursos más económicos suelen ser más lentos que los más caros. El coste económico como objetivo es muy considerado en otros entornos distribuidos como la computación Cloud.

Por otra parte, la computación verde también conocida como Green IT está en auge dentro del campo de la computación durante estos últimos años. La computación verde consiste en permitir a las organizaciones realizar un uso más eficiente y racional de los recursos tecnológicos, reduciendo costes mientras se adoptan tecnologías y métodos que respeten el medio ambiente. Por tanto, el ahorro de energía se está convirtiendo en un nuevo objetivo a tener en cuenta en metaplanificadores en entornos distribuidos. El tiempo de ejecución y el consumo de energía son también objetivos conflictivos entre sí, ya que los recursos con más rendimiento frecuentemente implican un mayor consumo de energía. El ahorro de energía hoy en día está en auge en todos los entornos computacionales.

Para tratar con estos problemas una visión multiobjetivo es considerada en esta Tesis para buscar las mejores soluciones que minimicen a la vez los objetivos conflictivos mencionados. Gracias al conocido y ampliamente utilizado simulador GridSim [21] se han podido realizar diversos estudios con diferentes topologías distribuidas, workflows y propuestas de optimización multiobjetivo basadas en algoritmos evolutivos e inteligencia colectiva. También se han incluido comparaciones con otros resultados publicados en la literatura y metaplanificadores grid reales. Todos estos estudios incluyen los análisis estadísticos correspondientes.

Thesis director: Miguel A. Vega-Rodríguez María Arsuaga-Ríos

Multiobjective Optimization for Job Scheduling in Distributed Computing Environments.

Summary

The paradigm of grid computing is defined as a parallel, distributed system that allows sharing, selecting, and collecting autonomous resources geographically distributed in a dynamic way [41]. All these actions are carried out during the execution time of a job operation depending on the availability, capacity, cost, and quality of the resources required by the users. This approach emerges from the synergy between the cooperation of the computing resources with a decentralized control.

Therefore, one of the most important and challenging actors, that participates in this decentralized control, is the meta-scheduler, also known as a resource broker. This service is implemented in the middleware, which facilitates grid environment management. The main function of a meta-scheduler is to assign jobs to suitable resources by following computational and quality of service requirements demanded by users. Grid computing is widely used in the scientific world because it facilitates the execution of complex experiments (Set of interdependent jobs) that require high performance and throughput.

On one hand, scientists often have to consider deadlines and budgets for experiments related to important projects. Because of that, the optimization of execution time and cost is a key factor to consider in the job scheduling process carried out by the meta-schedulers. However, these types of objectives are in conflict with each other, because generally cheaper resources are usually slower than expensive ones.

Economic cost is an objective that is often considered in other distributed systems such as Cloud Computing.

On the other hand, Green Computing, also known as Green IT, is becoming a popular topic in the computational field in the last few years. Green Computing consists of enabling organizations to make a more rational and efficient use of their technological resources and reduce costs while adopting technologies and working methods that respect the environment. Energy consumption is a new valuable objective to optimize for meta-schedulers. Execution time and energy consumption are also conflicting objectives, because faster resources frequently imply higher energy consumptions. Energy reduction is considered in all types of computational environments.

In order to deal with these problems, a multi-objective approach is considered in this Ph.D. thesis to research the best solutions that minimize, at the same time, the mentioned conflicting objectives. Thanks to the well-known and widely used GridSim [21] simulator, several studies have been carried out with different distributed topologies, workflows, and multi-objective optimization proposals based on evolutionary and swarm intelligence algorithms. We have also included comparisons with other results published in the literature and with real grid meta-schedulers. All these studies include the corresponding statistical analysis.

Contents

Listing of tables

Listing of figures

this thesis is dedicated to my parents, Mariló and Juan, because they were always cheering me up and believing in me. they are the best of my life.

i would like also to dedicate this thesis to Manolo, who was supporting me in almost all the period of my thesis. thanks for giving me your smile every day.

moreover, i would like to thanks to all people that was helping me during this period:

thanks to Juan José and Isabel for being always there, even when i was far away from them.

thanks to my friends from Murcia: Mar (my adopted sister), Pilar, Patricia, Marta, Rocío, both Anas and Irene, for their patience and love during this long period and thanks to my new Greek friends: Elsa, Eva, Natasha, Gina and Mónica (almost Greek) for understanding me and giving me force during the writing of this thesis.

thanks to Almudena for helping me during my first abstracts and her advices.

thanks to Jose Manuel for giving me his enthusiasm from the beginning of my career until now and I hope to keep that relation in the future.

thanks to Miguel Angel, my supervisor, because without him i could not have finished this thesis; thanks to his advice and help.

Acknowledgments

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Morbi commodo, ipsum sed pharetra gravida, orci magna rhoncus neque, id pulvinar odio lorem non turpis. Nullam sit amet enim. Suspendisse id velit vitae ligula volutpat condimentum. Aliquam erat volutpat. Sed quis velit. Nulla facilisi. Nulla libero. Vivamus pharetra posuere sapien. Nam consectetuer. Sed aliquam, nunc eget euismod ullamcorper, lectus nunc ullamcorper orci, fermentum bibendum enim nibh eget ipsum. Donec porttitor ligula eu dolor. Maecenas vitae nulla consequat libero cursus venenatis. Nam magna enim, accumsan eu, blandit sed, blandit a, eros.

"I dream my painting and I paint my dream".

Vincent van Gogh

1 Introduction

In this chapter, a general overview of the problem with a short explanation of the purpose of this study and the multi-objective approaches studied in this thesis are briefly introduced in section 1.1. The outline of scientific purposes and objectives for the research are performed in section 1.2 and the corresponding contributions carried out in this thesis are described in section 1.3. Finally, the structure of this thesis is enumerated in section 1.4.

1.1 General Overview

Grid computing consists of a decentralized control of heterogeneous and geographically distributed resources [42]. Meta-schedulers are the actors that participates in this decentralized control, and their main function is assign the jobs to suitable resources by accomplishing computational requirements and quality of service de-

manded by users [41].

Grid computing is widely used in the scientific world to solve complex experiments that require high performance. Scientists often have to consider deadlines, budgets or energy consumption from their experiments related to important projects. Because of that, the optimization of execution time and cost, and/or energy consumption is a key factor to consider in the job scheduling process carried out by the meta-schedulers. However, these types of objectives are in conflict with each other, because cheaper resources are usually slower than expensive ones or faster resources can consume more energy, necessitating a multi-objective optimization.

In this thesis, a study of different multi-objective algorithms is presented to solve this multi-objective problem. The implemented algorithms are based on Swarm Intelligence and evolutionary populations. Swarm intelligence is a kind of intelligence that emerges from collaboration and competition among individuals. Evolutionary algorithms are based on population evolution with only parental relationships between their members. Five multi-objective algorithms from different fields (biology, physics, and sociology) have been adapted and evaluated:

- Multi-Objective Gravitational Search Algorithm (MOGSA) is built from the Gravitational Search Algorithm [77], and the agents (candidate solutions) are considered as planets that interact among them according to the physical field.

- Multi-Objective Artificial Bee Colony (MOABC) is based on the Artificial Bee Colony [50], [49] from the biological field, and its collaborating agents are represented by bees.

- Multi-Objective Small World Optimization (MOSWO) is a multi-objective approach based on the small world phenomenon from the sociology field point of view. The MOSWO approach implements some practices from the single-objective algorithm Tabu Small World Optimization (TSWO) [70], and its agents are represented as nodes in a social network.

- Multi-Objective Firefly Algorithm (MO-FA) is based on the Firefly Algorithm [97], which is inspired from firefly behaviour, and it is biological algorithm.

- Multi-Objective Brain Storm Algorithm (MOBSA) is based on a novel algorithm called Brain Storm Optimization (BSO) ([79], [80]). The BSO algorithm is inspired by humans behaviour (sociological field), when a brainstorming process is applied in a group, especially with different backgrounds, in order to resolve a complex problem.

One of the main contributions of this research is the adaption of these algorithms to deal with multi-objective requirements that are in conflict with each other (execution time and economic cost or energy consumption). Therefore, to give more reliability to this multi-objective study, an evaluation with well-known multi-objective algorithms has been accomplished: Non-dominated Sorting Genetic Algorithm II (NSGA-II) [29], which is the standard multi-objective algorithm and Multi-Objective Heterogeneous Earliest Finish Time (MOHEFT) [37], a recent multi-objective algorithm that has demonstrated very good results in the workflows scheduling field. All these evaluations include the corresponding statistical analysis.

GridSim [21] is the simulator used to implement all the meta-schedulers. GridSim is a powerful simulator for grid environments, it is very well-known and used in the field. GridSim allows us to configure complex topologies and resource features, therefore, their results are very close to the results that would be obtained in real grid environments.

In this thesis, six workflows–Gaussian, Gauss-Jordan, LU decomposition, FindMax, Fast Fourier Transform (FFT), and Stencil–have been deployed with all the implemented meta-schedulers to study their behaviour in each situation. In addition, two different grid environments have been used to reinforce the study of their behaviour in different scenarios. Also, the best meta-scheduler, in this case MOABC, has been compared with two real meta-schedulers. The Workload Management System (WMS) [47], by gLite (Lightweight Middleware for Grid Com-

puting) [38], is a European middleware, which is the most extended in grid environments. The second meta-scheduler is the Deadline Budget Constraint (DBC) of Nimrod-G [22], which is used to show the relevance of our results.

1.2 Objectives

The main objective of this thesis is the design of a multi-objective optimization meta-scheduler to resolve the job scheduling problem in distributed environments, such as Grid computing. Execution time, economic cost, or energy consumption optimizations are applied in the approaches presented in this thesis. Therefore, real quality parameters should be taken into account making the problem more complex and necessitating the use of complex heuristics and multi-objective meta-heuristics to achieve an effective solution. The concrete objectives of this thesis are presented as follows:

- Study and analysis of real meta-schedulers on grid environments: WMS and DBC.
- Study of new technologies for energy saving in distributed environments.
- Analysis and study of all necessary concepts related to multi-objective optimization problems.
- Objective functions analysis and implementation for each multi-objective optimization problem proposed in this thesis.
- Study and analysis of the simulator used to obtain the required data, GridSim.
- Bibliography study about the job scheduling problem in distributed environments and the strategies carried out for solving it.
- Study, evaluation, and extension of the multi-objective meta-heuristics that are applied to solve the problem, by analysing their suitability for each one accordingly to the characteristics of the problem.

- Development and parametric adjustment of the algorithm approaches with the purpose of obtaining quality results to solve the problem.
- Statistical analysis and comparison of the obtained results for each algorithm regarding different datasets (grid topologies and scientific workflows).
- Extension design and development from the simulator, GridSim, to study the multi-objective job scheduling problem in distributed environments.

1.3 Contributions

The contributions of this thesis are established from the following points of view:

- First, from the scientific point of view, the approach presented represents an advanced and deep understanding of evolutionary and swarm intelligence algorithms and multi-objective meta-heuristics in general, to resolve complex optimization problems. Particularly, these strategies are being applied to Grid computing. However, these techniques could be extrapolated to other fields, where techniques are needed to solve complex problems, which can not be tackled in another way.
- New strategies have been designed to provide multi-objective optimization in the scientific field. These strategies are not only based on biology inspiration or physics, as is usually common in the bio-inspired optimization world, but also on social inspiration through algorithms based on human behaviour, such as the brainstorming process or small world phenomena. These new approaches imply an extension of the multi-optimization study.
- Regarding the fields where this research is applied, this project represents a study from two points of view of the job scheduling problem in distributed environments, the simultaneous execution time and cost optimization view and the energy consumption and responsiveness optimization view. Therefore, the multi-objective optimization of distributed infrastructures in meta-

schedulers leads to suppose an economic and ecological advance by offering better responsiveness for scientific and private projects.

- Finally, from the technical and research point of view, this research contributes to the extension of the well-known and often used simulator GridSim [21] with the implementation of novel multi-objective algorithms (evolutionary and swarm intelligence algorithms) able to optimize the aforementioned objectives, by offering a multi-objective study platform with metric calculations, such as hypervolume, set coverage, and statistical analysis.

1.4 Thesis Organization

This thesis is organized as follows:

- Chapter 2 presents the related work of the grid scheduling problem and the current multi-objective advances provided by literature.
- Chapter 3 describes the problem statement with an introduction of the multi-objective approach.
- Chapter 4 explains the five multi-objective approaches presented in this thesis and also the application of the standard and well-known NSGA-II algorithm to the problem.
- Chapter 5 details the test environments used with the aim to make the experiments reproducible for future researches.
- Chapter 6 presents the methodology followed, indicating the multi-objective quality metrics used, the statistical analysis applied, and the parametrical study performed for all the algorithms.
- Chapters 7 and 8 analyse and compare the quality and performance of all multi-objective algorithms presented and compare other standard and well-known multi-objective algorithms, such as NSGA-II, MOHEFT. Real grid meta-schedulers, such as WMS and DBC are also studied.

- Chapter 9 summarizes the conclusions obtained by this thesis and discusses future lines to follow.
- Annexes A and B elaborate on all the research achievements.

"Study the past if you would define the future" .

Confucius

2 Related Work

This chapter is divided in two sections. Each section details the development of each problem presented in this research: The Economic cost and Execution Time Optimization Problem and the Energy Consumption and Responsiveness Optimization Problem for grid meta-schedulers.

2.1 Economic Cost and Execution Time Optimization for Job Scheduling in Grid Environments

Real grid meta-schedulers have been considered for evaluating the quality of the multi-objective algorithms proposed for optimizing the execution time and cost requirements demanded by users. Currently, the most used meta-scheduler is the Workload Management System (WMS) [47], by gLite (Lightweight Middleware for Grid Computing) [38], a European middleware, which is the most extended

in grid environments. The second meta-scheduler studied is the Deadline Budget Constraint (DBC) of Nimrod-G ([22], [1]). Its algorithm adjusts both the budget and the deadline per job specified by the user. Moreover, the Nimrod-G system is mentioned as the only one that optimizes execution time and economic cost according to a recent study of task scheduling [92]. Xhafa and Abraham [95] reveal the complexity of the job scheduling problem and show the utility of meta-heuristiscs for the multi-objective design of grid schedulers.

At present, multi-objective algorithms are emerging in literature for the optimization of objectives that are in conflict with each other. The optimization of these objectives tends to be resolved by different versions of multi-objective evolutionary algorithms [103]. Evolutionary algorithms are usually based on genetic algorithms or other bio-inspired algorithms, such as artificial weed colonies [58] or particle swarms ([51], [24]). They have demonstrated a high performance in optimizing multiple functions in complex environments. Execution time optimization is one of the most important goals for job scheduling problems on grid systems ([39], [60], [87]). These research efforts have tried to fulfil deadlines or minimize response time for scientific applications with independent or dependent jobs. Multi-objective approaches are also emerging to solve the job scheduling problem. Genetic algorithms are usually applied to this problem by optimizing execution time and cost ([99], [100], [101], [85], [55]).

Experimental results show that multi-objective genetic algorithms offer better solutions than classic meta-heuristics, such as Simulated Annealing (SA), Duplex, and Min-Min to control the resources located in high scale distributed systems [76]. However, the test environments do not take into account specific topologies with network configurations such as the following: speed transmission, delay, Maximum Transfer Unit (MTU), etc. They also lack specifications such as resource location or computational resource features (operating system, number of machines, CPUs, speed, cost, etc.) by being generally homogeneous. Moreover, the majority of the research does not consider workflows that follow the DAG model design (Dirigid Acyclic Graph), which implies the study of workflows with dependent jobs .

New multi-objective algorithms are usually compared with the standard NSGA-II (Non-dominated Sorting Genetic Algorithm II) ([29], [28]) to evaluate quality. NSGA-II is a very popular algorithm in the multi-objective optimization field due to its high efficiency demonstrated in the literature. For this reason, the proposed algorithms are compared with NSGA-II.

At this point, the contributions for the multi-objective optimization problem in terms of economic cost and execution time are presented. In this research, the problem is tackled by multi-objective algorithms from evolutionary techniques and swarm intelligence behaviours.

On one hand, five multi-objective algorithms have been developed as follows: Multi-Objective Gravitational Search Algorithm (MOGSA) ([3], [14]), Multi-Objective Artificial Bee Colony algorithm (MOABC) [15], Multi-Objective Small World Optimization (MOSWO) [4], Multi-Objective Firefly Algorithm (MO-FA) [5] and Multi-Objective BrainStorming Algorithm (MOBSA) ([12], [8]). Note that MO-FA has been studied after using the algorithm for the energy consumption and responsiveness optimization research carried out in this thesis. All these algorithms have been implemented in a multi-objective version from their well-known single-objective version: TSWO [70] → MOSWO, GSA [77] → MOGSA, ABC [50] → MOABC, BSO [80] → MOBSA, FA [97] → MO-FA.

On the other hand, these algorithms are compared with NSGA-II, which is a recognized standard algorithm in the multi-objective optimization field, to study the efficacy and efficiency of the approaches presented. Moreover, all the algorithms are also compared with the multi-objective version of HEFT [102] [93], one of the most-used algorithms in workflow scheduling, Multi-Objective Heterogeneous Earliest Finish Time (MOHEFT) [37], in order to show the quality of the proposed algorithms. It is worth highlighting that these multi-objective approaches have been developed to solve the job scheduling issues in grid computing in order to optimize execution time and economic cost, but they could also be used in future research to solve other problems that require multi-objective optimization.

In this thesis, comparisons have been accomplished with other algorithms and

techniques collected in the literature. On one hand, the proposed algorithms have been compared with the DBC algorithm and with the meta-scheduler WMS. The DBC algorithm is based on an algorithm that tries to keep cost within budget requirements and time within the deadline given. On the other hand, a comparison with the grid meta-scheduler WMS has been carried out. Finally, a comparative study among the proposed algorithms is presented by using two real grid topologies. IBERGRID [16] and the combination of EU DataGrid [84] with the resource features of the testbed WWG [22]. This comparative study has high relevance in the distributed computing field, because it presents an exhaustive analysis of six multi-objective algorithms on two real topologies using six different workflows ([88], [94], [19]) with dependent jobs. In contrast to all the research previously found in literature by other authors, in this thesis, specific and detailed features of the network architectures and computing resources are presented as well as the detailed specifications of the applied workflows.

2.2 Energy Consumption and Responsiveness Optimization for the Job Scheduling problem in Grid environments

Green computing, also known as Green IT has been a trendy topic in the computing field during the few last years. Green computing allows organisations rational, efficient use of their resources and to the reduction of costs by adopting new methods and technologies that respect the environment. Many technological advances such as the energetic efficiency in super computers are emerging. In fact, a list of supercomputers that use green technologies in the world, called 'The Green 500' [26], is published. This list details some of the new energy saving advances. The Grid infrastructure could contribute to the reduction of the energy consumption thanks to its sharing policies [43]. However, although Grid matches some concept points of Green computing, Grid scheduling is still one of the most important problems to resolve from standpoint of energy saving.

In the Wieczorek et al. [92] model the job scheduling problem is classified in different taxonomies by taking into account current Grid systems, however,

among all these systems energy consumption optimization is not considered. Green Computing is creating opportunity for a new goal for job scheduling on grid environments. Current research deals with physical energy management by switching off idle resources ([18], [46], [31], [68]) or the cores of processing for multiprocessors [59], or by reducing energy consumption for data communication [66].

Dynamic Voltage and Frequency Scaling (DVFS) [89] is the most-used technique in current research. DVFS optimizes energy consumption by decreasing the voltage and clock frequency (CPU speed) for non-critical or idle nodes. The DVFS technique is used by heuristics or greedy algorithms to balance energy consumption with execution time using predefined weights. The work of Khan et al. ([54], [52], [53], [65]) takes into account a multi-objective approach for job scheduling in grid systems from a low-level point of view. This research simulates the power off and DVFS techniques.

Although the research of Khan et al. ([54], [52], [53], [65]), considers the optimization of both execution time and energy consumption, it lacks complex grid topologies and detailed specifications for the heterogeneous grid resources in experimental tests. Moreover, it does not provide experiments with detailed information of the workflows executed on the grid and is therefore unreproducible.

Recently, a novel multiobjective approach, MOHEFT, is emerging for optimizing both energy saving and makespan in distributed environments ([40], [36], [34], [35]). MOHEFT ([37]) is based on the well-known HEFT approach ([86], [102], [93]) for optimizing workflow-scheduling problems. MOHEFT has demonstrated higher quality results than SPEA 2 ([105] [100]), optimizing execution time and energy consumption.

Our multi-objective approaches detail more information about the test environments and workflows used to solve these problems. This research contribution assumes the use of ecological resources that are currently emerging using software and other techniques, such as CLUES [44] or EnergySaving Cluster (ESC) [32]. These techniques manage idle resources and other approaches per site or machine. Therefore, our proposal is the perfect complement to the ecological software that is emerging. Aiming at testing the effectiveness of our multi-objective approaches,

we studied each of them to resolve our problem for energy consumption ([5], [7], [11], [10], [13], [9]). In addition, a comparison with MOHEFT ([37], [34], [35], [36]) is performed because of its efficiency as a multi-objective workflow-scheduling algorithm. NSGA-II and the real grid schedulers, WMS and DBC, are compared and analysed as well.

All the multi-objective meta-heuristics approaches and the real grid meta-schedulers published previously in literature have been studied. Furthermore, a statistical analysis ([78], [57], [62]) is applied to the proposed multi-objective evolutionary algorithms using different scenarios. These scenarios are heterogeneous with the objective of evaluating the algorithms' performance. In each topology mentioned in the previous problem ([16], [84], [22]), six different workflows ([88], [94], [19]) are executed with respect to the job numbers, execution time per job, and the data transfer rates between them. Due to the topologies' heterogeneity and the variety of workflows performed, this study has important relevance in the distributed computing domain.

"There are unsolved problems, not solved, that keep the mind active".

Erwin Guido Kolbenheyer

3

Problem Statement

Traditionally, real world problems are solved by optimization techniques that minimize or maximize a single objective. Many of these problems usually have more than one objective to accomplish. Timing optimization is one of the most important objectives in scheduling problems, but other objectives are also considered important, such as the length of the schedule, the availability and cost of resources (machine scheduling), preferences of human resources (workforce scheduling), compliance with regulations (educational timetables), etc. Traditional techniques try to combine multiple objectives into a single scalar value by using weighted methods according to the importance suggested by the experts. However, in many instances, these objectives have the same level of importance and conflict with each other. Currently, multi-objective optimization techniques are emerging in scheduling problems ([81], [74]) allowing the opportunity to optimize more than one objective with the same importance and providing decision support for end

users. In this chapter, a general multi-objective optimization problem and its application to the different fields related to scheduling problems are defined. Finally, the problem statement of the research carried out in this thesis is presented in last section.

3.1 Multi-objective Optimization Definition

In the current marketplace, diverse real-world problems from different fields are solved by multi-objective optimization techniques. In this section, background and basic multi-objective concepts are described. In general, a Multi-objective Optimization Problem (MOP) is defined as the task of finding a decision variable vector that satisfies the problem constraints and optimizes an objective function vector [55]. The functions are usually in conflict each other, due to the difficulties in improving one objective function without negatively affecting another objective function. Therefore, the term optimization means to find a solution vector with acceptable values for all objective functions. Multi-objective problems must optimize m objective functions at the same time. The optimization process can deal with the minimization/maximization of all the objective functions. More information about multi-objective optimization can be found in the following references: [106], [107], [28] and [25] . The most important definitions are explained in this section.

Definition 1: ***Multi-objective Optimization Problem (MOP).*** A Multi-objective Optimization Problem includes a set of n parameters (decision variables), a set of m objective functions, and a set of k constraints. Objective function and constraints are a function of the decision variables. The mathematical definition is shown as follows (Equation 3.1):

$$\begin{aligned} \textit{Optimize} \quad & \vec{y} = \vec{f}(x) = (f_1(x), f_2(x), \ldots, f_m(x)) \\ \textit{subject to} \quad & \vec{e}(x) = (e_1(x), e_2(x), \ldots, e_k(x)) \leq \phi \\ \textit{where} \quad & \vec{x} = (x_1, x_2, \ldots, x_n) \in X \\ & \vec{y} = (y_1, y_2, \ldots, y_m) \in Y \end{aligned} \tag{3.1}$$

where x is the decision vector that belongs to the decision space X and y is the objective vector that is represented in the objective space Y. The decision variables vector can be discrete or continuous while the objective functions can be linear or nonlinear and discrete or continuous. The function $f: X \rightarrow Y$ is a transformation of the decision variables vector x on a response vector y (see Figure 3.1.1).

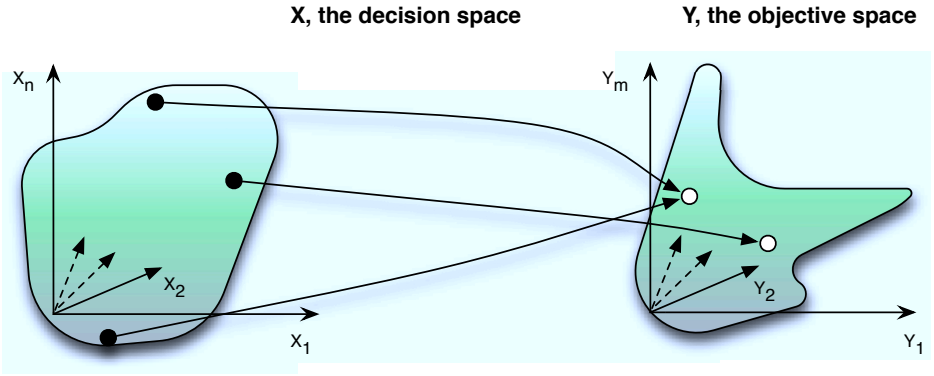

Figure 3.1.1: Multi-objective optimization function [6].

In single-objective optimization problems, the optimization process obtains one optimum solution that minimizes or maximizes a single objective function. When the problem is multi-objective, 'optimum' must be redefined because of the presence of multiple, conflicting objectives. The conflict among objectives does not allow the improvement of one without negatively impacting the others. Thus, a multi-objective optimization finds the best compromise among those objectives. The best compromise is called Pareto optimum.

Definition 2: ***Pareto Optimality.*** A decision vector $x \in X_f$ is a Pareto Optimum regarding a set $A \subseteq X_f$ if and only if:

$$\nexists a \in A: \; a \prec x \tag{3.2}$$

where X_f is the set of feasible solutions that satisfy the constraints of the problem $(\vec{e}(x))$. This definition specifies x as a Pareto Optimum if not other feasible vector a exists that decreases any of the objective functions without simultaneously

increasing another (assuming minimization in all the objective functions). This comparison is also known as dominance ($\prec$).

Definition 3: ***Pareto Dominance.*** For any two decision vectors a and b (assuming a minimization problem):

$$\begin{aligned} a \prec b \ (a \textit{ dominates } b) &\iff \vec{f}(a) \ < \ \vec{f}(b) \\ a \preceq b \ (a \textit{ weakly dominates } b) &\iff \vec{f}(a) \ \leq \ \vec{f}(b) \\ a \sim b \ (a \textit{ is indifferent to } b) &\iff \vec{f}(a) \ \not\leq \ \vec{f}(b) \ \wedge \ \vec{f}(b) \ \not\leq \ \vec{f}(a) \end{aligned} \tag{3.3}$$

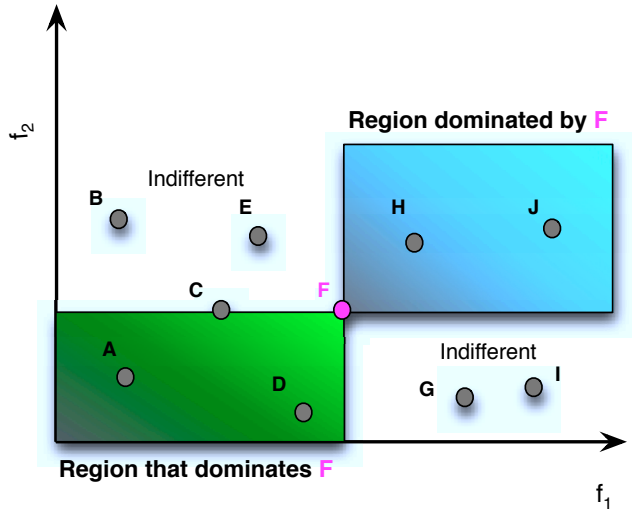

Figure 3.1.2: Graphical representation of dominance regions regarding the solution F [6].

Figure 3.1.2 shows a graphical representation of dominance regions regarding the solution F in a minimization problem. The solutions that are in the green region (A, C, and D) dominate solution F. In the case of A and D, both values for the objective functions are better than those obtained by F. Also, despite C and F obtaining the same value for the objective function f_2, solution C has better value for the objective function f_1. On the other hand, F is not dominated by the solutions

B, *E*, *G*, and *I*, being equally beneficial. Finally, the solutions in the blue region, *H* and *J*, are dominated by *F*.

Definition 4: ***Optimal Pareto Set.*** The Optimal Pareto set for a multi-objective problem is denoted as P^* and is defined as:

$$P^* := \{x \in X \mid \nexists x' \in X, \quad \vec{f}(x') \preceq \vec{f}(x)\} \tag{3.4}$$

The optimal Pareto set contains all the solutions from the decision space whose objective vectors can not be improved simultaneously. The objective vectors from the optimal Pareto set are called non-dominated solutions and generate the Pareto Front (PF^*).

Definition 5: ***Pareto Front.*** The Pareto front PF^* from an optimal Pareto set P^* is defined as:

$$PF^* := \{a = \vec{f}(x) \mid x \in P^*\} \tag{3.5}$$

Generally, the solutions found by multi-objective optimization are presented as Pareto front plots generated from each algorithm in order to evaluate them. In Figure 3.1.3, a Pareto front example is shown. Solutions that represent the best possible trade-offs among the objectives are the aim of the search (in case of Figure 3.1.3, solutions lying on the "knee" of the Pareto curve).

3.2 Multi-objective Optimization applied on Scheduling Problems

Scheduling problems appear continuously in diverse real-world situations. The entities (people, tasks, vehicles, meetings, etc.) usually follow a space-time pattern in which some constraints must be fulfilled and certain objectives have to be achieved. The scheduling process objective is to find the optimal schedules that satisfy user needs. Many scheduling problems are multi-objective by nature because users normally have more than one objective, such as minimizing the length of a schedule, satisfying preferences among human resources (workforce schedul-

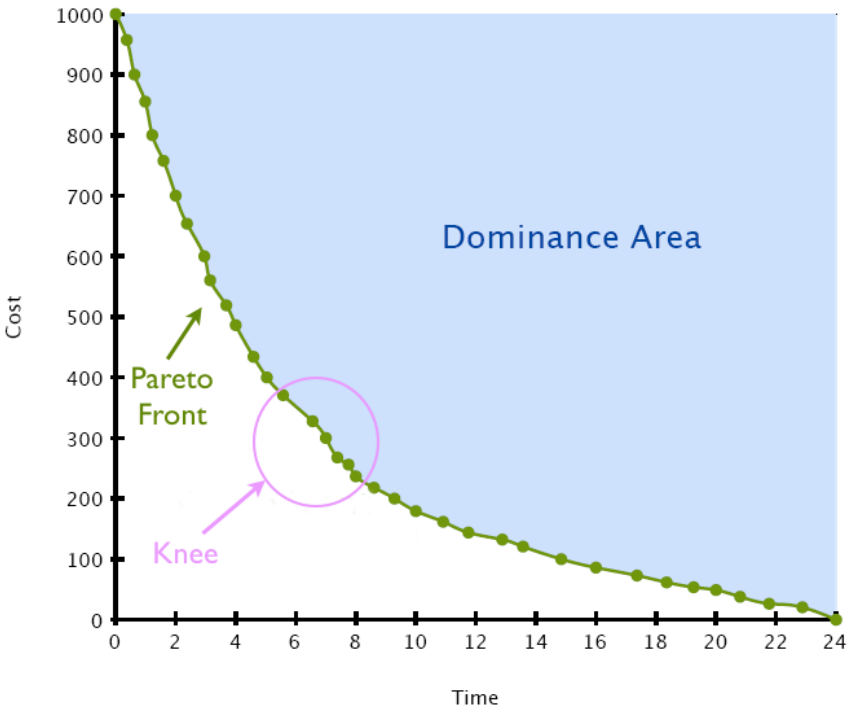

Figure 3.1.3: Pareto Front example with two objective functions [6].

ing), maximizing compliance with regulations (educational timetables), minimizing the tardiness of orders (production scheduling), optimizing task/job scheduling (machine scheduling), etc. As we mention previously, traditionally these multi-objective problems are solved using a weighted combination of the objectives in order to manage one objective function. However, in many real multi-objective scheduling problems, it is more desirable to offer consideration for different objectives separately in order to obtain a better trade-off among all the conflicting objectives. This approach is given by the Pareto optimization technique. Multi-objective algorithms are emerging that use the Pareto optimality to support decision makers in almost all the domains. Next, some of the outstanding multi-objective scheduling problems are briefly presented. An extension of this part of the chapter can be found in [82] and [74].

3.2.1 Multi-objective Workforce Scheduling

Personal scheduling manages the requirements of employees and employers while taking into account working regulations. Mobasher [72] carried out an example of a multi-objective approach in a clinical context. This approach consists of tackling a multi-objective nurse scheduling problem in which shift preferences as a proxy for job satisfaction and patient workload as a proxy for patient dissatisfaction are considered. Another more general approach, applicable to nurse scheduling is presented by Li et al. [63]. They introduce a hybrid algorithm combining goal programming and meta-heuristic search to create compromise solutions in difficult employee scheduling problems. In the context of airlines, two objectives are optimized, which minimize airline operation cost and maximize crew staff satisfaction. Moudani et al. [73] use a genetic algorithm and a greedy algorithm to manage multi-objective optimization. Yannibelli and Amandi [98] have proposed another multi-objective evolutionary algorithm to optimize the project scheduling problem using human resources. This approach aims to minimize the makespan for the project and assign the most effective set of human resources to each project activity.

3.2.2 Multi-objective Educational Timetables

Educational timetables are critical in terms of inaccurate predictions of student enrollment, mistakes in event lists or resource availability, and inadequate selection of hard and soft constraints [82]. A multi-objective linear programming model was proposed by Ismayilova et al. [48] to consider administration and instructor preferences by using a weighted priority to schedule the class-course timetable. Educational timetables also consider exams timetables, Côté et al. [27] present a multi-objective evolutionary algorithm that optimizes the maximum free time for students while satisfying the clashing constraint exam conflicts without regard to the seating capacity. Moreover, this multi-objective approach takes into account the timetable length as an optimization objective. The Balanced Academic Curriculum Problem is other type of educational timetables that consists of assigning courses to teaching terms satisfying prerequisites and balancing the credit course load within each term. Castro et al. [23] presented a multi-objective genetic algorithm to deal with this problem.

3.2.3 Multi-objective Production Scheduling

In manufacturing, the purpose of scheduling is to minimize production time and costs, by informing a production facility when to run production, what manpower is needed, and on which equipment should be utilized. Production scheduling aims to maximize the efficiency of the operation and reduce costs. Currently, multi-objective production scheduling problems are widely studied in several domains in real life [61]. Different objectives should be considered in production scheduling problems as the makespan (response time), the mean completion time (the mean of slower activity during production), the maximal tardiness or the mean tardiness (the maximum and minimum mean times obtained after several production runs). In the study carried out by Loukil et al. [67], these objectives are optimized with a multi-objective simulated annealing approach. Moreover, multi-objective algorithms offer decision support; Mansouri et al. [69] aim to identify the gaps in decision-making support based on multi-objective optimization

(MOO) for make to order supply chain management. Although, these scheduling problems are based on the economy, multi-objective approaches allow consideration for other aspects, such as the intangible value of freshness in their products [2].

3.2.4 Multi-objective Machine Scheduling

Machine scheduling refers to problems where a set of jobs or tasks has to be scheduled for processing in one or more machines [74]. Each job or task consists of one or more operations (sub-tasks) and usually, a number of additional constraints must be also satisfied. Examples of such constraints are precedence relations between the jobs and the limited availability of resources. Machine scheduling is widely studied, and multiple multi-objective approaches are applicable to it [74]. Xiong et al. [96] address a robust scheduling for a flexible job-shop scheduling problem with random machine breakdowns. Makespan and robustness objectives are simultaneously optimized using a multi-objective evolutionary algorithm. There exist other approaches, such as the Hamta et al. study [45] where more objectives are optimized, such as cycle time, total equipment cost, and the smoothness index. This research deals with a single-model assembly line balancing problem, where the operation times of tasks are unknown variables, and the only known information is the lower and upper bounds for the operation time of each task. In this chapter, a type of machine scheduling problem is going to be presented as the focus of study. This research is related to the multi-objective grid scheduling problem and it is going to be explained in the following section.

3.3 Multi-objective Grid Scheduling Optimization

Job scheduling is a challenging task in grid environments, especially if many objectives need to be accomplished. In this research, we have presented this problem in a general way in order to optimize the execution time and the economic cost or energy consumption at the same time and with the same level of importance.

Execution time, as we have mentioned in the Related Work chapter, is the essential objective to be accomplished in scheduling problems, however for many distributed systems the economic cost has an important role for those systems and experiments that require financial models. Moreover, energy saving is becoming an important goal along with execution time objective for big and distributed systems, such as Grid Computing due to its high demand of power. Due to this, we present a generic model with two objective functions to optimize this pair of objectives.

In our case, we solve a multi-objective problem with two objectives (economic cost and execution time or energy consumption and response time). Given a set of jobs $J = |j_i|, i = \emptyset, .., m - 1$ and a set of resources $R = |r_z|, z = \emptyset, .., n - 1$ under the constraint that there are dependence relationships among jobs, the fitness functions are described as follows:

$$Min\ F = (F_1, F_2) \tag{3.6}$$

$$F_1 = \sum Economic\ Cost/Energy\ Consumption(j_i, f_z(j_i)) \tag{3.7}$$

$$F_2 = Maximum\ Time(j_i, f_z(j_i)) \tag{3.8}$$

where $f_z(j_i)$ is a mapping function that assigns job j_i onto resource r_z. The objective function F_1 returns the energy consumption or economic cost (depending on the problem to solve) for processing the experiment (set of jobs) and the objective function F_2 reports its completion time. Both objectives are minimized. Energy consumption or economic cost are the sum of energy consumption or economic cost for all the grid resources used executing the corresponding jobs. However, the execution time is calculated by considering the completion time which happens when the slowest job is executed.

Job scheduling is even more critical in terms of execution time when the jobs have dependencies between them. Dependent jobs influence on the total execu-

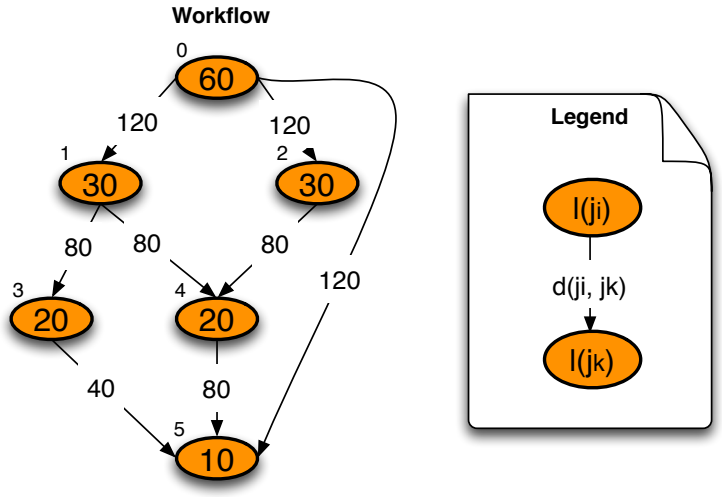

Figure 3.3.1: A simple workflow that follows a weighted directed acyclic graph (DAG) model.

tion time, because a workflow (or experiment) is modelled by a weighted directed acyclic graph (DAG) $JG = (V, E, l, d)$, where V is a set of nodes and E is a set of edges. Each node $j_i \in V$ represents a job and is assigned a constant length $l(j_i)$ representing its length in terms of Millions of Instructions (MI). Each edge $\langle j_i, j_k \rangle \in E$ $(j_i, j_k \in V)$ from j_i to j_k represents the precedence constraint, so that job j_k cannot be executed until job j_i has been completed successfully and j_k receives all necessary data from j_i. The length (bytes) of the data transfer $d(j_i \rightarrow j_k)$ between jobs is denoted by a label in the edge $j_i \rightarrow j_k$. In Figure 3.3.1 an example is shown.

Autonomous agents represent candidate solutions for the problem. In this work, the same data structure represents the agents for each multi-objective algorithm studied in this thesis. Each agent is a candidate solution and contains the following objects (see Figure 3.3.2):

1. The allocation vector a represents the assignment between jobs and resources. The length of the allocation vector is denoted by $|J|$, such that $a(i) = z$, where $\emptyset \leq i < |J|$ and $\emptyset \leq z < |R|$, that is job j_i is assigned to resource r_z. The same

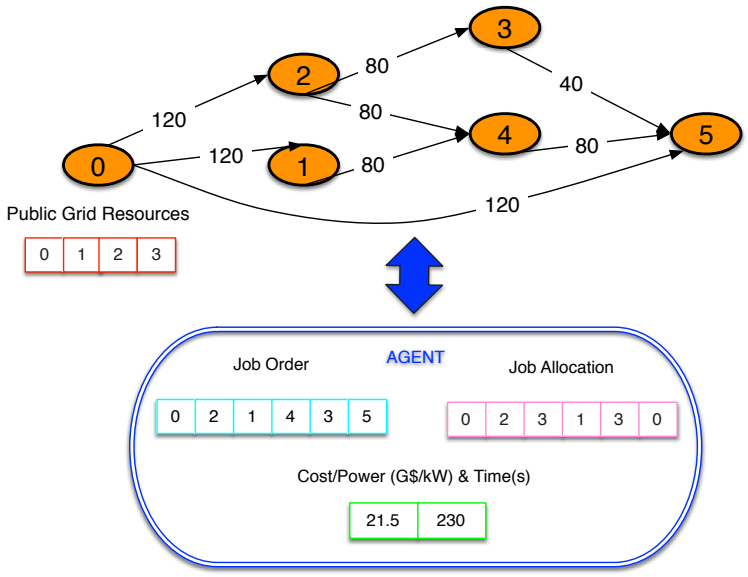

Figure 3.3.2: Agent representation as a candidate solution.

job in different resources could have different economic cost, energy consumption, and execution time. The symbol $|J|$ indicates the total number of jobs to allocate, and $|R|$ indicates the total number of available grid resources.

2. The order vector o indicates the execution order of the jobs. Different orders have different consequences for the execution time in the case of dependencies between jobs. The length of the order vector is expressed as $|J|$, such that $o(p) = i$, where $\not\emptyset \leq i; p < |J|$, and each job j_i appears once in the vector. This representation is based on the work [85].

3. Economic cost (measured in Grid dollars, G$) or Energy consumption (measured in kilowatts, kW) and Execution time (measured in seconds, s) are obtained while taking into account the two vectors mentioned. GridSim returns all these three different metrics.

"Look deep into nature, and then you will understand everything better".

Albert Einstein

4

Multi-Objective Optimization Algorithms

In this chapter, six multi-objective algorithms from different fields are explained and adapted to solve the grid scheduling problem.

4.1 Multi-Objective Gravitational Search Algorithm

Gravitational Search Algorithm [77] (GSA) is a swarm algorithm within the physics field, because its agents represent planets that have masses with different sizes exerting gravitational attractions among them through different dimensions. These attractions follow the Newtonian gravity law as metaheuristics. Thus, the biggest masses exert more force of attraction than those with less mass, positioning themselves as the best solutions. In this paper, a multi-objective version, called Multi-Objective Gravitational Search Algorithm (MOGSA), is presented to deal with the job scheduling problem in grid environments. In this new algorithm, the di-

mensions correspond to the combination of the two vectors: allocation and order vector. This combination is denoted as U_{A+O}. The gravitational forces are applied to each element of the vector U_{A+O}, and its size is the sum of the allocation and order vector size. MOGSA manages the agents considering the multi-objective context. The main steps of MOGSA are shown in Algorithm 1.

Algorithm 1 MOGSA pseudocode

INPUT: Population Size, G_o, Min_{Kbest}, α, ε
OUTPUT: Set of Solutions

```
1: Initialize population of solutions;
2: Evaluate population (Time and Cost/Energy consumption);
3: while not stop condition do
4:    Update Gravitational constant;
5:    Calculate size of masses;
6:    Calculate Force and Acceleration between masses;
7:    Update Velocity and Position per each mass;
8:    Select Set of Best Solutions (Pareto Front);
9:    Generate New Population;
10: end while
```

MOGSA has the same parameters of GSA [77].

- Population size is the number of agents that participate per iteration.
- G_o is the initial gravitational force that acts in each dimension of the agents that compose the population.
- Min_{Kbest} indicates the minimum of agents that exert their force over others. In GSA not all the agents exert their force over others, only the best ones, those that have bigger masses. Initially, the *Kbest* attribute has the same value as population size, and it decreases over the duration of the algorithm until it reaches the value of Min_{Kbest}, but always maintaining the best masses.
- α and ε are parameters used in equations 4.2 and 4.5 respectively.

The output returned by MOGSA is the Pareto front found after completing the algorithm. This Pareto front contains the best set of solutions found that minimize the fitness functions with the same level of importance.

4.1.1 Initialization and Evaluation

The algorithm starts randomly initiating the population. The allocation vector assigns randomly available grid resources to the jobs that compound the workflow. The initialization of the order vector is similar to the allocation vector randomly selecting the sent order for jobs, while considering the precedence constraints from the workflow. This initialization process is the same in the other algorithms implemented. However, to facilitate the management of the dimensional position for each agent, MOGSA joins the allocation and order vectors to form U_{A+O} in order to more quickly calculate the exerted force per dimension. Once the allocation and the order vectors are created per agent, GridSim returns the execution time and cost/energy consumption according to the vectors.

The evaluation begins using the ranking operator from NSGA-II [29]. This operator is implemented in MOGSA to classify the agents per Pareto fronts. This ranking operator assigns a rank per agent to the corresponding front. Then, a second operator, crowding distance, from NSGA-II is applied, in order to calculate the multi-objective fitness (*MOFitness*) per agent. This value is calculated by equation 4.1.

$$MOFitness(X_i) = (2^{(X_i.r)} + \frac{1}{1 + X_i.cd})^{-1} \tag{4.1}$$

where X_i denotes the agent *i*, *r* indicates the rank of the Pareto front, and *cd* represents the crowding distance. MOFitness is used to sort the agents according to quality. Agents with less MOFitness represent better solutions.

4.1.2 Update Gravitational Constant

Gravitational constant G is initialized at the beginning with the value of G_o, but it is reduced as the time goes by, in order to control the search accuracy (see equation

4.2).

$$G = G_o \exp^{(-a)\frac{t}{T}} \tag{4.2}$$

where t is the current time, T is the total time, and a is a parameter used to measure the reduction of G.

4.1.3 Calculate size of masses

Heavier masses mean more efficient agents, so better agents exert higher attractions and walk more slowly. The size of masses is calculated considering their fitness and they are updated following equations 4.3 and 4.4.

$$X_i.q = \frac{MOFitness_{worst} - X_i.MOFitness}{MOFitness_{worst} - MOFitness_{best}} \tag{4.3}$$

$$X_i.m = \frac{X_i.q}{\sum_1^N X_j.q} \tag{4.4}$$

where $MOFitness_{worst}$ and $MOFitness_{best}$ are the highest and lowest values of *MOFitness*, respectively and N is the population size.

4.1.4 Calculate Force and Acceleration between masses

Acceleration is caused by the exerted force between masses in all these dimensions. Therefore, MOGSA calculates the exerted force from the *Kbest* agents on the rest of the population for each dimension. That means, only *Kbest* agents act on the whole population. At first, *Kbest* has the same value of the population size, but *Kbest* is decreasing as the time goes by until Min_{Kbest} is reached. To calculate the corresponding force (see equation 4.5) per each pair of agents a Euclidean distance is calculated following equation 4.6. Euclidean distance is carried out from the U_{A+O} vectors of the pair of agents. Using in the first part of the vector (the allocation vector), the distance between the resource numbers and the second part

(the order vector) is the distance between the order positions.

$$X_i.F_j^d = G \times \frac{X_i.m \times X_j.m}{R_{i,j} + \varepsilon} \times (X_j.U_{A+O}^d - X_i.U_{A+O}^d) \tag{4.5}$$

$$R_{i,j} = \|X_i, X_j\|, \forall i, 1 \leq i \leq N; \forall j, 1 \leq j \leq K_{best} \tag{4.6}$$

The total force that acts on Xi is the sum of a random weight in all its dimensions (equation 4.7). MOGSA applies this stochastic feature to consider the exploration process. In swarm algorithms, exploration and exploitation processes are required to avoid local optima and achieve optimal solutions.

$$X_i.f^d = \sum_{j \in K_{best}, j \neq i}^{N} rand\,[0,1] \times X_i.F_j^d \tag{4.7}$$

Agents obtain their acceleration taking into account the total mass and the forces that are exerted in all dimensions (see equation 4.8).

$$X_i.a^d = \frac{X_i.f^d}{X_i.m} \tag{4.8}$$

4.1.5 Update Velocity and Position per each mass

The acceleration provokes the update in the velocity and the position of the agents. However, during the calculation of the velocity not only does acceleration affects the update, but a random number is applied to improve the exploration process (see equations 4.9 and 4.10).

$$X_i.v^d = rand\,[0,1] \times X_i.v^d + X_i.a^d \tag{4.9}$$

$$X_i.U_{A+O}^d = X_i.U_{A+O}^d + X_i.v^d \tag{4.10}$$

The new agent position is updated, modifying the values of the U_{A+O} vector, the identifier of the resource in the case of the allocation vector, or the number of the

position in the order vector. In case the last vector, the precedence constraint is considered.

4.1.6 Select Set of Best Solutions (Pareto Front)

The current best solutions are calculated from the ranking operator that classifies the population per Pareto fronts. MOGSA has been improved to avoid early stagnations, so a stagnation evaluation is implemented on the new population. When stagnation occurs along several iterations, a mutation process is applied to the agents that are not changed regarding the previous population. This mutation process is subdivided in two types of mutations for each vector (allocation and order). These mutations use heuristics of the job scheduling problem to do a local search. Both, allocation and order vectors are randomly generated and heuristics are applied. The random order is compared with other built from a greedy algorithm. The greedy algorithm is based on the precedence constraint of the workflow. The method consists of first executing the parent jobs with more dependent jobs. The mutation process for the allocation vector is more complex, because more heuristics are considered, such as the following: order of resources according to speed/cost (or energy consumption), time per job in its specific resource, overhead time (prediction of execution for the entire workflow), and wait time caused by the precedence between jobs.

Finally, the resulting best Pareto front will be saved as a set of best solutions, and MOGSA restarts a new generation until the time limited is expired.

4.2 Multi-Objective Artificial Bee Colony

Artificial Bee Colony (ABC) ([50], [49]) is a single-objective swarm algorithm within the biological field. This algorithm is based on the collective behaviour of its agents, bees, to find the best nectar from the flowers. The main feature of the ABC algorithm is that its agents have different behaviours. Some bees move in a multidimensional search space by selecting nectar sources considering their last experience and the experience of their fellow hive members. However, other

bees move randomly without regard to experience and influence. When they find a better nectar source (flower position), they memorize it and forget the previous flower position. That means, ABC also combines exploration and exploitation processes with an effort to equilibrate them. These processes are collected from the behaviour of three kinds of bees: employed, onlooker, and scout bees.

- Onlookers are waiting in the work area to select a good flower, previously chosen by employed bees.
- Employed bees initially go to the flowers and dance according to the quality of the nectar found. Onlookers choose them according to best dance.
- Scouts are those that perform random scans to find other flowers, usually based on heuristics.

In this research, a multi-objective version, called Multi-Objective Artificial Bee Colony (MOABC), is implemented and adapted to solve the job scheduling problem in grid environments. The fundamental steps of the MOABC are shown in Algorithm 2.

Algorithm 2 MOABC pseudocode

INPUT: Population Size, Mutation Probability
OUTPUT: Set of Solutions

```
1: Initialize population of solutions;
2: Evaluate population (Time and Cost/Energy consumption);
3: while not stop condition do
4:     Multi-Objective Exploitation Process (Employed and Onlooker Bees);
5:     Multi-Objective Exploration Process (Scout Bees);
6:     Select Set of Best Solutions (Pareto Front);
7:     Generate New Population;
8: end while
```

MOABC manages the same parameters as the ABC algorithm ([50], [49]). In particular, this swarm algorithm stands out for its simplicity with a small number of parameters.

- Population size indicates the number of agents, bees, per iteration.
- Mutation probability is used in the mutation process during the algorithm execution.

The output of the MOABC algorithm is composed of a set of solutions, bees, obtained by calculation of the Pareto front after completing the algorithm execution.

4.2.1 Initialization and Evaluation of the population

The initialization of the population is similar to MOGSA. However, the population begins considering only half of the population size given by the parameters. This half is for creating the employed bees. During the algorithm execution, new bees (onlookers and scouts) are added until reaching the total population size. These first employed bees are also comprised of the allocation and order vectors, which are randomly created, taking into account the workflow precedence constraint. GridSim provides their execution time and cost/energy consumption according to the vectors. Once employed bees are created, they are classified per Pareto fronts using the ranking operator from the NSGA-II algorithm, and crowding distance is calculated to evaluate the quality of the solutions found.

4.2.2 Multi-Objective Exploitation Process

In the multi-objective exploitation process, employed and onlooker bees search for the best solutions based on their last experience and the experience of their fellow bees. This process begins generating neighbour bees from the employed bees. This step uses two types of mutations, one per vector (allocation and order) to generate a neighbour bee.

- The order mutation is in charge of modifying the order vector while considering the DAG model from the workflow. First of all, each job is selected while taking into account the mutation probability. The process identifies the last position of the order vector that is occupied by one parent of the job

to mutate. After that, the positions of the child jobs are searched in the order vector to select the first position. Finally, the process randomly chooses a new position for the job to mutate among the last parent position and the first child position.

- The allocation mutation provides a neighbour allocation vector using the mutation probability. The jobs selected to mutate randomly choose a resource identification number from the list of available grid resources.

Once employed and neighbour bees are created, they compete to be selected for the next generation. This competition uses two operators from the standard NSGA-II. First, the ranking operator is applied to calculate the rank for each bee. The bee with best rank is the winner. In case of the bees have the same rank, the crowding distance operator is used to break the tie. The bee with more crowding distance is the winner, and replaces the other bee.

All the winner bees are ordered according to their solution quality using the ranking and crowding distance operators. A method of roulette selection is used to generate these new bees following the bee probability equation 4.11 (that is, using a linear bias [75]).

$$bee.pb = \frac{\frac{1}{bee.p}}{\sum_i \frac{1}{bee_i.p}} \tag{4.11}$$

Once onlookers are selected by this process, the neighbours generation is applied to these bees in the same way as the employed bees, with winner bees replacing the loser bees.

4.2.3 Multi-Objective Exploration Process

The multi-objective exploration process is executed by a new kind of bee, the scout bee, whose behavior mimics the heuristics of the job scheduling problem. In this problem, one scout bee is enough to execute this process. The scout bee randomly generates its vectors, allocation and order, and then modifies them using information from the problem without the experience of the hive members. For each

vector a different process is performed considering different heuristics. These processes are similar to those mentioned in the stagnation mutation in the MOGSA.

- Heuristic generation for the order vector begins generating a random order vector according to the precedence constraint as processed in the population initialization. At the same time, another order vector is generated using a greedy algorithm that takes into consideration that jobs with more dependent jobs must be executed first. The random order is modified in effort to achieve the greedy order without violating the job dependencies.

- Heuristic generation for the allocation vector begins generating a random allocation vector. Then, the algorithm applies heuristics for the job processing time $\frac{MI}{MIPS}$, where *MI* denotes the job length and *MIPS* is the speed of the assigned resource. The duration of the workflow execution is calculated according to the dependencies between jobs. This process assumes that jobs with no dependencies between them can be executed simultaneosly. After that, each job is assigned to a resource that reduces the current total execution time. The available grid resources are sorted according to the processing speed/cost (or energy consumption) value. The overhead time is also considered by the competing jobs (jobs without dependencies on each other that are allocated to the same resource).

4.2.4 Select Set of Best Solutions

This set of solutions is selected from the set of all types of bees. All the bees (employed, onlooker, and scout bees) are ranked, and the bees that comprise the best Pareto front are the best solutions for the current iteration. This new, best Pareto front is saved and will be compared with the Pareto front from the next iteration.

Finally, a new population is selected by the ranking operator, saving the number of employed bees from the first fronts. Therefore, if the employed number is 50, the first 50 bees from the best Pareto fronts are chosen.

4.3 Multi-Objective Small World Optimization

Multi-Objective Small World Optimization (MOSWO) is a multi-objective approach based on the small world phenomenon in the sociological field. Swarm meta-heuristics in the context of complex networks date back to Milgram's pioneer experiment, which originated the famous term "six degrees of separation" [71]. More recently, Watts and Strogatz ([91][83]) reported a mathematical model for a network attaining the small world (SW) effect and bridged the SW algorithms to the computer science community. Small-world phenomenon soon became an active field of research in complex systems and related problems due to its inter(trans)-disciplinary nature, combining sociology, physics, biology, mathematics, and computer science ([33], [90], [64]).

The cornerstone of the SW phenomenon is the exploitation of local connections combined with a few long-range links to find the shortest paths in a network. This approach has been shown to offer an efficient searching strategy [56]. The MOSWO algorithm implements some practices from the Tabu Small World Optimization (TSWO) [70] algorithm, but in our case, it is used to optimize more than one objective. The TSWO algorithm is also based on the Small World Algorithm (SWA) [33] by using a Tabu search for the local search operator. The solution space is represented as a small-world network, and the optimization process attempts to find the shortest path from a candidate solution to an optimal solution. The TSWO algorithm demonstrated better performance than SWA due to its decimal encoding in contrast to the binary enconding used by SWA. Moreover, both single-objective algorithms have been compared with genetic algorithms (GAs) ([33], [64], [70]), providing better results than the last ones. In [33], seven two-dimensional functions have been evaluated by SWA and GA from 50 simulations. The measurements considered the time to reach the optimal value and the maximum, minimum, average, and standard deviation of the optimal value, respectively. Results show that SWA has a stronger ability to break away from the local optimum than GA. One hundred percent of simulations performed by SWA achieve the optimal solution in all the functions. Meanwhile, GA achieves 68.85%

successful optimizations. The standard deviation of SWA is approximately 50% less (in all the functions) than the GA standard deviation, which indicates that SWA has better robustness and stability than GA. Moreover, a decimal-coding version of SWA (DSWOA) is compared with the orthogonal genetic algorithm with quantization (OGA/Q) to solve high-dimensional functions [64]. The simulation results of several benchmark functions show that SWA can locate beneficial solutions, stabilizes well, and converges at a fast rate. Work carried out by [70] demonstrates that TSWO locates the global optimum in each of 30 runs performed by six two-dimensional modal functions considered also in [33]. In contrast, the DGA (Decimal-coding Genetic Algorithm) tends to be trapped in local optima. In the same work, TSWO is also compared with PSO (Particle Swarm Optimization): TSWO performs better than PSO and DGA on the average value and standard error of 30 runs. The MOSWO algorithm applies the Tabu search as a local search operator with a multi-objective perspective, following the efficiency of the single-objective algorithm, TSWO. The main steps of MOSWO are shown in Algorithm 3.

Algorithm 3 MOSWO pseudocode

INPUT: Population Size, Mutation Probability
OUTPUT: Set of Solutions

```
Initialization and Evaluation (Time and Cost/Energy consumption);
while not stop condition do
    Multi-Objective Random Long-Range Operator (Γ);
    Multi-Objective Local Shortcuts Search Operator (Ψ);
end while
Stagnation and Best Solutions processes;
```

The required parameters are defined as follows:

- Population Size indicates the number of nodes that are included in the next iteration.
- Mutation Probability (Pmutation) provides the probability to indicate which

nodes are going to be modified by using the Multi-Objective Random Long-Range Operator (Γ) .

The output is a set of solutions, typical in multi-objective algorithms, and it comes from the best Pareto front found until the last iteration of MOSWO.

4.3.1 Initialization and Evaluation

A random process is carried out to perform the initialization taking into account the dependencies between jobs. This initialization is applied to both vectors, the allocation and order vectors, which are generated from different processes. The allocation generation method assigns random resources to the jobs that make up the workflow; many jobs could be mapped to the same resource. The order vector is generated with a similar process in which a random job is assigned to an order position fulfilling the dependencies between jobs. The union of these vectors is denoted as U_{A+O}, which represents the coordinates of each node. Once both vectors are generated, GridSim provides cost/energy consumption and execution time values accordingly. After that, a ranking process is applied following the Pareto dominance concept.

4.3.2 Multi-Objective Random Long-Range Operator (Γ)

Multi-Objective Random Long-Range Operator (Γ) is applied to each node N_i, that does not appear in the set of best solutions (best Pareto front), following the mutation probability. This operator consists of the execution of two similar processes to modify both vectors. In the case of the allocation vector, μ and v jobs that were assigned previously to a resource are selected randomly, $1 \leq \mu < v \leq m$, where m is the number of jobs that belong to the workflow. Hence, it is also the length of the allocation and order vectors. Then, the resources assigned to μ and v jobs are swapped. The process for the order vector is similar, although, before applying the swap between order positions assigned to the jobs, a checking process is applied to avoid the dependency fault between jobs. Finally, a new node is created N_i' after this Γ operator.

4.3.3 Multi-Objective Local Shortcuts Search Operator (Ψ)

Multi-Objective Local Shortcuts Search Operator (Ψ) is executed to obtain the best set of neighbours, *Best* ($\xi(N_i')$), by using the Tabu search. Tabu search updates a list, called the Tabu list, following the *Least Recent Used* (*LRU*) strategy. The LRU strategy stores the studied nodes to avoid the repetition of their study. The Ψ operator starts by checking if the node N_i' is in the Tabu list before generating its neighbours. After that, it generates a set of neighbours by N_i' by modifying the allocation and order vectors. Due to performance issues, the algorithm generates a limited number of neighbours; this number is equal to the total number of jobs. On one hand, the allocation vector is crossed and the resource assignment is modified sequentially. This assignment process increases the resource identification by one. On the other hand, the order vector uses the same technique as the allocation vector, but in respect to the dependency constraint between jobs. Once the neighbours are generated, the Ψ operator obtains the best Pareto Front from the neighbours set from a ranking process that applies the Pareto dominance concept in the same way as in the initialization (subsection 4.3.1) . Then, the front is crossed in order to check whether its nodes are in the Tabu list. If one node is in the list, it is deleted from the front. Otherwise, it is included in the Tabu list in order to remove it from further study.

4.3.4 Stagnation and Best Solutions processes

Finally, the resulting front is built from all first fronts calculated per node in the current iteration. These fronts are joined with the current population. The Pareto front ranking as well as the crowding distance operator sort this new node set. Next, an improvement is applied to this algorithm in order to add further diversity and avoid stagnation. The least optimal node of the set is modified by using heuristics from the problem. Two different heuristics are applied per vector for each node. The heuristic for the order vector consists of modifying the vector by comparing it with another order vector calculated from a greedy algorithm. This greedy algorithm provides the best order of execution for the workflow according

to the number of dependencies. Top positions of this vector are assigned to the jobs that have more dependent jobs without violating the precedence constraint. The allocation vector heuristic calculates its processing time *MI/MIPS* for each job, where *MI* denotes the job length (in instructions) and *MIPS* indicates the speed of the job execution in the resource assigned in the allocation vector. The total time execution of the workflow is calculated with consideration for the dependencies between jobs, assuming those that have no dependencies between them can be executed simultaneously. Therefore, each job is assigned to the resource that reduces the total execution time previously calculated. Moreover, resources are sorted following the value of *processing speed/cost (or energy consumption)*. The overhead time is also taken into account by the competing jobs (without dependencies between them) that run sequentially in the same resource. Before finalizing the iteration, the population is reduced to its previous size by choosing the best nodes that compose the set (old population and neighbours) to obtain the new population and the set of best nodes (*Best Pareto Front*).

4.4 Multi-Objective Firefly Algorithm

The Firefly Algorithm (FA) [97] is a single-objective swarm algorithm based on the fireflies' behaviour and therefore, is within the framework of biology. Their brightness represents the value of the fitness function and is proportional to the attractiveness to each other. Therefore, for any two flashing fireflies, the less bright one will move towards the brighter one. The multi-objective approach (MO-FA) has to consider more than one fitness function. For the minimization problem the brightness attribute would be inversely proportional to the fitness functions. The MO-FA algorithm is described in Algorithm 4.

This algorithm requires the following four parameters: population size, β, γ, and α. The last three parameters are used in equation 4.13 to diversify the optimization process.

- Population size denotes the number of fireflies that are going to be studied

Algorithm 4 MO-FA pseudocode

INPUT: Population Size, β, γ, α
OUTPUT: Set of Solutions

```
1: Initialization and Evaluation (Time and Cost/Energy consumption);
2: while not stop condition do
3:     Multi-Objective Comparison among fireflies;
4:     Multi-Objective firefly attraction;
5: end while
6: Stagnation and Best Solutions processes;
```

per each iteration.

- The initial attractiveness between two fireflies is defined by β.
- The coefficient of light absorption is indicated by γ and represents the light absorbed from the air.
- The variable α is a randomization parameter to give more diversity to the algorithm.

The output is a set of solutions as it is a multi-objective algorithm.

4.4.1 Initialization and Evaluation

The algorithm starts with a random initialization of the fireflies' population taking into account the dependencies between jobs. When the fireflies are generated, GridSim returns the values of energy consumption in watts (or cost in G$) and the execution time in seconds. This generation directly affects the union of both vectors, the allocation and order vectors, which denotes the dimensional position of the fireflies. The union of these vectors is U_{A+O}. After this initialization, a ranking is applied for each firefly following the Pareto dominance concept in the same method carried out in MOSWO.

4.4.2 Multi-Objective Comparison among fireflies

Each firefly is compared with the others in order to detect if there is any firefly with more brightness. Those fireflies with a higher brightness value, which is denoted by the fitness function, are considered good candidate solutions. This comparison is carried out for each pair of fireflies according to the dominance concept. Those fireflies located in different Pareto fronts have different brightness and, therefore one that has less brightness will be attracted to the one with more brightness (Multi-Objective firefly attraction). In the case of the pair of fireflies is in the same Pareto front, they will not be attracted to each other.

4.4.3 Multi-Objective firefly attraction

The dominated firefly will move toward the firefly with better fitness values. The firefly attraction process first needs the calculation of the Euclidean distance. The Euclidean distance is the distance between the allocation vectors (resource distance) of each firefly and the difference between their order vectors (order positions)[1] (equation 4.12).

$$r_{i,j} = \|cs_i - cs_j\| = \sqrt{\sum_{k=1}^{dim}(cs_{i,k} - cs_{j,k})^2} \tag{4.12}$$

The movement of a firefly cs_i attracted to another, more attractive (brighter) firefly cs_j is determined by equation 4.13 (applied to every dimension of the fireflies).

$$cs_i = cs_i + \beta e^{-\gamma r_{i,j}^2}(cs_j - cs_i) + \alpha(rand - \frac{1}{2}) \tag{4.13}$$

where β indicates the attractiveness for $r = \phi$, γ is the coefficient of light absorption, and α is the randomization parameter, because *rand* denotes a random number between ϕ and 1. This stochastic feature applies the exploration processes for this algorithm. In swarm algorithms, exploration and exploitation processes are

[1] Notice *cs* refers to a candidate solution or firefly.

required to avoid local optima and achieve good solutions.

4.4.4 Stagnation and Best Solutions processes

Finally, a stagnation checking method is applied to the new firefly population. This method is an improvement with respect to the original FA in order to avoid the population stagnation over several iterations. Two mutation methods (one per vector) are applied to the stagnated fireflies. Both mutations consider heuristics from the job scheduling problem carrying out a local search. The order and allocation vectors are randomly generated but each one has its own heuristics. The random order vector is compared with other order vector built from a greedy algorithm. The greedy algorithm consists of the creation of an order vector in which the first positions are assigned for the jobs that have more dependent jobs. This fact implies that these jobs will be executed first. The mutation for the allocation vector is more complex, since it considers more heuristics, such as the first selection of the resources according to their speed/cost (or energy consumption), time per job in the resource selected, wait time based on the dependencies between jobs, and overhead time to predict the total execution time of the workflow. The new population is processed again until the stop condition is met. The fireflies are ranked, and the *best Pareto front* is extracted from the final population resulting in the set of solutions for the problem.

4.5 Multi-Objective Brain Storm Algorithm

Multi-Objective Brain Storm Algorithm (MOBSA) is based on a novel single-objective algorithm called Brain Storm Optimization (BSO) ([79], [80]) which belongs to the sociological field algorithm classification. The BSO algorithm is inspired by human behaviour when a brainstorming process is applied in a group, especially with different backgrounds, in order to resolve a complex problem. This process is widely applied in different fields, and it has been proven that for several problems that cannot be solved by a single person, when a group of people carry out a brainstorming process, the problem is usually solved with high probability. An

un-expectable intelligence can occur from this interactive collaboration of human beings. The MOBSA algorithm adds multi-objective properties to BSO algorithm to optimize more than one objective. It is applied to the grid scheduling problem. The main steps of MOBSA are presented in Algorithm 5.

Algorithm 5 MOBSA pseudocode

INPUT: Population Size (n), Cluster Number (m), $P_{center-replacement}$, $P_{cluster-mutation}$, $P_{center-mutation}$, $P_{cell-mutation}$, $P_{center-cross}$
OUTPUT: Set of Solutions

```
1: Initialize population of solutions;
2: Evaluate population (Time and Cost/Energy consumption);
3: while not stop condition do
4:    Multi-Objective ideas ranking;
5:    Random cluster replacement;
6:    while not n ideas generation do
7:       Multi-Objective ideas generation;
8:    end while
9: end while
10: Best solutions selection;
```

This algorithm has less parameters than the BSO approach because heuristic mutation and crossover processes are implemented instead of the random Gaussian process proposed by Shi ([79], [80]). The required parameters are as follows:

- Population Size (n) indicates the total of ideas that are considered per iteration.
- Cluster Number (m) represents the number of groups of ideas that are similar. Each cluster of ideas has a center idea that represents one of the best ideas from this cluster.
- $P_{center-replacement}$ is the probability used to replace a cluster center from other randomly generated.
- $P_{cluster-mutation}$ is the probability of selecting the cluster where the mutation-crossover process is carried out.

- $P_{center-mutation}$ is the probability of mutating a center or other idea in the same cluster.
- $P_{cell-mutation}$ is the probability of mutating one cell per each vector, allocation or order vector.
- $P_{center-cross}$ is the probability of crossing two centers or two other ideas from two random clusters.

The output returned by MOBSA is the best Pareto front obtained until achieving the stop condition. This Pareto front contains the best set of solutions found that minimize the fitness functions. The main steps of MOBSA are explained in the following subsections.

4.5.1 Initialization and Evaluation

A random initialization is generated. The initialization process consists of two different methods of generation according to the features of allocation and order vectors that make up a new idea. The allocation generation method assigns a random resource per job that comprises execution of the workflow; many jobs could be mapped to the same resource. On the other hand, the order generation method indicates a random job per each order position and it has to consider an order constraint in the case of workflows with dependencies between jobs. Then, the resulting idea with the corresponding vectors (allocation and order) is executed in GridSim in order to obtain both fitness values: energy consumption (watts) or cost (G$), and execution time (seconds). Once all the ideas are generated, a ranking process is applied following the Pareto dominance concept.

4.5.2 Multi-Objective ideas ranking

This step serves the purpose of the brainstorming process when some (e.g. 3 or 5) clients act as the owners of the problem to pick up several ideas (e.g. one per each owner) as better ideas for solving the problem. The number of owners is indicated

by the m parameter that represents the number of clusters. Then, m ideas are selected from the best Pareto fronts to be considered as centers or better ideas picked up by problem owners. The rest of the ideas are clustered, taking into account their similarity with the centers by calculating their Euclidean Distance (Equation 4.14).

$$ED_{i,j} = \|X_i, X_j\|, \forall i, 1 \leq i \leq m; \forall j, 1 \leq j \leq n - m \tag{4.14}$$

Ideas with more similarity are located in the same cluster. Center ideas usually have more chances to be used to generate new ideas than other ideas from the same cluster.

4.5.3 Random cluster replacement

This step enables the population to cover unexplored areas. It is similar to one of the steps from the real brainstorming process when a random object is picked up and the functions and appearance of the object are used as clues to generate more ideas. The MOBSA uses the probability $P_{center-replacement}$ to decide if a random cluster center is replaced by a newly generated idea. The new idea is generated with the same methods used by the initialization.

4.5.4 Multi-Objective ideas generation

This process simulates idea generation and is divided in two subprocesses: a Multi-Objective mutation process and a Multi-Objective crossover process. The generation of new ideas will be executed until n ideas are generated. If new ideas are generated, but are finally discarded, they still count as generated ideas.

Multi-Objective mutation process

The Multi-Objective mutation process simulates new idea generation inspired from a single existing idea and can be applied to a random cluster. An idea from this cluster will be mutated according to the probability indicated by $P_{cluster-mutation}$. Then, if the mutation process is selected, a center idea is mutated according to the

probability $P_{center-mutation}$, otherwise another idea from the same cluster is mutated. Two heuristic and multi-objective mutation processes are implemented in order to manage the mutation of both allocation and order vectors.

- The order mutation modifies the order vector taking into consideration the job dependency constraint. Each job is selected while taking into account the mutation probability $P_{cell-mutation}$. The process identifies the last position that is occupied by one parent of the job to mutate. Then, the positions of the child jobs are searched in the order vector to select the first position. Finally, the process randomly chooses a new position for the job to mutate among the last parent position and the first child position.

- The allocation mutation provides a new allocation vector using the mutation probability $P_{cell-mutation}$. The selected jobs are assigned to random grid resources from the list of available resources.

The generated idea is evaluated by obtaining its energy consumption (or cost) and execution time values in order to compare it with the old idea. If they belongs to different Pareto fronts, the dominated idea is discarded, otherwise the crowding distance for each idea is calculated, and the idea with less value is discarded. Crowding distance operator is used to break ties in the standard NSGA-II [29].

Multi-Objective crossover process

This process simulates the new idea generation inspired from two existing ideas from two different idea clusters. Two cluster centers have the probability $P_{center-cross}$ to be selected for the crossover process, otherwise, two other ideas (not centers) from two different clusters are crossed generating two new ideas. This is the same as the mutation process; two crossover processes are implemented for allocation and order vectors.

- The allocation crossover randomly selects a position from the allocation vector, which indicates the identifier of the job. Parent vectors swap their vectors from the random position, creating two new individuals.

- The order crossover is similar to the allocation vector crossover but considers the dependencies between jobs. After the swapping process, the dependencies and possible repeated order positions are checked. To avoid this during the crossover, after swapping this method checks whether there is not an order position in the order vector. If it occurs, this position is stored, and when there is a repeated position, one stored position is randomly selected.

Finally, the two generated ideas are evaluated and compared with their parent ideas using the same methods as in the multi-objective mutation process. Only the two worst ideas are discarded.

4.5.5 Stagnation checking

In every iteration, when the set of best solutions is selected from a ranking process by applying the dominance concept, a stagnation checking process is performed in the new population. This method is an improvement with respect to the original BSO in order to avoid population stagnation during several iterations. Two heuristic methods (one per each vector) are applied to the worst idea. Both heuristic methods consider heuristics from the job scheduling problem by carrying out a local search. The order and allocation vectors are randomly initialized.

On one hand, the random order vector is compared with other order vector built from a greedy algorithm. This greedy algorithm consists of the creation of an order vector in which the first positions are assigned to the jobs that have more dependent jobs. This fact implies that these jobs will be executed first. On the other hand, the allocation heuristic method is more complex due to the consideration of more heuristics, such as the first selection of the resources according to their speed/cost (energy consumption), duration of job in the resource selected, wait time taking into account the dependencies between jobs, and the overhead time to predict the total workflow execution time.

4.6 Non-dominated Sorting Genetic Algorithm II

Non-dominated Sorting Genetic Algorithm II (NSGA-II) [29] is the most popular multi-objective genetic algorithm and is widely known due to its efficiency. In this research, this algorithm is applied to the job scheduling problem to prove the multi-objective quality of the proposed swarm algorithms. As a genetic algorithm, it has agents as individuals that comprise the evolutionary population. The main steps of NSGA-II are shown in Algorithm 6. The NSGA-II input has three parameters:

Algorithm 6 NSGA-II pseudocode

INPUT: Size of population, Number of Iterations or Maximum time of execution, Crossover and Mutation Probability
OUTPUT: Set of Solutions

```
1: Initialize population of solutions;
2: Evaluate population (Time and Cost/Energy consumption);
3: while not stop condition do
4:     Binary Tournament Selection;
5:     Crossover;
6:     Mutation;
7:     Select Set of Best Solutions (Pareto Front);
8:     Select New Generation;
9: end while
```

- Population size indicates the number of individuals that take part in the optimization.
- Crossover probability is the probability of interchanging the gens (allocation or order vector elements) to crossover with other individuals.
- Mutation probability is the probability of mutating the gens of an individual.

The output is the same as the previous algorithms; it is the set of best solutions found during the execution of the algorithm comprising a Pareto front.

4.6.1 Initialization and evaluation of the population

The initialization of the population follows the same process as the other algorithms. Population size indicates the number of individuals, and these individuals randomly create their allocation and order vectors with respect to the precedence constraint from the workflow execution. Moreover, GridSim provides the execution time and cost/energy consumption for each individual taking into account its vectors. After that, the evaluation is done by the NSGA-II operators, Ranking of Pareto fronts and crowding distance. The next processes, tournament selection, crossover, and mutation are in charge of creating the offspring population of the same size as the initial population.

4.6.2 Global optimum search

In NSGA-II the descendant population Q (size N) is created from the parent population P (size N). After that, these two populations are combined forming a new population R with size 2N. The population R is classified by a non-dominated sorting in different Pareto fronts. This allows a global verification of dominance between the parent and descendant populations. When the process of non-dominated sorting is finalized, the new population is generated from the non-dominated Pareto fronts. This new population starts its building from the best non-dominated front (F1) followed by the second Pareto front (F2) and so on. As the size of population R is 2N and the size of the origin population is N, not all the solutions from R will be part of the new population. Those fronts that cannot be added to the new population will be discarded. When the last front is being considered, the solutions that belong to it can exceed the size of the population, and then the crowding distance is applied to these solutions. The crowding distance permits the solutions selection located in the least-crowded areas (far away from other solutions) to complete the new population instead of randomly choosing the solutions. This process is not relevant for the first iterations of this algorithm, because many fronts survive to the next iteration. However, when the process advances, many solutions are located in the first front and second front, and they could have more than N solu-

tions. This fact makes the diversity and selection of quality solutions important. The idea is to always promote those solutions that ensure more diversity from the same Pareto front. When the population converges to the optimum Pareto front (the first one), the algorithm ensures that the solutions are not close to each other and that the global optimum search objective is achieved.

4.6.3 Binary tournament selection

Binary tournament selection takes care of choosing the parent individuals to cross. This method sorts the individuals based on non-domination with crowding distance assigned; the selection is carried out using a crowded-comparison-operator ($\prec n$).

- Non-domination rank for individual p indicates that the individual is in front F_i, so if p is in F_i its rank will be $p_{rank} = i$.
- Crowding distance $F_i(d_j)$: In this case, $p \prec n$ q if
  - $p_{rank} < q_{rank}$
  - Or if p and q belong to the same front F_i, then $F_i(d_p) > F_i(d_q)$ that is, the crowing distance should be greater for p.

The individuals are selected by using a binary tournament selection with a crowded-comparison-operator.

4.6.4 Crossover

The crossover process is divided in two crossover (one per vector, allocation and order vector). These two crossover processes are based on Talukder's work [85]:

- The allocation crossover randomly selects a position from the allocation vector, which indicates the identifier of the job. Then, the parent vectors swap their vectors from the random position, creating two new individuals.

- The order crossover is similar to the allocation vector crossover, but considers the precedence constraint. Therefore, after the swapping process, this method checks the dependencies and possible repeated order positions. To avoid this during the crossover, before swapping, this method checks whether there is not a position in the order vector. If it occurs, this position is stored and when there is a repeated position, one stored position is randomly selected.

These new individuals are evaluated by obtaining their execution time and cost/energy consumption and are added to the population set. Population size is set to $2N$.

4.6.5 Mutation

The mutation process is divided in two mutation processes for the two vectors. These mutations are the same as the ones described in the MOABC algorithm in Section 4.2. Execution time and cost/energy consumption are calculated for the new individuals.

4.6.6 Select set of best solutions

The set of best solutions is calculated by the ranking operator from the total set of individuals. The best Pareto front from the current iteration will be compared with the previous iteration set following the same process as the previous algorithms.

Finally, the new population is formed by choosing the individuals that comprise the best Pareto fronts until completing the new population indicated by the population size parameter.

"Be a yardstick of quality. Some people aren't used to an environment where excellence is expected".

Steve Jobs

5 Experimental Environment

In this chapter, we describe in detail the test environments used with the aim to make the experiments reproducible for future research. The whole experimental environment presented in this thesis (topologies, workflows, and algorithms) is implemented in the GridSim simulator with a complete configuration.

5.1 Scientific Workflows

A variety of workflows based on specific numerical computational problems such as Parallel Gaussian Algorithm, Parallel Gauss-Jordan Algorithm, Parallel LU decomposition [88], Find-Max Algorithm [94], Fast Fourier Transform (FFT) and Stencil [19], are tested. All of them follow a DAG model (Figure 5.1.1), and their complex structure allows a good study of the behaviour of our algorithms.

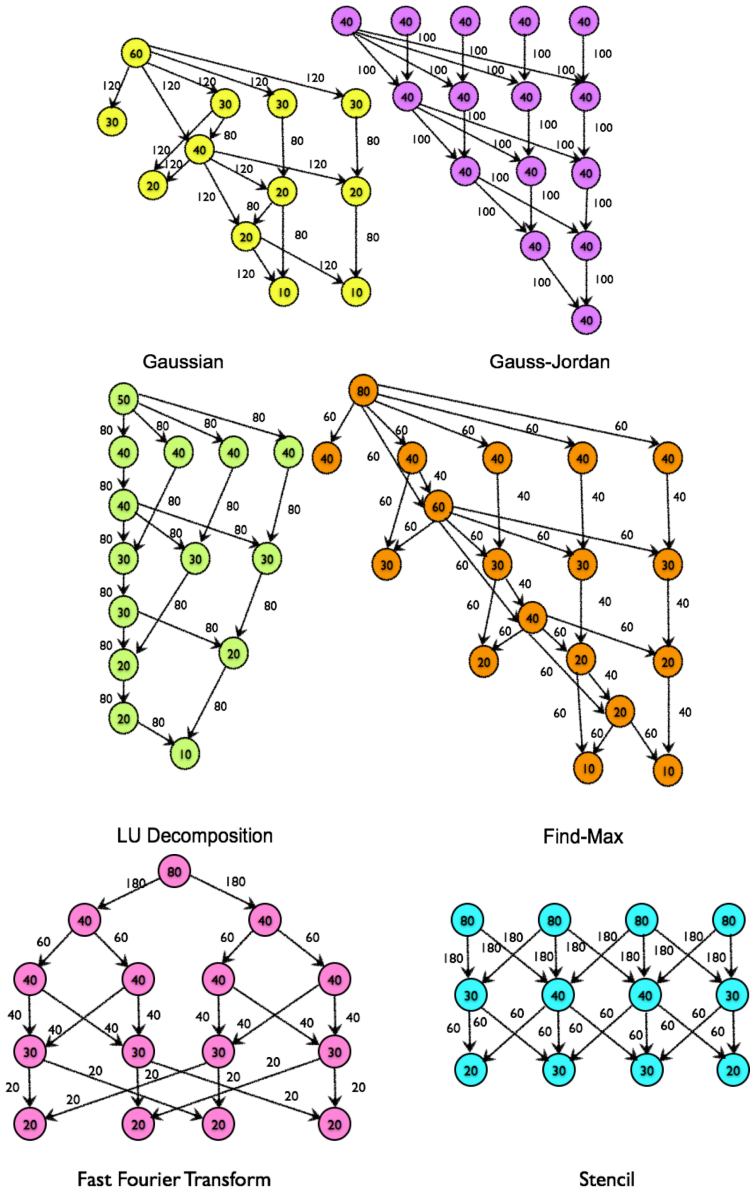

Figure 5.1.1: Workflows: Gaussian, Gauss-Jordan, LU Decomposition , Find-Max, Fast Fourier Transform (FFT), and Stencil.

5.2 Grid Topologies: EU DataGrid and WWG

The first topology used in the analysis is a combination between two testbeds. The topology information is based on the EU DataGrid testbed [84], and it is completed with grid resource characteristics from the WWG testbed [21]. The combination of these testbeds is shown in Figure 5.2.1 and in Table 5.2.1. Moreover, we have simulated three working nodes (WNs) per resource to add more complexity to this environment.

Table 5.2.1: Resource Characteristics WWG testbed.

Resource Name	Features (Vendor, Type, OS, CPUs/WN)	Resource Manager Type	MIPS /CPU	Price(G$) /CPU time	Power(W) /CPU time
LYON	Compaq, AlphaServer, OSF1, 4	Time-shared	515	8	67
CERN	Sun, Ultra, Solaris, 4	Time-shared	377	4	50
RAL	Sun, Ultra, Solaris, 4	Time-shared	377	3	50
IMPERIAL	Sun, Ultra, Solaris, 2	Time-shared	377	3	50
NORDUGRID	Intel, Pentium/VC820, Linux, 2	Time-shared	380	2	29.05
NIKHEF	SGI, Origin 3200, IRIX, 6	Time-shared	410	5	17
PADOVA	SGI, Origin 3200, IRIX, 16	Time-shared	410	5	17
BOLOGNA	SGI, Origin 3200, IRIX, 6	Space-shared	410	4	17
ROME	Intel, Pentium/VC820, Linux, 2	Time-shared	380	1	29.05
TORINO	SGI, Origin 3200, IRIX, 4	Time-shared	410	6	17
MILANO	Sun, Ultra, Solaris, 8	Time-shared	377	3	50

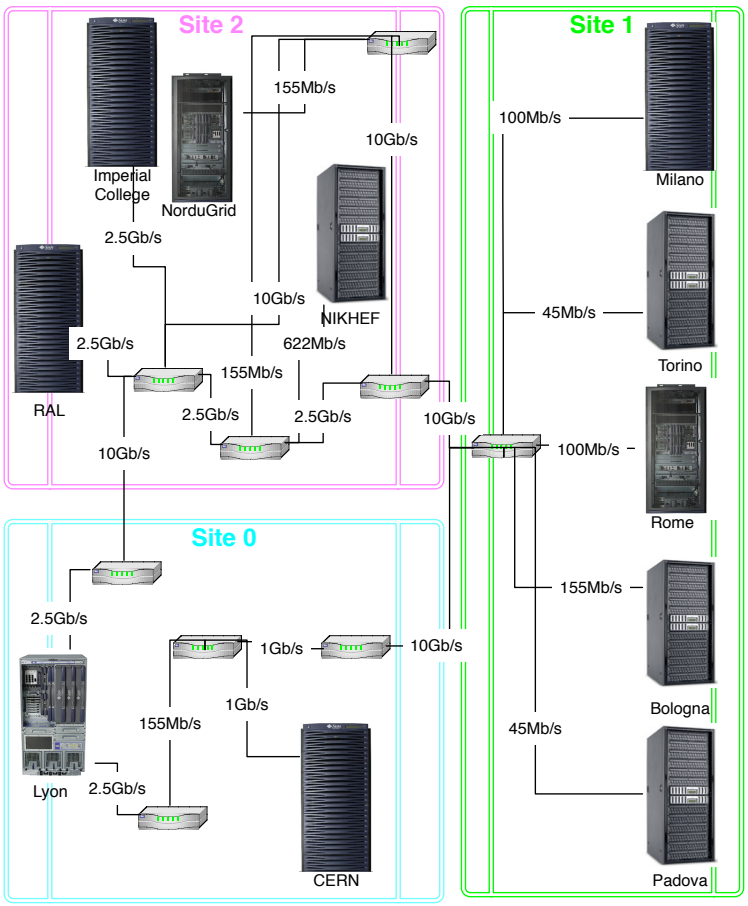

Figure 5.2.1: Testbed EU DataGrid.

5.3 Grid Topologies: IBERGRID

The second topology considered in the analysis is based on the Spanish Grid Initiative (NGI) [1] including the Portuguese Grid infrastructure to represent the Iberian Grid Infrastructure (IBERGRID) [2]. This testbed was carried out in the CETA-

[1] http://www.e-ciencia.es/ngi/.
[2] http://www.ibergrid.eu.

Figure 5.3.1: Site View from the IBERGRID infrastructure [16]

Ciemat [3] during the month of June 2011. The information of the grid sites is shown in Figure 5.3.1 and Table 5.3.1. The network communication is based on the RedIris-NOVA[4] using the combination of the RedIris[5] and the RCTS[6] (Figure 5.3.2), where the fiber ring communication network allows 10 Gbps. The cost of the resources is adapted following the EU Data Grid testbed. In order to take advantage of this complex topology, we have considered that the size of jobs sent per each workflow is measured in the Million of MI.

[3] http://www.ceta-ciemat.es/.
[4] http://www.redirisnova.es/caracteristicas/mapa-red.html.
[5] http://www.rediris.es/lared/mapa.html.
[6] http://www.umic.pt/.

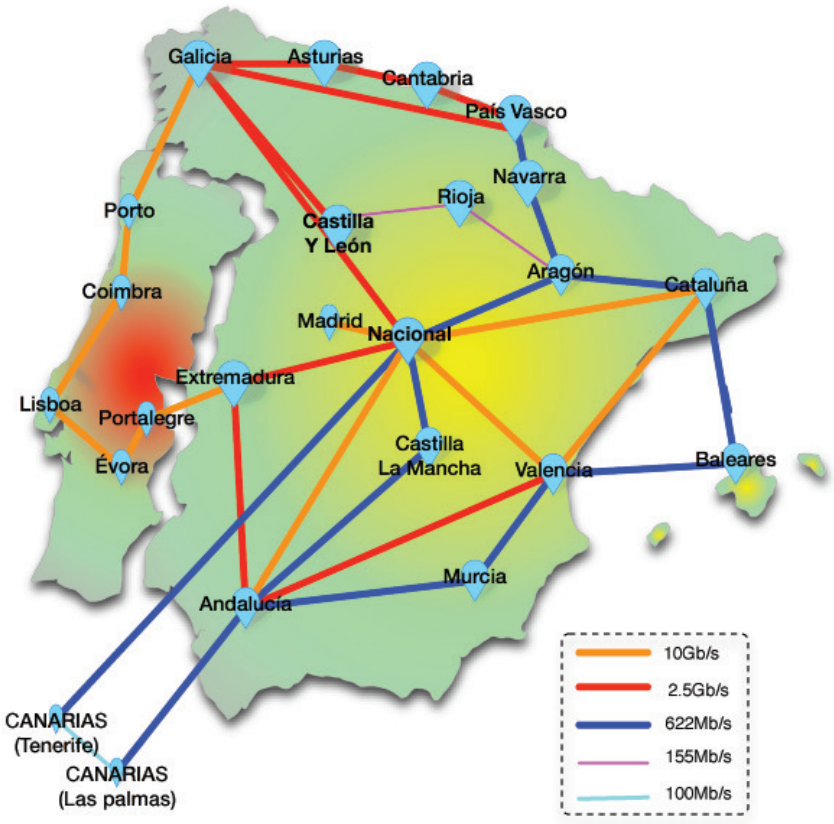

Figure 5.3.2: Communication Network: RedIris and RCTS maps [16].

5.4 GridSim

GridSim [21] is the simulator used to implement all the meta-schedulers. GridSim [20] is a Java-based toolkit for modelling and simulating distributed resource management in Grid environments. GridSim is based on SimJava, a general-purpose discrete-event simulation package implemented in Java. All components in GridSim communicate with each other through message passing operations defined by SimJava. It allows modelling of heterogeneous types of resources operating under space or time shared mode. The resource capability can be defined in the form of million instructions per second (MIPS) and can be located in any time

Table 5.3.1: Resource Characteristics IBERGRID testbed (¢G$ means cent of Grid dollars). Note: Some sites have more than one resource or computing element.

Resource Name	Features (Vendor, Type, OS, CPUs, WN)	Resource Manager Type	MIPS /CPU	Price(¢G$) /CPU time	Power(W) /CPU time
CESGA	AMD, Opteron 6174, Linux (Carbon), 21, 24	Space-shared	4400	56	115
USC	Intel, Xeon E5335, Linux (Boron), 22, 8	Space-shared	3990	18	80
UNICAN	Intel, Pentium D, CentOS, 241, 2	Space-shared	7505	91	130
CETA-Ciemat	Intel, Xeon 5130, Linux (Beryllium), 29, 4	Space-shared	6405	19	60
Ciemat-TIC	AMD, Opteron 270, Linux (Boron), 74, 4	Space-shared	3608	27	95
Ciemat-LCG	Intel, Xeon E5450, Linux (Boron), 118, 8	Space-shared	5985	142	80
SGAI-CSIC	AMD, Opteron 246, Red Hat (Tikanga), 9, 2	Space-shared	3983	2	89
BIFI	Intel, Xeon E5650, Linux (Boron), 31, 12	Space-shared	5334	50	80
IFIC1	Intel, Xeon E5420, Linux (Boron), 47, 8	Space-shared	5004	47	80
IFIC2	Intel, Xeon E5420, Linux (Boron), 106, 8	Space-shared	4988	106	80
IFISC	Intel, Xeon L5520, Linux (Boron), 60, 8	Space-shared	4534	55	60
IAA-CSIC	Intel, Xeon X7350, Red Hat (Tikanga), 32, 16	Space-shared	5863	75	130
Uporto1	AMD, Opteron 250, Linux (Boron), 23, 2	Space-shared	4786	6	89
Uporto2	AMD, Opteron 250, Linux (Boron), 11, 2	Space-shared	4779	3	89
Uminho1	Intel, Xeon E5420, Linux (Beryllium), 12, 1	Space-shared	4991	2	80
Uminho2	Intel, Xeon E5420, Linux (Boron), 8, 1	Space-shared	4988	1	80
LIP-Coimbra1	Intel, Xeon E5472, Linux (Boron), 22, 8	Space-shared	5985	26	80
LIP-Coimbra2	Intel, Xeon E5472, Linux (Boron), 22, 8	Space-shared	5985	26	80
NCG-INGRID	Quad-Core AMD, Opteron 2356, Linux (Boron), 128, 8	Space-shared	4600	118	75
CFP-IST	Intel, Core i7 980, Linux (Boron), 4, 12	Space-shared	6747	8	130
LIP-LISBON	Intel, Xeon E5472, Linux (Boron), 2, 8	Space-shared	5985	2	80
IFCA1	Intel, Xeon E5420, Linux (Boron), 182, 8	Space-shared	6254	228	80
IFCA2	Intel, Xeon E5420, Linux (Boron), 182, 8	Space-shared	6254	228	80
IFCA3	Intel, Xeon E5420, Linux (Boron), 182, 8	Space-shared	6254	228	80
IEETA	Intel, Xeon E5130, Linux (Boron), 4, 2	Space-shared	4989	1	65

zone. Moreover, applications with different parallel application models can be simulated. The GridSim toolkit is suitable for application scheduling simulations in Grid Computing environments. GridSim is of great value in testing new algorithms and strategies in a controlled environment. By using GridSim, it is possible to perform repeatable experiments and studies that are not possible in a real dynamic Grid environment. The main advantage of GridSim is that various allocation or scheduling policies can be implemented and integrated into GridSim easily, by extending them from one of the existing classes. Research students in the GRIDS Laboratory [30] tend to be heavy users of GridSim extending it whenever necessary for their own research needs. In the last five years, GridSim has been

continuously extended in this manner to include many new capabilities and has also received contributions from external collaborators. It has been chosen due to these advantages and as well as its ability to configure complex topologies and resource features, such as processing speed, MIPS, cost of resources per time unit, and/or energy consumption. Furthermore, thanks to the GridSim's flexibility, it has been modified to support workflows with dependent jobs, due to the importance of controlling the execution time in this type of workflow (child jobs are required to wait until the parent jobs are successfully executed).

"A methodology will lack the precision of a technique but will be a firmer guide to action than a philosophy. Where a technique tells you 'how' and a philosophy tells you 'what', a methodology will contain elements of both 'what' and 'how'."

Peter Checkland

6 Methodology

The methodology followed during the analysis process is presented in this chapter. In this thesis, two widely used quality indicators are applied and the corresponding explanation, motivation and definition are described in section 6.1. Moreover, in order to reinforce the reliability of the results presented in this thesis a statistical analysis is performed, the process of this analysis is explained in section 6.2. Finally, the specific parameters of the algorithms implemented in this research are enumerated in section 6.3 with the aim to make this research reproducible for other researches.

6.1 Quality Assessment in Multiobjective Optimization

In order to evaluate and compare the quality of the sets of non-dominated solutions from out multi-objective approaches, it is normally considered in the objec-

tive space towards the global Pareto-optimal front or simply Pareto front. A fair comparison between two approximations should be performed by using the Pareto dominance concept. However, since the Pareto dominance is not a total order, not all points are comparable.

In our problem we have considered two quality indicators widely used in the literature: *Hypervolume (HV)* and *Set Coverage (SC)*. Moreover, none of them need to know the Pareto-optimal to compare with, which is ideal to apply in our thesis. Furthermore, *hypervolume* and *set coverage* are complementary, because the *hypervolume* is a unitary quality indicator, while the *set coverage* is a binary indicator, that allows a suitable evaluation and comparison from different point of view over our multi-objective approaches.

Following, we present the definition of these quality indicators with the corresponding figures.

Definition 1: *Hypervolume (HV)* [107]. It measures the volume (in the objective function space) covered by members of a non-dominated set of solutions. In a multiobjective optimization problem with M minimization objective functions, we calculate the size of the region of the objective space (hypervolume) of a non-dominated set of solutions $A = a^1, .., a^k$ bounded by a reference point $r = (r_1, \ldots, r_M)$. The corresponding hypercube for each member a^i of set A is calculated as follows: $h(a^i) = [a^i_1, r_1] \times \ldots \times [a^i_M, r_M]$. In this way, as it is shown in equation 6.1 the hypervolume of A is computed by the union of these $|A|$ hypercubes, with repeatedly covered hypercubes being counted once, where L refers to Lebesgue [17] measure.

$$HV(A, r) = L\left(\bigcup_{i=1}^{|A|} h(a^i) | a^i \in A\right) \tag{6.1}$$

In Figure 6.1.1 we present an illustration of the 2-dimensional and 3-dimensional hypervolume of a set of non-dominated solutions.

Definition 2: *Set Coverage (SC)* [104]. If we suppose two sets of non-dominated

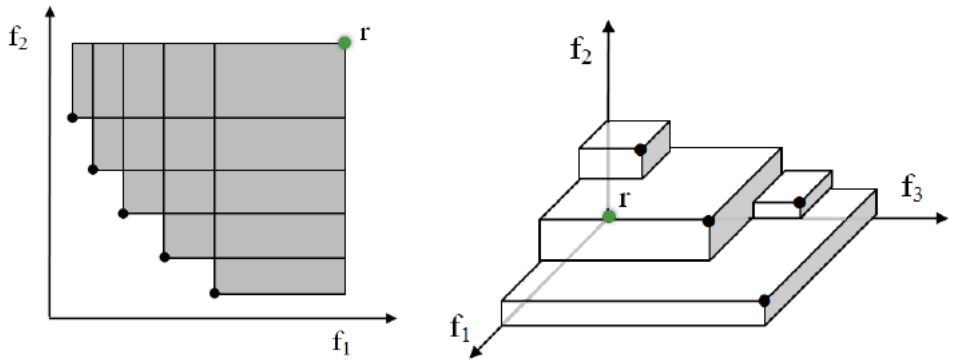

Figure 6.1.1: Hypervolume (HV), quality indicator. 2-dimensional and 3-dimensional HV.

solutions $A = a^1, \ldots, a^k$ and $B = b^1, \ldots, b^j$, this indicator measures the fraction of non-dominated solutions in B; which are covered by the non-dominated solutions in A as:

$$SC(A, B) = \frac{|\{b \in B; \exists a \in A : a \preceq b\}|}{|B|} \tag{6.2}$$

If $SC(A, B) = 1$, all points in B are dominated by or equal to points in A, whereas $SC(A, B) = \phi$ means that none of the points in B are covered by the set of A. As the dominance operator is not symmetric, it is necessary to calculate both $SC(A, B)$ and $SC(B, A)$, since $SC(B, A)$ is not necessarily equal to $1 - SC(A, B)$. An example of this indicator is shown in Figure 6.1.2. In Figure 6.1.2, the black points (red) are the solutions in A, and the grey (green) points are the solutions in B. As we may observe, the front A dominates 30% of the front B, whereas the front B dominates 60% of the front A.

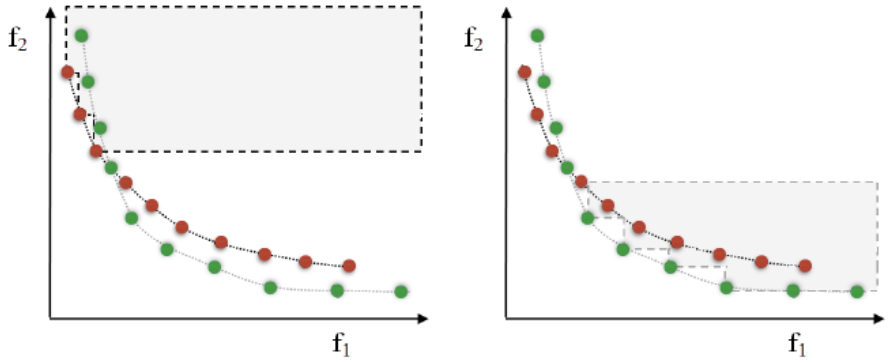

Figure 6.1.2: Set Coverage (SC), quality indicator. SC(A, B) and SC(B, A)

6.2 Statistical Reliability

In this thesis, with the aim of making a comparison among the algorithms with a certain level of confidence, and because we are dealing with stochastic algorithms, we have performed a statistical analysis of the results obtained. This statistical analysis has been applied to all the results in this thesis.

More specifically, the statistical analysis applied in this research is shown in Figure 6.2.1. This statistical analysis is based on 30 independent repetitions for every experiment. The first test carried out consists of the calculation of the residual normality, in our case, Kolmogorov-Smirnov [78]. This test checks whether or not the results' values follow a Gaussian distribution. In the case of non-Gaussian distributions, a non-parametric analysis is applied (the well-known Kruskal-Wallis [57]). However, if a Gaussian distribution is found, a homogeneity of variances test is also carried out (Levene [62]). Finally, if the homogeneity is positive, an ANOVA [78] analysis is performed; otherwise Kruskal-Wallis analysis is applied. The confidence level considered in this work is always 95% in the statistical tests with a significance level of 5% or p-values under 0.05. The differences are unlikely to have occurred by chance with a probability of 95%.

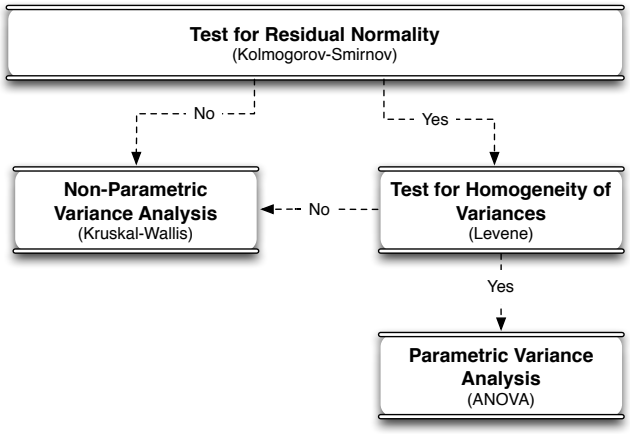

Figure 6.2.1: Statistical Analysis Process performed in this work.

6.3 Parametrization

Parameter settings for each multi-objective algorithm are presented. We have previously performed a study (each single experiment was repeated 30 times independently) to find the best parameter configuration for each algorithm in order to solve our problem. The parameters used in the algorithms have been properly adjusted after performing numerous experiments. First, we performed several runs to observe which is the most influential parameter. Thus, the most influential parameter will be the first to be set up, and so on. As said, in each experiment we have performed 30 independent runs, which are sufficient to ensure the statistical significance of the obtained results. To select the tested values for each parameter, we have examined its range of possible values, establishing a minimum of five possible uniformly distributed values. If two distinct values present similar results, we establish a new intermediate value and repeat the process. The parameter value that finally achieves the best results (using the hypervolume [106] as quality metric) in most workflows will be established. This process is repeated for all parameters and all algorithms.

- MOGSA: Population size = 25, G_o = 10000, Min_{Kbest} = 5, α = 2, ε = 1, maxi-

mum time of execution = 2 minutes.

- MOABC: Population size = 101 (50 employed bees, 50 onlooker bees and 1 scout bee), mutation probability = 0.25, maximum time of execution = 2 minutes.
- MOSWO: Population Size = 100, $P_{mutation}$ = 0.25, maximum time of execution = 2 minutes.
- MO-FA: Population Size = 100, β = 0.2 , γ = 1, α = 1, maximum time of execution = 2 minutes.
- MOBSA: Population Size (n) = 25, Cluster Number (m) = 5, $Pcenter_{replacement}$ = 0.4, $Pcluster_{mutation}$ = 0.8, $Pcenter_{mutation}$ = 0.8, $Pcluster_{cross}$ = 0.25, maximum time of execution = 2 minutes.
- NSGA-II: Population Size = 100, $P_{crossover}$ = 0.9, $P_{mutation}$ = 0.1, selection by binary tournament, maximum time of execution = 2 minutes.

"There can be economy only where there is efficiency."

Benjamin Disraeli

7

Results and Analysis: Economic Cost and Execution Time Optimization

This chapter is divided to three main analyses. The first one performs a statistical analysis of our multi-objective approaches, which are compared with the standard and well-known multi-objective algorithm NSGA-II. In the second section, the best multi-objective algorithm from our previous study is compared with the latest and well-known multi-objective scheduling algorithm, MOHEFT. Finally, the third study compares our best multi-objective algorithm with two real grid meta-schedulers: WMS and DBC.

7.1 Multi-Objective comparison

In this analysis, we compare the multi-objective algorithms presented in Chapter 4 (MOGSA, MOABC, MOSWO, MO-FA, and MOBSA) from the physical, biological, and sociological fields with the standard and well-known NSGA-II, which is an often used algorithm in multi-objective optimization. The first part of this section evaluates the algorithms on the EU DataGrid and WWG combination testbed, and the second part uses the IBERGRID testbed as a grid environment (see Chapter 5 for more details).

7.1.1 EU DataGrid and WWG combination testbed

First, we compare the six algorithms using the hypervolume metrics. Tables 7.1.1 and 7.1.2 show the median and interquartile range for each algorithm. We observe that the reliability of the swarm algorithms (MOABC, MOGSA, MOSWO, and MOBSA) is greater than the standard NSGA-II in all the cases with a lower interquartile range.

Table 7.1.1: MOGSA, MOABC, and MOSWO Hypervolume properties by workflow. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Workflows	Median (Interquartile Range)			Reference Point
	MOGSA	MOABC	MOSWO	(Time (s), Cost (G$))
Gaussian	55.06 (0.37)	57.44 (0.23)	53.18 (0.62)	(1000, 10000)
Gauss-Jordan	55.53 (0.43)	57.29 (0.63)	54.45 (0.38)	(1200, 22000)
LU	54.48 (0.39)	56.57 (0.14)	52.71 (0.89)	(1200, 22000)
Find-Max	54.23 (0.15)	56.57 (0.14)	53.45 (0.32)	(2000, 10000)
FFT	55.94 (0.38)	57.41 (0.66)	54.79 (0.31)	(1200,22000)
Stencil	57.40 (0.42)	59.61 (1.08)	56.20 (0.31)	(1000, 15000)
Average	55.44 (0.35)	57.48 (0.45)	54.13 (0.47)	-

The MOABC algorithm presents the best average, 57.48, followed by MOGSA

with an average value of 55.44, while NSGA-II only achieves a value of 45.60 on average, by being the worst algorithm in this test.

Table 7.1.2: MO-FA, MOBSA, and NSGA-II Hypervolume properties by workflow. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Workflows	Median (Interquartile Range)			Reference Point
	MO-FA	MOBSA	NSGA-II	(Time (s), Cost (G$))
Gaussian	46.39 (1.47)	51.93 (0.60)	46.40 (1.11)	(1000, 10000)
Gauss-Jordan	47.51 (0.92)	54.07 (0.06)	47.17 (0.46)	(1200, 22000)
LU	48.30 (0.77)	51.34 (0.78)	48.02 (1.27)	(1200, 22000)
Find-Max	36.10 (1.83)	53.06 (0.39)	36.23 (1.51)	(2000, 10000)
FFT	49.21 (0.91)	54.15 (0.43)	48.88 (0.89)	(1200,22000)
Stencil	46.82 (0.94)	55.94 (0.39)	46.91 (1.38)	(1000, 15000)
Average	45.72 (1.14)	53.41 (0.44)	45.60 (1.10)	-

Moreover, a statistical boxplot is provided for each workflow in order to visualize and reinforce the statistical results (see Figure 7.1.1).

In Figure 7.1.2, all the Pareto fronts are presented for each workflow and algorithm. Results show that all the algorithms follow the same behaviour regarding the studied workflows. However, the MOABC's Pareto fronts always dominate the solutions obtained by the rest of the multi-objective algorithms.

Moreover, a numerical comparison of these six algorithms with regard to set coverage metrics is given in Table 7.1.3. Each cell gives the fraction of non-dominated solutions evolved by algorithm B, which are covered by the non-dominated points achieved by algorithm A [106]. Again, MOABC covers all the results obtained by NSGA-II and almost all the solutions (88.24%) found by MOGSA (our second best algorithm in this evaluation).

Figure 7.1.1: Boxplot Gaussian, Gauss-Jordan, LU, FindMax, FFT, and Stencil workflows. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

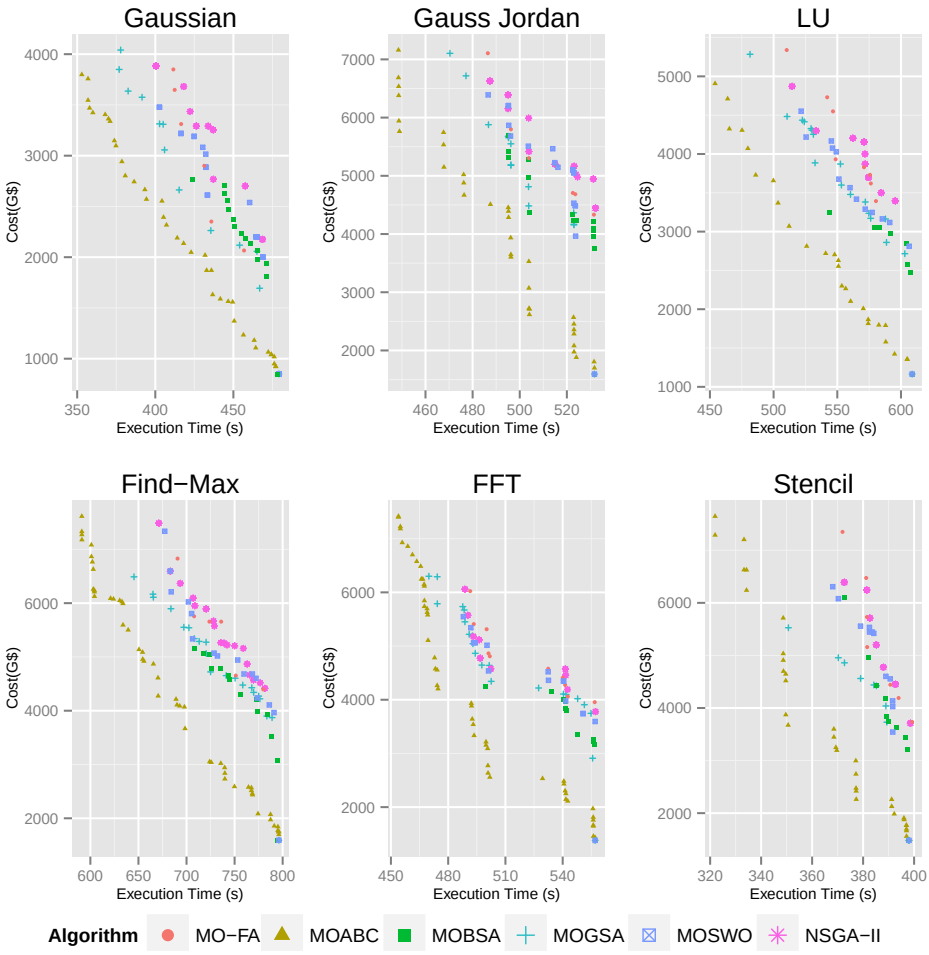

Figure 7.1.2: Pareto fronts by workflow and algorithm. In every case, from the 30 repetitions, the closest Pareto front to the median hypervolume is shown. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Table 7.1.3: Set Coverage comparison of MOGSA, MOABC, MOSWO, MO-FA, MOBSA and NSGA-II by workflow. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Gauss-Jordan	LU	Find-Max	FFT	Stencil	
MOGSA	MOABC	0.00	0.00	3.70	0.00	2.00	5.71	1.90
MOABC	MOGSA	92.30	92.85	83.33	90.47	90.47	80.00	88.24
MOGSA	MOSWO	81.81	85.71	68.42	80.00	66.66	92.30	79.15
MOSWO	MOGSA	0.00	7.14	0.00	15.00	23.80	10.00	9.32
MOGSA	MO-FA	83.33	100.00	100.00	100.00	90.00	100.00	95.55
MO-FA	MOGSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOGSA	MOBSA	92.85	53.33	50.00	50.00	36.36	45.45	54.66
MOBSA	MOGSA	7.69	28.57	27.77	35.71	33.33	20.00	25.51
MOGSA	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOGSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MOSWO	90.90	92.85	100.00	93.33	91.66	92.30	93.50
MOSWO	MOABC	0.00	2.94	3.70	0.00	2.00	2.85	1.91
MOABC	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MOBSA	93.33	93.33	100.00	92.85	100.00	90.91	95.07
MOBSA	MOABC	0.00	0.00	3.70	7.54	2.00	5.71	3.16
MOABC	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOSWO	MO-FA	50.00	57.14	87.50	83.33	100.00	71.42	74.89
MO-FA	MOSWO	45.45	28.57	7.14	26.66	0.00	38.46	24.38
MOSWO	MOBSA	36.66	20.00	12.50	0.00	9.09	27.27	17.58
MOBSA	MOSWO	55.55	66.66	50.00	56.25	46.66	53.84	54.83
MOSWO	NSGA-II	100.00	72.72	100.00	74.47	90.00	71.42	84.77
NSGA-II	MOSWO	0.00	16.66	7.14	20.00	8.33	0.00	8.69
MO-FA	MOBSA	60.00	0.00	100.00	0.00	0.00	0.00	26.66
MOBSA	MO-FA	16.66	85.71	0.00	100.00	100.00	100.00	62.56
MO-FA	NSGA-II	88.88	81.81	66.66	58.82	30.00	57.14	63.88
NSGA-II	MO-FA	0.00	0.00	37.50	33.33	70.00	42.85	30.61
MOBSA	NSGA-II	100.00	90.90	77.77	100.00	100.00	100.00	94.77
NSGA-II	MOBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00

As we mentioned in the previous chapter, we carried out a statistical analysis to make a comparison with a major level of confidence. The confidence level considered in this work is always 95% in the statistical tests with a significance level of 5% or p-values under 0.05. The differences are unlikely to have occurred by chance with a probability of 95%. The summary of the statistical study is presented in Table 7.1.4, where only those data sets in which no statistical differences were found (p-value > 0.05) are shown. The MOABC algorithm results are significantly different from the results obtained by the other algorithms in all cases, including MOGSA, which is our second-best algorithm. Therefore, the evidence points to MOABC as our winner algorithm in this complete multi-objective comparison.

Table 7.1.4: Statistical analysis of the comparison among MOGSA, MOABC, MOSWO, MO-FA, MOBSA, and NSGA-II using hypervolume metrics. The non-significant differences are shown in this table. EU DataGrid and WWG tesbeds. Economic Cost and Execution Time Optimization.

	MOGSA	MOABC	MOSWO	MO-FA	MOBSA	NSGA-II
MOGSA		-	-	-	-	-
MOABC			-	-	-	-
MOSWO				-	-	-
MO-FA					-	#Gaussian, #Gauss-Jordan, #LU, #Find-Max, #FFT, #Stencil
MOBSA						-
NSGA-II						

7.1.2 IBERGRID testbed

The same multi-objective study is carried out for the IBERGRID testbed. Hypervolume metrics is calculated for each algorithm and workflow. Tables 7.1.5 and 7.1.6 present the median and interquartile values of hypervolumes.

Hypervolume metrics show that MOABC obtains a better set of solutions than the rest of algorithms with an average of 58.94, followed by the average value of MOBSA (53.91). In this evaluation NSGA-II obtains almost the worst average (38.30).

Table 7.1.5: MOGSA, MOABC, and MOSWO Hypervolume properties by workflow. IBERGRID testbed. Economic Cost and Execution Time Optimization.

Workflows	Median (Interquartile Range)			Reference Point
	MOGSA	MOABC	MOSWO	(Time (s), Cost (¢G$))
Gaussian	56.39 (0.90)	60.69 (0.35)	52.59 (1.03)	(65000, 6500000)
Gauss-Jordan	53.03 (0.46)	58.16 (1.92)	51.11 (0.75)	(80000, 8000000)
LU	52.99 (0.74)	58.55 (0.71)	48.58 (1.24)	(80000, 8000000)
Find-Max	52.91 (0.50)	57.76 (0.69)	49.31 (1.15)	(105000, 10500000)
FFT	53.72 (0.79)	58.26 (0.72)	50.59 (1.20)	(80000, 8000000)
Stencil	53.62 (0.77)	60.23 (1.08)	50.52 (1.26)	(60000, 6000000)
Average	53.77 (0.69)	58.94 (0.91)	50.45 (1.10)	-

Table 7.1.6: MO-FA, MOBSA, and NSGA-II Hypervolume properties by workflow. Testbed IBERGRID. Execution Time and Cost Optimization.

Workflows	Median (Interquartile Range)			Reference Point
	MO-FA	MOBSA	NSGA-II	(Time (s), Cost (¢G$))
Gaussian	44.74 (2.16)	55.80 (0.79)	44.80 (2.46)	(65000, 6500000)
Gauss-Jordan	35.09 (4.67)	53.77 (0.46)	34.39 (3.79)	(80000, 8000000)
LU	40.00 (3.15)	53.41 (0.97)	40.27 (2.76)	(80000, 8000000)
Find-Max	37.65 (2.83)	52.87 (0.93)	38.42 (3.46)	(105000, 10500000)
FFT	37.10 (3.71)	53.53 (0.57)	38.04 (3.43)	(80000, 8000000)
Stencil	33.44 (4.27)	54.13 (1.48)	33.88 (6.59)	(60000, 6000000)
Average	38.00 (3.46)	53.91 (0.86)	38.30 (3.74)	-

The corresponding boxplot for this test is provided in Figure 7.1.3. The results obtained are similar to the results performed in the study with the WWG and EU DataGrid testbed, except that MOBSA is second-best algorithm in this case.

Furthermore, graphs of the resulting Pareto fronts are displayed in Figure 7.1.4, and the direct comparison with the set coverage metrics is calculated in Table 7.1.7. These tests prove that MOABC usually covers almost all the solutions from the other algorithms. This second test environment demonstrates a similar behaviour as the first one, highlighting MOABC as the best meta-scheduler in terms of execution time and cost optimization. Moreover, it also proves that swarm algorithms have better behaviour than NSGA-II in dealing with the job scheduling problem in grid environments.

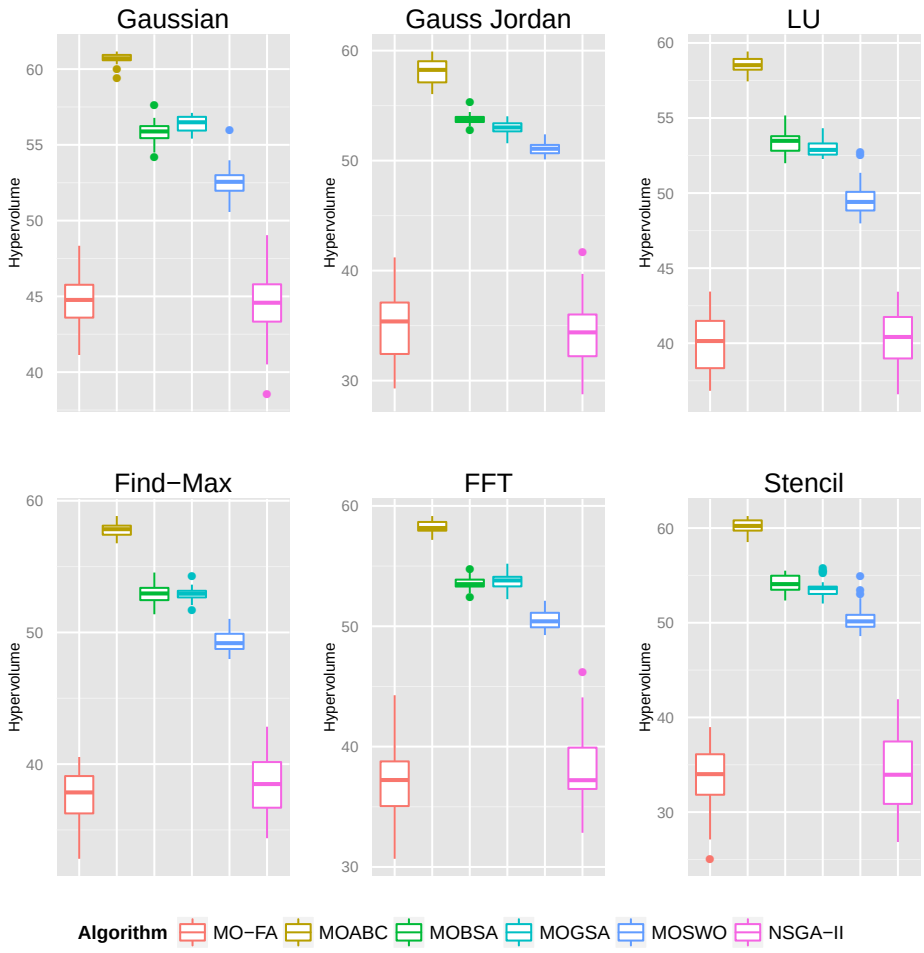

Figure 7.1.3: Boxplot Gaussian, Gauss-Jordan, LU, FindMax, FFT, and Stencil workflows. IBERGRID testbed. Economic Cost and Execution Time Optimization.

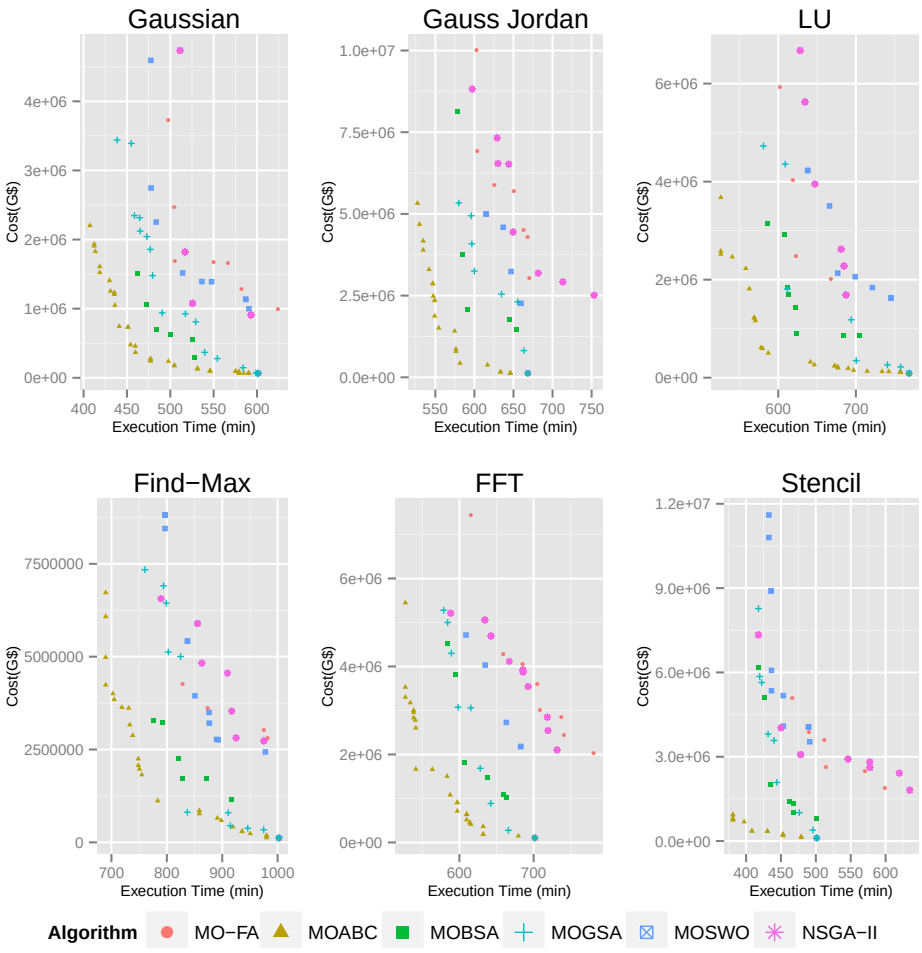

Figure 7.1.4: Pareto fronts by workflow and algorithm. In every case, from the 30 repetitions, the closest Pareto front to the median hypervolume is shown. IBERGRID testbed. Economic Cost and Execution Time Optimization.

Table 7.1.7: Set Coverage comparison of MOGSA, MOABC, MOSWO, MO-FA, MOBSA, and NSGA-II by workflow. IBERGRID testbed. Economic Cost and Execution Time Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Gauss-Jordan	LU	Find-Max	FFT	Stencil	
MOGSA	MOABC	12.50	4.54	3.33	7.40	0.00	0.00	4.63
MOABC	MOGSA	93.75	88.88	100.00	81.81	100.00	90.00	92.41
MOGSA	MOSWO	88.88	80.00	85.71	90.00	80.00	90.00	85.76
MOSWO	MOGSA	0.00	22.22	12.50	9.09	11.11	69.23	20.69
MOGSA	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	MOGSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOGSA	MOBSA	33.33	28.57	22.22	42.85	14.28	11.11	25.39
MOBSA	MOGSA	50.00	44.44	37.50	36.36	33.33	40.00	40.27
MOGSA	NSGA-II	100.00	100.00	100.00	85.71	100.00	100.00	97.61
NSGA-II	MOGSA	0.00	0.00	0.00	9.09	0.00	0.00	1.51
MOABC	MOSWO	90.00	85.71	100.00	81.81	100.00	92.30	91.64
MOSWO	MOABC	5.00	4.54	3.33	3.70	3.45	0.00	3.34
MOABC	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MOBSA	66.66	71.42	88.88	71.42	100.00	88.88	81.21
MOBSA	MOABC	12.50	0.00	3.33	3.70	3.45	0.00	3.83
MOABC	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOSWO	MO-FA	85.71	57.14	0.00	50.00	77.77	66.67	56.21
MO-FA	MOSWO	0.00	0.00	42.85	9.09	0.00	0.00	8.66
MOSWO	MOBSA	0.00	28.57	11.11	14.28	14.28	22.22	15.08
MOBSA	MOSWO	90.00	85.71	100.00	72.73	100.00	84.61	88.84
MOSWO	NSGA-II	50.00	100.00	33.33	57.14	80.00	50.00	61.74
NSGA-II	MOSWO	20.00	0.00	14.28	27.27	0.00	53.84	19.23
MO-FA	MOBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOBSA	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	NSGA-II	25.00	50.00	66.67	42.85	0.00	37.50	37.00
NSGA-II	MO-FA	42.85	28.57	0.00	50.00	44.44	50.00	35.98
MOBSA	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00

The statistical analysis performed with the previous testbed is also applied to the datasets obtained with the IBERGRID topology. Once more, MOABC presents significantly different results compared to the results obtained by the other multi-objective algorithms in all the cases with p-values under 0.05. We can observe that MOGSA and MOBSA do not have significant differences for four of the six workflows executed in the grid. Therefore, both are good enough for the second best algorithm in this last evaluation, however taking into account the perfomance of MOGSA in the previous testbed, we could say that MOGSA is our second most promising option. Finally, again, MO-FA and NSGA-II do not have significant differences for this problem.

Table 7.1.8: Statistical analysis of the comparison among MOGSA, MOABC, MOSWO, MO-FA, MOBSA, and NSGA-II using hypervolume metrics. The non-significant differences are shown in this table. IBERGRID testbed. Economic Cost and Execution Time Optimization.

	MOGSA	MOABC	MOSWO	MO-FA	MOBSA	NSGA-II
MOGSA		-	-	-	#LU, #Find-Max, #FFT, #Stencil	-
MOABC			-	-	-	-
MOSWO				-	-	-
MO-FA					-	#Gaussian, #Gauss-Jordan, #LU, #Find-Max, #FFT, #Stencil
MOBSA						-
NSGA-II						

7.2 MOHEFT comparison

HEFT (Heterogeneous Earliest Finish Time) algorithm ([102], [93]) is one of the most popular algorithms for workflow-scheduling. Recently, its multi-objective version, MOHEFT [37] has been compared with two state-of-the-art approaches: the classical HEFT algorithm used in single-objective scheduling and the SPEA2 algorithm used for multi-objective optimization problems. The MOHEFT algorithm optimizes both execution time and energy consumption objectives in computational workflow-scheduling [34]. Other new research compares MOHEF with greenHEFT, which is the single-objective scheduling for energy saving [35]. The results of MOHEFT show its superiority regarding its competitors and due to this, we compare our best algorithm, MOABC, with MOHEFT to demonstrate the quality of our approach. We have adapted MOHEFT so that it optimizes execution time and cost.

The MOHEFT algorithm improves and extends the HEFT algorithm. The only additional input parameter of MOHEFT is the size of the set of tradeoff solutions, *K*. The authors do not indicate an exact value for K, but they say [35]: "Given a set of n activities and m resources, the computational complexity of HEFT (and greenHEFT) is $O(n \times m)$ ... Considering that the set of tradeoff solutions is K, the extra loop in MOHEFT performs only K iterations, rendering a complexity of $O(n \times m \times K)$. Usually, the number of tradeoff solutions is a constant much lower than n and m ... Thus, the complexity can be approximated as almost $O(n \times m)$, as in HEFT (or greenHEFT)". For this reason, we have used K=5 in our experiments.

The comparative study between MOABC and MOHEFT is carried out by applying the multi-objective metrics used previously (section 7.1). In Tables 7.2.1 and 7.2.2, the hypervolume results demonstrate a higher quality of MOABC over MOHEFT regarding all the workflows and topologies studied. Note, that the interquartile range is not evaluated since MOHEFT is deterministic by always being equal to $\emptyset$.

Table 7.2.1: MOABC vs. MOHEFT: Hypervolume. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Workflows	Median		Reference Point
	MOABC	MOHEFT	(Time (s), Cost (G$))
Gaussian	57.44	48.70	(1000, 10000)
Gauss-Jordan	57.29	51.31	(1200, 22000)
LU	57.74	50.77	(1200, 22000)
Find-Max	56.57	43.89	(2000, 10000)
FFT	57.41	48.58	(1200,22000)
Stencil	59.61	46.72	(1000, 15000)
Average	57.67	48.32	-

Table 7.2.2: MOABC vs. MOHEFT: Hypervolume. IBERGRID testbed. Economic Cost and Execution Time Optimization.

Workflows	Median		Reference Point
	MOABC	MOHEFT	(Time (s), Cost (¢G$))
Gaussian	60.69	47.12	(65000, 6500000)
Gauss-Jordan	58.16	51.62	(80000, 8000000)
LU	58.55	52.53	(80000, 8000000)
Find-Max	57.76	52.14	(105000, 10500000)
FFT	58.26	52.58	(80000, 8000000)
Stencil	60.23	44.17	(60000, 6000000)
Average	58.94	50.02	-

The coverage between both algorithms is also evaluated. Once more, MOABC has good results in all the workflows and covers more solutions than MOHEFT by obtaining 61.66% of the coverage average against the 12.55% of MOHEFT (see Table 7.2.3 and 7.2.4).

Table 7.2.3: Set Coverage comparison of MOABC and MOHEFT per each workflow. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Jordan	LU	Find-Max	FFT	Stencil	
MOABC	MOHEFT	50.00	100.00	50.00	50.00	100.00	100.00	75.00
MOHEFT	MOABC	15.00	14.70	7.40	9.43	0.00	0.00	7.75

Table 7.2.4: Set Coverage comparison of MOABC and MOHEFT per each workflow. IBERGRID testbed. Economic Cost and Execution Time Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Jordan	LU	Find-Max	FFT	Stencil	
MOABC	MOHEFT	50.00	40.00	50.00	50.00	50.00	50.00	48.33
MOHEFT	MOABC	20.00	33.33	16.67	10.00	24.13	0.00	17.35

Moreover, the statistical analysis shows that both algorithms are significantly different with respect to all the instances studied ($p\text{-value} < 0.05$) (see Table 7.2.5).

Table 7.2.5: Statistical analysis of the comparison between MOABC and MOHEFT by using hypervolume metrics. The non-significant differences are shown in this table. EU DataGrid and WWG combination testbed and IBERGRID testbed. Economic Cost and Execution Time Optimization.

EU DataGrid and WWG combination testbed		
	MOABC	MOHEFT
MOABC		-
MOHEFT		

IBERGRID testbed		
	MOABC	MOHEFT
MOABC		-
MOHEFT		

7.3 Real grid meta-schedulers comparison

This second kind of analysis compares the best multi-objective algorithm, MOABC, studied in the previous analysis, with respect to two current, popular meta-schedulers (WMS and DBC) with all the workflows and the two topologies. Also, each experiment performed in our study includes 30 independent executions per algorithm and workflow. First, the WWG and EU DataGrid testbed is studied.

WMS has been deployed by taking into account the free CPUs of every resource to calculate the rank of preferences. In these tests, the meta-scheduler is located in the European Council for Nuclear Research (CERN). Table 7.3.1 shows the time and cost obtained for each meta-scheduler and workflow. In these results, we can see that MOABC always offers a better solution than WMS and DBC in both objectives at the same time. However, DBC provides similarly beneficial solutions. Because of that, we performed another test in order to check the performance of the meta-schedulers by providing a range of deadline constraints.

Table 7.3.1: Results of MOABC, WMS, and DBC by workflow. Time (s) and Cost (G$). EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Workflows	MOABC		WMS		DBC	
	Time	Cost	Time	Cost	Time	Cost
Gaussian	478.11	848.14	482.68	3434.82	480.82	850.00
Gauss-Jordan	531.71	1593.74	534.70	6405.20	533.41	1593.74
LU	608.76	1164.00	612.29	4684.63	610.46	1164.00
Find-Max	796.07	1591.74	797.67	2929.98	797.77	1591.74
FFT	542.62	2113.00	561.24	5560.52	544.96	2583.00
Stencil	377.18	2742.99	400.02	5973.12	378.98	2368.90

In Table 7.3.2, three restrictive deadlines are performed per each workflow. Results show that WMS is not consistently able to accomplish the execution of entire workflows and DBC requires complexity to achieve it, while MOABC optimizes the solutions and always completes the execution of the workflows.

Using the IBERGRID topology, results show a similar behaviour regarding to

Table 7.3.2: MOABC, WMS, and DBC: Successfully executed jobs with regard to deadline variation. EU DataGrid and WWG combination testbed. Economic Cost and Execution Time Optimization.

Workflows	Constraint	MOABC			WMS			DBC		
	Deadline	Time	Cost/ Job	Jobs	Time	Cost/ Job	Jobs	Time	Cost/ Job	Jobs
Gaussian	460	456.41	102.85	12	460.00	325.93	10	460.43	79.63	10
	445	441.62	132.39	12	445.00	340.36	9	445.64	105.44	10
	430	422.92	170.66	12	430.00	321.06	7	430.73	82.63	9
Gauss-Jordan	530	524.01	125.44	15	530.00	547.12	14	525.71	121.35	15
	515	504.11	174.45	15	515.00	419.81	14	505.82	112.12	15
	500	496.24	240.29	15	500.00	422.67	14	500.08	101.84	14
LU	560	556.62	161.81	14	560.00	389.22	13	560.80	123.14	12
	545	540.89	194.24	14	545.00	372.89	10	545.00	178.67	12
	530	525.97	200.85	14	530.00	344.95	10	500.08	123.62	10
Find-Max	750	749.87	143.86	18	721.42	399.25	15	750.68	99.03	15
	735	724.08	169.54	18	721.42	440.07	15	735.68	117.52	15
	720	697.80	226.12	18	667.32	445.13	12	720.68	149.14	15
FFT	540	529.43	168.84	15	507.13	426.90	11	540.06	170.80	11
	525	501.83	168.84	15	507.13	426.90	11	525.06	191.90	11
	510	501.83	168.84	15	507.13	426.90	11	510.06	179.89	11
Stencil	380	377.18	175.38	12	373.48	533.25	10	379.03	177.40	12
	365	350.59	276.15	12	319.42	612.54	8	365.60	203.92	10
	330	321.94	387.32	12	319.42	612.54	8	330.70	174.71	8

the previous topology studied. The meta-scheduler is located in Ciemat-TIC in all the tests. In Table 7.3.3, results indicate MOABC is the best meta-scheduler for both objectives. Moreover, when we applied the deadline constraints (see Table 7.3.4), both WMS and DBC were not able to execute all the jobs that comprise the workflows. However, MOABC has no problems in carrying out the experiments. The jobs are interdependent; therefore the execution of all the jobs may be necessary.

Table 7.3.3: Results of MOABC, WMS, and DBC by workflow. Time (s) and Cost (¢G$). IBERGRID testbed. Economic Cost and Execution Time Optimization.

Workflows	MOABC		WMS		DBC	
	Time	Cost	Time	Cost	Time	Cost
Gaussian	36085.88	641.53	30081.41	75940.03	36090.01	641.53
Gauss-Jordan	38759.33	1256.25	55437.59	4490.21	38764.00	1309.65
LU	45442.98	908.74	63754.48	32930.58	45445.66	1114.26
Find-Max	60140.90	1202.76	72328.42	3013.16	60145.21	1202.76
FFT	42098.00	1042.35	52927.30	2611.39	42102.02	1239.45
Stencil	28735.14	1202.76	37663.20	2812.18	29072.86	1289.88

Table 7.3.4: MOABC, WMS, and DBC: Successfully executed jobs with regard to deadline variation. IBERGRID testbed. Economic Cost and Execution Time Optimization.

Workflows	Constraint	MOABC			WMS			DBC		
	Deadline	Time	Cost/ Job	Jobs	Time	Cost/ Job	Jobs	Time	Cost/ Job	Jobs
Gaussian	35000	34751.61	57.91	12	30081.41	6328.33	12	35000.68	60.14	10
	30000	29862.43	198.76	12	28409.53	7119.20	10	30000.92	71.28	9
	25000	24818.09	1523.87	12	21724.07	8700.93	6	25000.27	82.42	6
Gauss-Jordan	40000	38759.33	83.75	15	33262.57	2993.61	12	38764.00	87.31	15
	37500	37007.08	251.72	15	33262.57	2993.61	12	37410.48	398.24	15
	35000	34894.49	289.36	15	33262.57	2993.61	12	35000.23	80.18	14
LU	46000	45442.98	64.91	14	44350.05	2827.31	9	45445.66	79.59	14
	38500	38488.01	230.20	14	36034.15	3118.33	6	38500.38	74.17	10
	32000	31560.51	266.40	14	24946.66	3143.28	5	32000.05	89.10	9
Find-Max	61000	60140.90	66.82	18	52728.25	210.00	11	60145.21	66.82	18
	51500	47014.72	621.80	18	45915.17	243.91	7	51500.85	83.52	12
	42000	41376.77	3735.86	18	30130.15	234.34	6	42000.41	77.77	11
FFT	43500	42098.00	69.49	15	40173.89	229.56	7	42102.02	82.63	15
	37750	36925.65	274.55	15	30130.11	267.82	3	37750.71	94.77	11
	32000	32030.00	2117.87	15	30130.11	267.82	4	32000.23	122.21	7
Stencil	29500	28735.14	100.23	12	27619.46	318.03	6	29072.86	107.49	12
	26250	25894.05	389.32	12	20086.46	401.72	4	26250.86	115.26	8
	23000	22868.49	660.75	12	20086.46	401.72	4	23000.50	128.65	8

"We simply must balance our demand for energy with our rapidly shrinking resources. By acting now we can control our future instead of letting the future control us."

Jimmy Carter

8

Results and Analysis: Energy Consumption and Responsiveness Optimization

This chapter analyses the multi-objective optimization problem for energy consumption and responsiveness following the same sections in the previous problem in order to have a homogeneous frame. First, the multi-objective algorithms proposed in this thesis are compared with the standard multi-objective algorithm NSGA-II. Later, our best multi-objective algorithm selected in the previous test is compared with MOHEFT in order to prove its quality with this strong multi-objective scheduling algorithm. And finally, a comparison with the real meta-schedulers WMS and DBC will be carried out.

8.1 Multi-Objective comparison

8.1.1 EU DataGrid and WWG combination testbed

In this section we starts by comparing the six multi-objective algorithms using hypervolume metrics with the WWG and EU DataGrid topology. Tables 8.1.1 and 8.1.2 show the median and interquartile for each algorithm and workflow. MOABC provides the best average for the hypervolume (64.69), while NSGA-II provides the worst average value (56.41). MOGSA is the second best algorithm in this test by obtaining a 63.14 average value.

Table 8.1.1: MOGSA, MOABC, and MOSWO Hypervolume properties by workflow. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Workflows	Median (Interquartile Range)			Reference Point
	MOGSA	MOABC	MOSWO	(Time (s), Power (kW))
Gaussian	62.14 (0.47)	63.79 (0.00)	60.73 (0.73)	(1300, 113)
Gauss-Jordan	67.98 (0.17)	68.94 (0.32)	67.69 (0.19)	(3000, 130)
LU	63.27 (0.43)	65.19 (0.00)	62.51 (0.55)	(2000, 120)
Find-Max	62.33 (0.15)	64.47 (0.13)	60.97 (0.62)	(2200, 220)
FFT	60.74 (0.20)	62.15 (0.60)	59.92 (0.15)	(1600, 160)
Stencil	62.38 (0.31)	63.65 (0.00)	61.60 (0.28)	(1400, 140)
Average	63.14 (0.28)	64.69 (0.17)	62.23 (0.42)	-

Table 8.1.2: MO-FA, MOBSA and NSGA-II Hypervolume properties per each workflow. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Workflows	Median (Interquartile Range)			Reference Point
	MO-FA	MOBSA	NSGA-II	(Time (s), Power (kW))
Gaussian	57.60 (1.30)	60.56 (0.68)	57.18 (1.04)	(1300, 113)
Gauss-Jordan	59.06 (1.46)	65.90 (0.72)	58.64 (1.85)	(3000, 130)
LU	56.90 (0.85)	61.71 (0.85)	53.34 (1.23)	(2000, 120)
Find-Max	56.92 (1.05)	59.93 (0.59)	57.04 (1.20)	(2200, 220)
FFT	55.34 (1.08)	59.37 (0.59)	55.60 (1.61)	(1600, 160)
Stencil	56.69 (1.40)	61.64 (0.78)	56.67 (1.79)	(1400, 140)
Average	57.08 (1.19)	61.51 (0.70)	56.41 (1.45)	-

In the boxplot (Figure 8.1.1) we can see the results graphically.

Moreover, the Pareto fronts are shown in Figure 8.1.2, in order to see the number of solutions and the coverage between them. However, we also provide the numerical values in Table 8.1.3 for clarification. Once more, MOABC covers most of the solutions obtained by the other algorithms with more than 90% in all the cases, and almost no other algorithms cover its solutions. The MOGSA is the second best algorithm, but its coverage is slower with an average of 62.52%, and the rest of algorithms covers an average of 29.11% of its solutions.

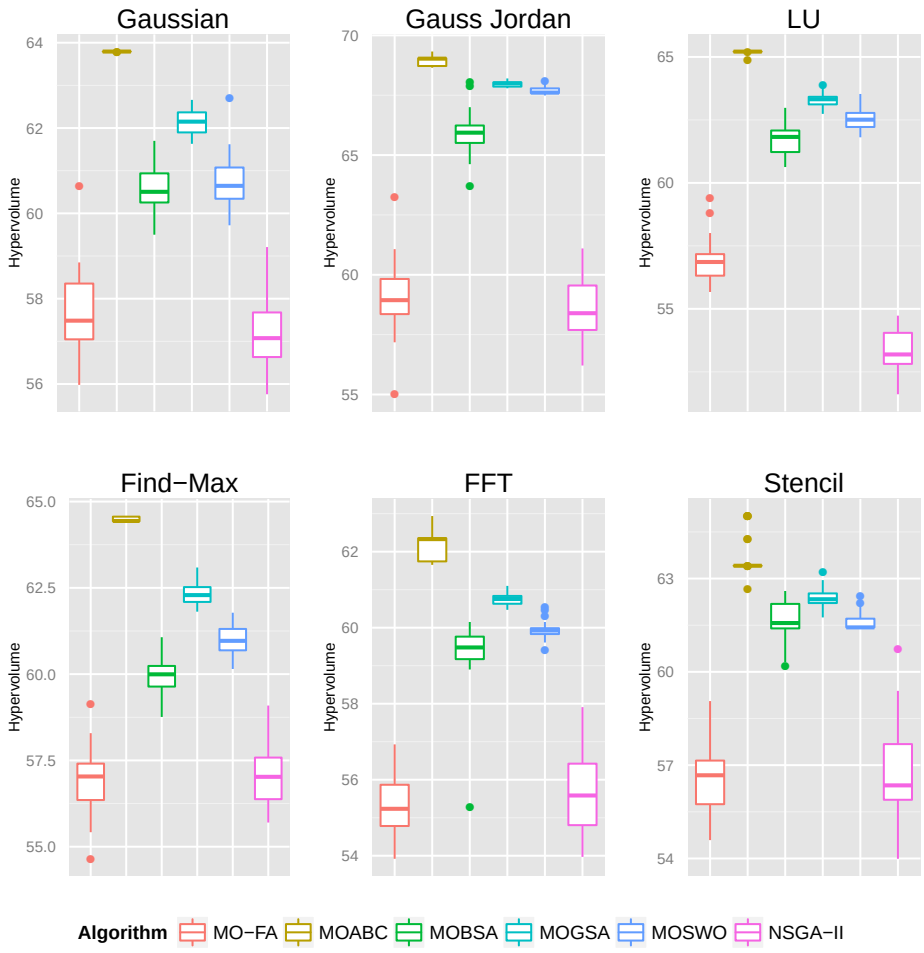

Figure 8.1.1: Boxplot Gaussian, Gauss-Jordan, LU, FindMax, FFT and Stencil workflows. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

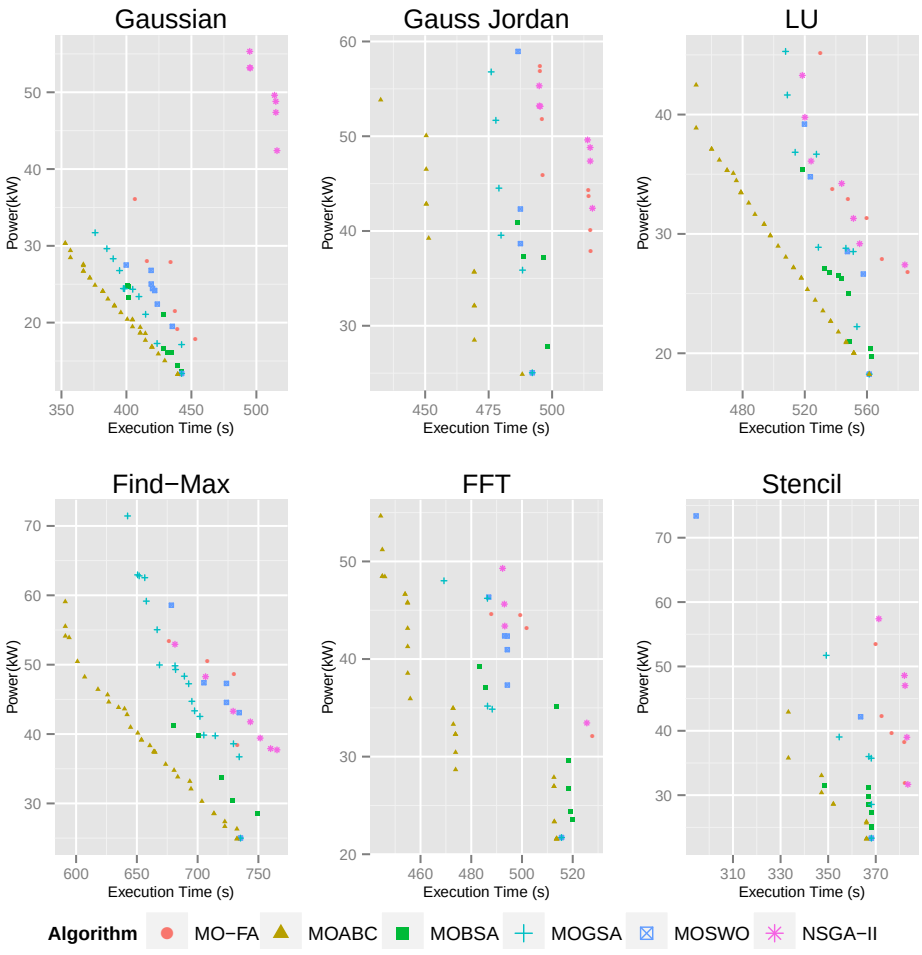

Figure 8.1.2: Pareto fronts by workflow and algorithm. In every case, from the 30 repetitions, the closest Pareto front to the median hypervolume is shown. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Table 8.1.3: Set Coverage comparison of MOGSA, MOABC, MOSWO, MO-FA, MOBSA, and NSGA-II by workflow. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Gauss-Jordan	LU	Find-Max	FFT	Stencil	
MOGSA	MOABC	0.00	0 .00	2.63	0.00	0.00	0.00	0.44
MOABC	MOGSA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MOGSA	MOSWO	100.00	75.00	40.00	83.33	83.33	33.33	69.16
MOSWO	MOGSA	8.33	16.67	22.22	0.00	0.00	33.33	13.42
MOGSA	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	MOGSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOGSA	MOBSA	44.44	100.00	22.22	20.00	57.14	14.28	43.01
MOBSA	MOGSA	25.00	0.00	44.44	36.84	20.00	66.67	32.16
MOGSA	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOGSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MOSWO	100.00	100.00	100.00	100	100	66.66	91.11
MOSWO	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MOBSA	88.89	100.00	100.00	100.00	100.00	100.00	98.15
MOBSA	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOSWO	MO-FA	50.00	75.00	66.67	75.00	75.00	60.00	66.94
MO-FA	MOSWO	0.00	0.00	0.00	33.33	0.00	0.00	5.55
MOSWO	MOBSA	11.11	50.00	22.22	20.00	42.85	14.28	26.74
MOBSA	MOSWO	50.00	50.00	60.00	50.00	66.67	66.67	57.22
MOSWO	NSGA-II	100.00	100.00	100.00	71.42	100.00	100.00	95.23
NSGA-II	MOSWO	0.00	0.00	0.00	16.66	0.00	0.00	2.77
MO-FA	MOBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOBSA	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	NSGA-II	100.00	57.14	28.57	28.57	50.00	100.00	60.71
NSGA-II	MO-FA	0.00	25.00	33.33	50.00	50.00	0.00	26.38
MOBSA	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00

To finalize the multi-objective comparison with the WWG and EU DataGrid topology, a statistical analysis is performed following the workflow in Figure 6.2.1. The statistics show that MOABC presents a significant difference among all the algorithms, being the best algorithm for this multi-objective comparison (see Table 8.1.4).

Table 8.1.4: Statistical analysis of the comparison among MOABC, MOGSA, MOSWO, MOBSA, MO-FA, and NSGA-II using hypervolume metrics. The non-significant differences are shown in this table. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

	MOGSA	MOABC	MOSWO	MO-FA	MOBSA	NSGA-II
MOGSA		-	-	-	-	-
MOABC			-	-	-	-
MOSWO				-	#Gaussian, #Stencil	-
MO-FA					-	#Gaussian, #Gauss-Jordan, #Find-Max, #FFT, #Stencil
MOBSA						-
NSGA-II						

8.1.2 IBERGRID testbed

Following a new evaluation of the multi-objective algorithms for energy consumption and responsiveness optimization is performed with the IBERGRID topology (see Chapter 5 for detailed information). First, the comparison starts with the hypervolume medians and interquartiles for each algorithm and workflow (see Tables 8.1.5 and 8.1.6).

Table 8.1.5: MOGSA, MOABC, and MOSWO Hypervolume properties by workflow. IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

Workflows	Median (Interquartile Range)			Reference Point
	MOGSA	MOABC	MOSWO	(Time (s), Power (kW))
Gaussian	58.23 (0.14)	59.70 (0.09)	57.59 (0.26)	(120000, 12000)
Gauss-Jordan	56.83 (0.00)	57.24 (0.13)	56.83 (0.00)	(180000, 18000)
LU	60.09 (0.23)	60.12 (0.26)	59.81 (0.00)	(170000, 17000)
Find-Max	55.36 (0.16)	56.95 (0.19)	55.08 (0.00)	(200000, 20000)
FFT	57.57 (0.00)	58.43 (0.24)	57.57 (0.00)	(170000, 17000)
Stencil	59.61 (0.00)	61.92 (0.28)	59.61 (0.00)	(170000, 17000)
Average	57.94 (0.08)	59.06 (0.19)	57.74 (0.04)	-

Table 8.1.6: MO-FA, MOBSA, and NSGA-II Hypervolume properties by workflow. IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

Workflows	Median (Interquartile)			Reference Point
	MO-FA	MOBSA	NSGA-II	(Time (s), Power (kW))
Gaussian	48.50 (1.43)	58.14 (0.29)	49.74(1.17)	(120000, 12000)
Gauss-Jordan	42.08 (1.09)	56.83 (0.00)	43.41 (1.48)	(180000, 18000)
LU	49.57 (0.99)	61.71(0.85)	50.86 (1.01)	(170000, 17000)
Find-Max	43.72 (1.10)	55.39 (0.24)	44.86 (1.14)	(200000, 20000)
FFT	45.07 (1.69)	57.60 (0.00)	46.01 (1.39)	(170000, 17000)
Stencil	46.57 (1.63)	59.61 (0.00)	47.83 (0.94)	(170000, 17000)
Average	45.91 (1.32)	58.21 (0.23)	47.11 (1.18)	-

Results show that MOABC provides the highest values (59.06 on average), followed by MOBSA with an average of 58.21. Although NSGA-II provides a lower average (47.11) this time, it surpasses MO-FA with an average of 45.91. A boxplot is also provided in Figure 8.1.3 to reinforce the data visualization of Tables 8.1.5 and 8.1.6.

Furthermore, the resulting Pareto fronts are displayed in Figure 8.1.4, where is visually verified that MOABC surpasses the other multi-objective algorithms. The coverage metrics is also calculated and proves that MOABC is the multi-objective algorithm that covers more solutions (see Table 8.1.7).

Figure 8.1.3: Boxplot Gaussian, Gauss-Jordan, LU, FindMax, FFT, and Stencil workflows. Testbed IBERGRID. Responsiveness and Energy Consumption.

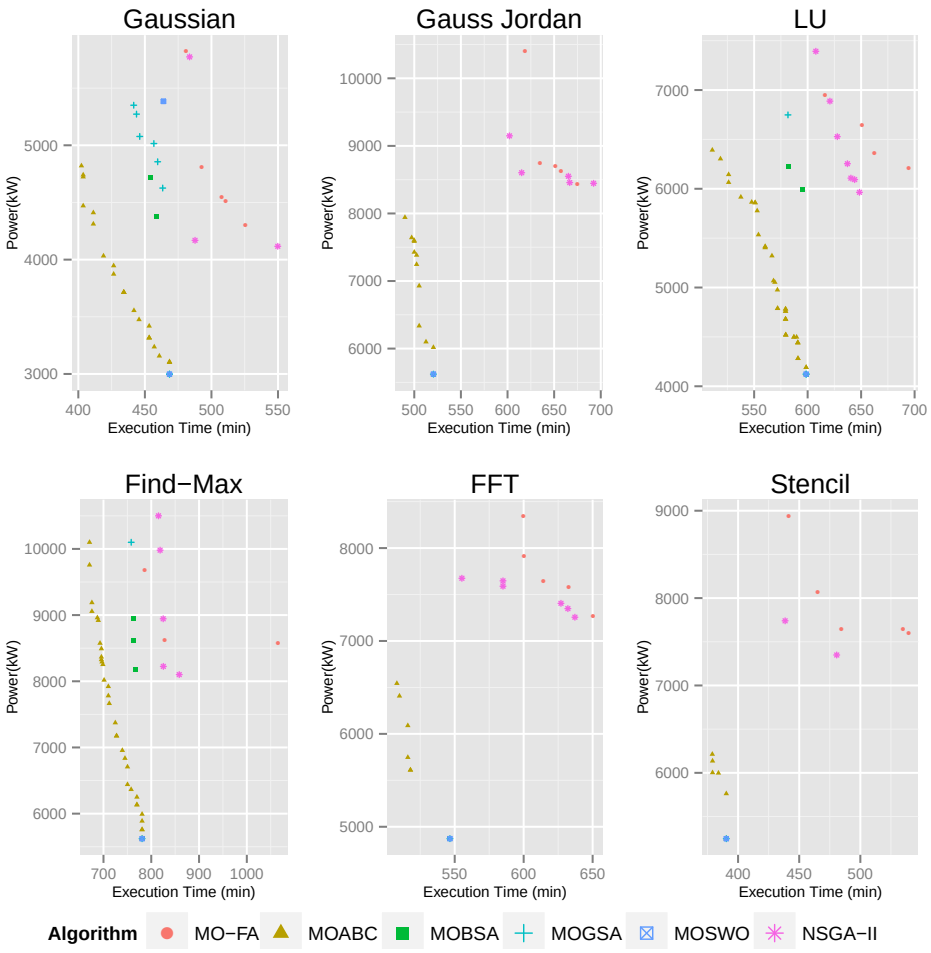

Figure 8.1.4: Pareto fronts by workflow and algorithm. In every case, from the 30 repetitions, the closest Pareto front to the median hypervolume is shown. In Gauss-Jordan, FFT, and Stencil, the solutions from MOBSA and MOGSA are overlapped by (are the same solutions as) the solutions from others of the algorithms. IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

Table 8.1.7: Set Coverage comparison of MOGSA, MOABC, MOSWO, MO-FA, MOBSA, and NSGA-II by workflow. IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Gauss-Jordan	LU	Find-Max	FFT	Stencil	
MOGSA	MOABC	4.76	7.69	3.57	0.00	0.00	12.50	4.75
MOABC	MOGSA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MOGSA	MOSWO	100.00	100.00	100.00	0.00	100.00	100.00	83.33
MOSWO	MOGSA	14.28	0.00	50.00	50.00	0.00	100.00	35.71
MOGSA	MO-FA	100.00	60.00	50.00	100.00	100.00	60.00	78.33
MO-FA	MOGSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOGSA	MOBSA	33.33	100.00	33.33	0.00	100.00	100.00	61.11
MOBSA	MOGSA	42.86	100.00	0.00	50.00	100.00	0.00	48.81
MOGSA	NSGA-II	66.67	60.00	57.14	60.00	50.00	50.00	57.30
NSGA-II	MOGSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MOSWO	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MOSWO	MOABC	4.76	0.00	3.57	0.00	0.00	12.50	3.47
MOABC	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOABC	MOBSA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MOBSA	MOABC	4.76	7.69	0.00	3.33	0.00	0.00	2.63
MOABC	NSGA-II	100.00	100.00	100.00	100.00	100.00	100.00	100.00
NSGA-II	MOABC	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOSWO	MO-FA	80.00	60.00	50.00	66.67	60.00	60.00	62.78
MO-FA	MOSWO	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOSWO	MOBSA	33.33	0.00	33.33	0.00	0.00	100.00	27.78
MOBSA	MOSWO	100.00	100.00	0.00	100.00	100.00	0.00	66.67
MOSWO	NSGA-II	100.00	60.00	57.14	60.00	50.00	50.00	62.86
NSGA-II	MOSWO	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MO-FA	MOBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MOBSA	MO-FA	100.00	100.00	100.00	100.00	100.00	100.00	100.00
MO-FA	NSGA-II	0.00	20.00	0.00	40.00	0.00	0.00	10.00
NSGA-II	MO-FA	40.00	40.00	75.00	66.67	100.00	60.00	63.61
MOBSA	NSGA-II	66.67	60.00	71.42	80.00	50.00	50.00	63.01
NSGA-II	MOBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00

In the IBERGRID topology, MOABC, again, presents significant differences from the rest of the multi-objective algorithms according to the statistics described in Figure 6.2.1. We can see that the algorithms MOGSA, MOSWO, and MOBSA do not have significant differences for many of the workflows executed in the grid (see Table 8.1.8).

Table 8.1.8: Statistical analysis of the comparison among MOABC, MOGSA, MOSWO, MOBSA, MO-FA, and NSGA-II using hypervolume metrics. The non-significant differences are shown in this table. IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

	MOGSA	MOABC	MOSWO	MO-FA	MOBSA	NSGA-II
MOGSA		-	#Gauss-Jordan, #FFT	-	#Gaussian, #Gauss-Jordan, #LU, #Find-Max, #FFT	-
MOABC			-	-	-	-
MOSWO				-	#Gauss-Jordan	-
MO-FA					-	-
MOBSA						-
NSGA-II						

8.2 MOHEFT comparison

As MOABC is the best of our multi-objective algorithms after the tests performed in last section, we compare it with one of the best multi-objective scheduling algorithms in literature, MOHEFT. First, we calculate the median hypervolume for each algorithm and workflow. Then, we can appreciate that MOABC surpasses MOHEFT in the studied topologies, WWG and EU DataGrid topology and IBERGRID, and for all the workflows (see Tables 8.2.1 and 8.2.2).

Table 8.2.1: MOABC vs. MOHEFT: Hypervolume. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Workflows	Median		Reference Point
	MOABC	MOHEFT	(Time (s), Power (kW))
Gaussian	63.79	60.28	(1300, 113)
Gauss-Jordan	68.94	60.91	(3000, 130)
LU	65.19	60.18	(2000, 120)
Find-Max	64.47	59.71	(2200, 220)
FFT	62.15	48.61	(1600, 160)
Stencil	62.38	44.17	(1400, 140)
Average	64.69	55.64	-

Table 8.2.2: MOABC vs. MOHEFT: Hypervolume. Testbed IBERGRID. Energy Consumption and Responsiveness Optimization.

Workflows	Median		Reference Point
	MOABC	MOHEFT	(Time (s), Power (kW))
Gaussian	60.69	54.14	(120000, 12000)
Gauss-Jordan	58.16	56.34	(180000, 18000)
LU	58.55	57.53	(170000, 17000)
Find-Max	57.76	50.38	(200000, 20000)
FFT	58.26	52.96	(170000, 17000)
Stencil	53.39	44.21	(170000, 17000)
Average	59.06	52.59	-

In order to reinforce this comparison, we performed the coverage metrics and observed that MOABC covers a higher percentage of solutions than MOHEFT for both topologies and all the executed workflows. The coverage results are collected in Tables 8.2.3 and 8.2.4.

Table 8.2.3: Set Coverage comparison of MOABC and MOHEFT by workflow. EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Jordan	LU	Find-Max	FFT	Stencil	
MOABC	MOHEFT	50.00	100.00	50.00	100.00	50.00	0.00	58.33
MOHEFT	MOABC	8.33	33.33	31.57	11.76	3.22	0.00	14.70

Table 8.2.4: Set Coverage comparison of MOABC and MOHEFT by workflow. IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

Coverage A $\geq$ B (%)								
Algorithm		Workflows						Average
A	B	Gaussian	Jordan	LU	Find-Max	FFT	Stencil	
MOABC	MOHEFT	50.00	50.00	50.00	50.00	50.00	0.00	41.66
MOHEFT	MOABC	28.57	49.84	28.57	40.00	16.67	0.00	27.27

Moreover, the statistical analysis shows that both algorithms are significantly different (p-value < 0.05) (see Table 8.2.5). With this whole comparison, we can conclude that MOABC provides better results than MOHEFT for the energy consumption and responsiveness optimization problem in both topologies.

Table 8.2.5: Statistical analysis of the comparison between MOABC and MOHEFT using hypervolume metrics. The non-significant differences are shown in this table. EU DataGrid and WWG combination testbed and IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

EU DataGrid and WWG combination testbed		
	MOABC	MOHEFT
MOABC		-
MOHEFT		

IBERGRID testbed		
	MOABC	MOHEFT
MOABC		-
MOHEFT		

8.3 Real grid meta-schedulers comparison

The last study consists of comparing MOABC, our best algorithm, with two real grid meta-schedulers, WMS and DBC. In Table 8.3.1, Time and Power values are obtained for each algorithm and workflows from the testbed WWG and EU DataGrid. The results show that MOABC offers the minimum energy consumption and execution time. The improvement that MOABC provides is at least 28% of reduction in the energy consumption.

Table 8.3.1: Results of MOABC, WMS, and DBC by workflow. Time (s) and Power (kW). Testbed EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Workflows	MOABC		WMS		DBC	
	Time	Power	Time	Power	Time	Power
Gaussian	439.48	13.2	482.68	42.93	480.82	24.69
Gauss-Jordan	488.24	24.75	534.70	80.06	533.41	46.29
LU	551.53	20.02	612.29	58.55	596.66	34.91
Find-Max	732.37	24.84	803.06	80.31	797.77	46.24
FFT	513.66	21.58	561.24	69.50	558.73	40.09
Stencil	366.12	23.22	400.02	74.66	399.68	43.03

Moreover, we have applied the deadline constraint in order to study how the meta-schedulers react regarding it (see Table 8.3.2). The meta-schedulers WMS and DBC, have problems with completing the workflows most of the time, while MOABC always executes all the jobs. Since the jobs are interdependent, the execution of all the jobs is usually necessary.

We carried out the same study for the IBERGRID topology and once more, MOABC provides the best results for both objectives, energy consumption and responsiveness, as can be seen in Table 8.3.3.

Table 8.3.2: Study restricted by deadline to check the jobs executed successfully (Time (s) and Power (kW)). EU DataGrid and WWG combination testbed. Energy Consumption and Responsiveness Optimization.

Workflows	Constraint	MOABC			WMS			DBC		
	Deadline	Time	Power/ Job	Jobs	Time	Power/ Job	Jobs	Time	Power/ Job	Jobs
Gaussian	460	439.48	1.10	12	460.00	4.01	10	460.43	2.31	10
	445	439.48	1.10	12	445.00	4.16	9	445.64	2.42	10
	430	429.50	1.25	12	430.00	4.16	9	430.73	2.40	9
Gauss-Jordan	530	488.24	1.65	15	530.00	5.34	14	525.71	2.99	15
	515	488.24	1.65	15	515.00	5.34	14	505.82	3.23	15
	500	488.24	1.65	15	500.00	5.34	14	500.08	3.08	14
LU	560	551.53	1.43	14	560.00	4.53	12	560.80	2.63	12
	545	541.58	1.55	14	545.00	4.90	10	545.00	2.81	12
	530	521.68	1.81	14	530.00	4.90	10	530.10	2.85	10
Find-Max	750	732.37	1.38	18	721.42	4.99	15	750.68	2.87	15
	735	732.37	1.38	18	721.42	4.99	15	735.68	2.87	15
	720	713.44	1.58	18	667.32	5.56	12	720.68	2.87	15
FFT	540	513.66	1.43	15	507.13	5.33	11	540.06	3.07	11
	525	513.66	1.43	15	507.13	5.33	11	525.06	3.07	11
	510	473.83	1.91	15	507.13	5.33	11	510.06	3.07	11
Stencil	380	366.12	1.93	12	373.48	6.65	10	379.03	3.85	12
	365	352.16	2.38	12	319.42	7.65	8	365.60	4.06	10
	350	347.20	2.75	12	319.42	7.65	8	350.70	4.41	8

Table 8.3.3: Results of MOABC, WMS, and DBC by workflow. Time (s) and Power (kW). IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

Workflows	MOABC		WMS		DBC	
	Time	Power	Time	Power	Time	Power
Gaussian	28109.49	2998.32	30081.41	4278.31	36090.01	4169.94
Gauss-Jordan	31231.00	5621.40	55437.59	15799.63	38764.00	7818.15
LU	25916.60	4122.58	63754.48	15867.68	45445.66	5733.51
Find-Max	46846.80	5621.58	72328.42	13408.56	60145.21	7818.35
FFT	32791.34	4871.40	52927.30	11620.70	42102.02	6775.80
Stencil	23422.46	5246.52	37663.20	12514.23	29072.86	7317.13

Regarding the deadline restriction, WMS and DBC still have problems with resolving all the jobs from the workflows, however MOABC always executes all of them (see Table 8.3.4). Therefore, we can confirm that MOABC also provides the best results in the IBERGRID topology for the studied datasets.

Table 8.3.4: MOABC, WMS, and DBC: Successfully executed jobs with regard to deadline variation. IBERGRID testbed. Energy Consumption and Responsiveness Optimization.

Workflows	Constraint	MOABC			WMS			DBC		
	Deadline	Time	Power/ Job	Jobs	Time	Power/ Job	Jobs	Time	Power/ Job	Jobs
Gaussian	35000	28109.49	249.86	12	30081.41	356.52	12	35000.68	390.92	10
	30000	28109.49	249.86	12	28409.53	401.08	10	30000.92	407.65	9
	25000	24676.96	359.31	12	21724.07	490.19	6	25000.27	480.03	6
Gauss-Jordan	40000	31231.00	374.76	15	33262.57	1053.30	12	38764.00	522.11	15
	37500	31231.00	374.76	15	33262.57	1053.30	12	37410.48	532.66	15
	35000	31231.00	374.76	15	33262.57	1053.30	12	35000.23	521.21	14
LU	46000	25916.60	294.47	14	44350.05	994.79	9	45445.66	410.02	14
	38500	25916.60	294.47	14	36034.15	1097.19	6	38500.38	482.12	10
	32000	25916.60	294.47	14	24946.66	1105.96	5	32000.05	495.64	9
Find-Max	61000	46846.80	312.31	18	52728.25	934.51	11	60145.21	434.35	18
	51500	46846.80	312.31	18	45915.17	1085.41	7	51500.85	542.92	12
	42000	41814.03	460.73	18	30130.15	1042.85	6	42000.41	548.59	11
FFT	43500	32791.34	324.79	15	40173.89	1021.57	7	42102.02	451.72	15
	37750	32791.34	324.79	15	30130.11	1191.82	3	37750.71	535.85	11
	32000	31074.74	374.08	15	30130.11	1191.82	3	32000.23	603.39	7
Stencil	29500	23422.46	437.21	12	27619.46	318.03	6	29072.86	609.76	12
	26250	23422.46	437.21	12	20086.46	1787.68	4	26250.86	749.21	8
	23000	22747.43	500.22	12	20086.46	1787.68	4	23000.50	842.89	8

"You know a conjurer gets no credit when once he has explained his trick and if I show too much of my method of working, you will come to the conclusion that I am a very ordinary individual after all."

Arthur Conan Doyle (The Complete Adventures of Sherlock Holmes)

9 Conclusions and Future Work

In this chapter, we present the conclusions performed after the analysis carried out in the previous chapters. Future work tasks are also proposed to continue with the multi-objective scheduling optimization in distributed environments research line.

9.1 Conclusions

In this thesis, six multi-objective algorithms from different fields (physics, biology and sociology) are evaluated to resolve the job scheduling problem in grid environments. These algorithms have been applied to optimize typical and conflicting pairs of objectives: *Execution Time and Cost/Energy Optimization*. The evaluation is performed with six workflows with interdependent jobs and two different topologies with heterogeneous resources.

First, we have compared the proposed algorithms with the standard multi-objective algorithm NSGA-II in order to evaluate their quality as multi-objective algorithms. Results indicate that MOABC is the best algorithm for all the cases tested in terms of hypervolume and coverage metrics. A statistical analysis confirms that it has significant differences with regard to the rest of algorithms. Moreover, MOABC outperforms MOHEFT, the multi-objective version of HEFT, one of the most popular algorithms for workflow scheduling (that has proved to be better than SPEA2). Furthermore, MOABC presents a clear superiority in terms of cost/energy and execution time regarding the real schedulers: WMS from the most-used European grid middleware and the well-known DBC. We think that swarm intelligence is appropriate to solve problems similar to the grid scheduling problem.

Other existing proposals (such as NSGA-II or MOHEFT) generate new solutions by applying a recombination methodology that considers only the knowledge provided by certain parent solutions (for example, the two parents in NSGA-II or the previous partial solution used in the construction phase of MOHEFT). Swarm intelligence generates new solutions considering all the information gathered by the entire swarm. This collective behaviour allows the swarm to carry out the search for high-quality solutions. MOABC goes a step further and provides three black boxes, which allow different search procedures (one for the employees, another for the onlookers, and the scout). This schema provides much more versatility than MOGSA or MO-FA, which have mathematical formulas to calculate the individual updates following the best solutions from the swarm. In the case of MOABC, the flexibility provided by the black boxes as opposed to the mathematical formulas, allow us to adjust the operators to the current problem. For example, in some cases the mathematical formulas from MOGSA and/or MO-FA are not appropriate for certain problems, and if used to update an individual it could have a negative rather than positive effect on the outcome. The black boxes avoid that situation and provide the flexibility to intelligently adapt the operators to the specific problem. The search procedures from employed and onlooker bees in MOABC improve the exploitation process. Moreover, the onlookers are applied with higher probability for a best solution; the exploitation process is more

intense for the best solutions. The exploration process is carried out by the scout bee, which re-initiates the search by assigning new starter solutions to scout bees and avoiding the stagnation. This strategy allows the algorithm to explore undiscovered regions of the search space, generating promising solutions that will be exploited by other groups of bees.

The MOABC algorithm has demonstrated superior quality performance in all the analyses carried out in this thesis.

9.2 Future Work

The next immediate step in future work would be the implementation of MOABC in the middleware gLite to see the studied improvements with regard to WMS. This line will not be reproducible as we were doing in the study of this thesis. Due to the evaluation results, we can see that the MOABC results will be more promising for the WMS, by allowing multi-objective optimization with the confidence that this algorithm performs better than those presented in current literature to optimize the job scheduling problem.

The algorithms presented in this thesis are independent from any distributed platform. Therefore, multiple lines can be continued with the current research. In fact, cloud computing is an infrastructure where job scheduling is still an issue to optimize; MOABC and the rest of algorithms can be studied using CloudSim (a platform created by the authors of GridSim) to later implement the best algorithm for a cloud infrastructure. In this case both cost and energy optimization are interesting problems, with regards to the cloud business model. Moreover, another well-known distributed platform, Hadoop, also needs to optimize the job scheduling problem. Overall this line would benefit from optimizing energy consumption and execution time during the map-reduce process for the tasks involved.

Another point to address is the optimization of more than two objectives in distributed systems. The multi-objective approaches presented in this thesis are flexible enough to add more objectives for their optimization. The corresponding objective functions and their respective heuristics during the exploration and ex-

ploitation processes would need to be added. Conflicting objectives, such as load balancing with regard to cost/energy and execution time, would be considered in order to resolve the job scheduling problem in distributed systems.

"And, when you want something, all the universe conspires in helping you to achieve it."

Paulo Coelho (The Alchemist)

A

Scientific publications arising from this PhD Thesis

All the scientific publications achieved during the research period of this PhD thesis are collected and classified in this chapter.

A.1 International Journals

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez, Francisco Prieto-Castrillo. "Meta-Schedulers for Grid Computing based on Multi-Objective Swarm Algorithms". Applied Soft Computing, Elsevier Science, Amsterdam, Netherlands, Vol. 13, Issue 4, 2013, pp. 1567-1582, ISSN: 1568-4946. (Impact Factor = 2.679 in 2013, Q1)
- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Multi-Objective Energy

Optimization in Grid Systems from a Brain Storming Strategy". Soft Computing, Springer, New York, USA, 2014, pp. 1-14, ISSN: 1432-7643, DOI: 10.1007/s00500-014-1474-7. (Impact factor = 1.304 in 2013, Q2)

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Swarm Approach based on Gravity for Optimizing Energy Savings in Grid Systems". Journal of Heuristics, Springer, Dordrecht, The Netherlands, 2014, pp. 1-27, ISSN: 1381-1231, DOI: 10.1007/s10732-014-9253-2. (Impact factor = 1.359 in 2013, Q2)

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Multiobjective Small-World Optimization for Energy Saving in Grid Environments". Computer Journal, Oxford University Press, Oxford, England, 2014, pp. 1-14, ISSN: 0010-4620, DOI: 10.1093/comjnl/bxu045. (Impact factor = 0.888 in 2013, Q2)

- Francisco Prieto-Castrillo, Antonio Astillero, Elena Rey-Espada, María Botón-Fernández, María Arsuaga-Ríos. "Application workflow deployments on computing grids with a complex network topology". International Journal of Complex Systems in Science, vol.1 (2), pp. 89-95, 2011. ISSN: 2174-6036.

A.2 International Book Chapters

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Multi-objective Grid Scheduling", in: "Automated Scheduling and Planning. From Theory to Practice. SCI 505". Springer-Verlag, Berlin Heidelberg, Germany, 2013, pp. 225-249. ISBN: 978-3-642-39303-7.

A.3 International IEEE Conferences

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Cost Optimization based on Brain Storming for Grid Scheduling". Proceedings of the Fourth Inter-

national Conference on Innovative Computing Technology, IEEE Computer Society, Luton, United Kingdom, 2014, pp. 31-36. ISBN: 978-1-4799-4233-6.

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Energy Optimization for Task Scheduling in Distributed Systems by an Artificial Bee Colony Approach". Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC), IEEE Computer Society, Porto, Portugal, 2014, pp. 127-132. ISBN: 978-1-4799-5937-2.

- María Arsuaga-Ríos, Francisco Prieto-Castrillo, Miguel A. Vega-Rodríguez. "Multiobjective Optimization Comparison-MOSWO vs MOGSA-– for solving the Job Scheduling Problem in Grid Environments". 10th IEEE International Symposium on Parallel and Distributed Processing with Applications, IEEE Computer Society, Madrid, Spain, 2012, pp. 570-575. ISBN: 978-0-7695-4701-5.

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez, Francisco Prieto-Castrillo. "Evaluation of Multiobjective Swarm Algorithms for Grid Scheduling". 11th International Conference on Intelligent Systems Design and Applications, IEEE Computer Society, Córdoba, Spain, 2011, pp. 1104-1109. ISBN: 978-1-4577-1675-1 (**Best Student Paper Award**).

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez, Francisco Prieto-Castrillo. "Multi-objective Artificial Bee Colony for Scheduling in Grid Environments". Proceeding of the 2011 IEEE Swarm Intelligence Symposium (SIS 2011), IEEE Computational Intelligence Society, Paris, France, 2011, pp. 206-212. ISBN: 978-1-61284-051-2.

A.4 International LNCS Conferences

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Multi-objective Proposal based on Firefly Behaviour for Green Scheduling in Grid Systems", in: "Adap-

tive and Natural Computing Algorithms". Lecture Notes in Computer Science, vol. 7824. Springer-Verlag, 2013, pp. 70-79. ISBN: 978-3-642-37212-4.

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Multi-Objective Firefly Algorithm for Energy Optimization in Grid Environments", in: "Swarm Intelligence". Lecture Notes in Computer Science, vol. 7461. Springer-Verlag, 2012, pp. 350-351. ISBN: 978-3-642-32649-3.

- María Arsuaga-Ríos, Francisco Prieto-Castrillo, Miguel A. Vega-Rodríguez. "Small-World Optimization applied to Job Scheduling on Grid Environments from a Multi-Objective perspective", in: "Application of Evolutionary Computation". Lecture Notes in Computer Science, vol. 7248. Springer-Verlag, 2012, pp. 42-51. ISBN: 978-3-642-29177-7.

A.5 Other International Conferences

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez, Francisco Prieto-Castrillo. "Multi-Objective Swarm Optimization for Grid Scheduling". 7th European Conference on Complex Systems (ECCS 2010), The Complex Systems Society, Lisbon, Portugal, 2010, pp. 1-2.

- Francisco Prieto-Castrillo, Antonio Astillero, Elena Rey-Espada, María Botón-Fernández, María Arsuaga-Ríos. "Application Workflow Deployments on Computing Grids". Net-Works International Conference, El Escorial, Madrid, Spain, 2011.

A.6 National Conferences

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Optimización Energética basada en el Comportamiento de la Colonia de Abejas para la Planificación de Trabajos en la Grid". Actas de las XXV Jornadas de Paralelismo (JP 2014), Valladolid, Spain, 2014, pp. 109-116. ISBN: 978-84-697-0329-3.

- María Arsuaga-Ríos, Miguel A. Vega-Rodríguez. "Optimización Multiobjetivo para la Planificación de Tareas en la Grid a través de la Tormenta de Ideas". Actas de las XXV Jornadas de Paralelismo (JP 2014), Valladolid, Spain, 2014, pp. 117-123. ISBN: 978-84-697-0329-3.

"True optimization is the revolutionary contribution of modern research to decision processes. "

George Bernhard Dantzig

B

Other scientific achievements during the PhD Thesis

Different scientific achievements are performed during this PhD thesis and they are enumerated and classified in this chapter.

B.1 International Stays

- The European Organization for Nuclear Research, known as CERN is a European research organization whose purpose is to operate the world's largest particle physics laboratory[1], Switzerland. April 2012 – July 2012.

[1] www.cern.ch/.

B.2 Mentions and Awards

- BEST STUDENT PAPER AWARD: IEEE 11th International Conference on Intelligent Systems Design and Applications 2011 Paper: "Evaluation of Multiobjective Swarm Algorithms for Grid Scheduling".

B.3 Reviewer of International Journals

- Applied Soft Computing, Elsevier, ISSN: 1568-4946 (Impact Factor = 2.679 in 2013, Q1).
- Soft Computing, Springer, ISSN: 1432-7643 (Impact factor = 1.304 in 2013, Q2).
- Future Generation Computer Systems, Elsevier, ISSN: 0167-739X (Impact Factor 2.639, Q1).

B.4 Participation in International Conferences

- Programme Committee: PBio 2014[2] (International Workshop on Parallelism in Bioinformatics): IEEECluster 2014[3]
- Programme Committee: EVOstar Conference 2014[4] EvoINDUSTRY: The application of Nature-Inspired Techniques in industrial settings.
- Programme Committee: PBio 2013[5] (International Workshop on Parallelism in Bioinformatics): EuroMPI 2013[6]

[2] http://arco.unex.es/mavega/pbio2014/.
[3] http://www.cluster2014.org.
[4] http://www.evostar.org/2014/.
[5] http://arco.unex.es/mavega/pbio2013/.
[6] http://www.arcos.inf.uc3m.es/eurompi2013/Home.shtml.

- Programme Committee: EVOstar Conference 2013[7] EvoINDUSTRY: The application of Nature-Inspired Techniques in industrial settings.
- Programme Committee: International Workshop on AstroParticles Physics Advanced Computing 2012. IEEE ISPA 2012.

B.5 Participation in Research Projects

- Collaboration in the i3media project[8] with MediaPro and Telefonica for optimizing the adverstisement impact in media, University of Murcia, Spain. 2008 – 2009.
- Collaboration in the CERN-openlab programme [9] with Huawei for researching different cloud storage approaches. CERN, Switzerland. 2014 – Present.

B.6 Collaboration with other Institutions

- Centre for Energy-Related, Environmental and Technological Research, (CIEMAT)[10], Spain. 2010 – 2012.
- European Organization for Nuclear Research (CERN)[1], Switzerland. 2012 – Present.

B.7 Research Grants

- María Arsuaga-Ríos has been supported by the research grant FPI "Subject nº 6: Development of multi-objective optimization strategies for metaschedulers in GRID environments" published in Official State Gazette (BOE) nº 237 from the CIEMAT [10], Spain. Granted period 2011 – 2015. Rejected for a contract at CERN in 2012.

[7] http://www.evostar.org/2013/.
[8] http://i3media.barcelonamedia.org/.
[9] http://openlab.web.cern.ch/competence-centre/platform
[10] www.ciemat.es/

B.8 Teaching

- Master on Grid Computing and Parallelism (official postgraduate degree), University of Extremadura, Spain: Member of the teaching staff, giving theory and lab practices of the course "Fundamentals of Grid Computing", with the following teaching load:
  - Academic year 2011/2012: 1.5 ECTS (15 attendance hours).
  - Academic year 2010/2011: 1.2 ECTS (12 attendance hours).
- Technical Trainning Programme, European Organization for Nuclear Research (CERN), Switzerland: Member of the trainning staff, giving theory and lab practices of the courses "Distributed Front-End Software Architecture for the CERN Accelerators System" (8 Introduction and 8 Advanced courses) for more than 120 participants, with the following teaching load:
  - Introduction Courses: Academic year 2013: 12.8 ETCS (128 attendance hours).
  - Advanced Courses: Academic year 2013 : 6.4 ETCS (64 attendance hours).

References

[1] Abramson, D., Buyya, R., and Giddy, J. A computational economy for grid computing and its implementation in the nimrod-g resource broker. *Future Generation Computer Systems 18*, 8 (2002), 1061–1074.

[2] Amorim, P., Günther, H.-O., and Almada-Lobo, B. Multi-objective integrated production and distribution planning of perishable products. *International Journal of Production Economics 138*, 1 (2012), 89 – 101.

[3] Arsuaga-Ríos, M., Prieto-Castrillo, F., and Vega-Rodríguez, M. A. Multiobjective optimization comparison-moswo vs mogsa-for solving the job scheduling problem in grid environments. In *Parallel and Distributed Processing with Applications (ISPA)* (Leganes, Spain, 10-13 July 2012), IEEE 10th International Symposium, pp. 570–575.

[4] Arsuaga-Ríos, M., Prieto-Castrillo, F., and Vega-Rodríguez, M. A. Small-world optimization applied to job scheduling on grid environments from a multi-objective perspective. In *Applications of Evolutionary Computation* (Málaga, Spain, 11-13 April 2012), C. Chio and et al., Eds., vol. 7248 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 42–51.

[5] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Multi-objective firefly algorithm for energy optimization in grid environments. In *Swarm Intelligence* (Brussels, Belgium, 12-14 September 2012), M. Dorigo et al., Ed., vol. 7461 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 350–351.

[6] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Multi-objective grid scheduling. In *Automated Scheduling and Planning*, A. S. Uyar, E. Ozcan, and N. Urquhart, Eds., vol. 505 of *Studies in Computational Intelligence*. Springer Berlin Heidelberg, 2013, pp. 225–249.

[7] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. A multi-objective proposal based on firefly behaviour for green scheduling in grid systems. In *Adaptive and Natural Computing Algorithms* (Lausanne, Switzerland, 4-6 April 2013), M. Tomassini et al., Ed., vol. 7824 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 70–79.

[8] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Cost optimization based on brain storming for grid scheduling. In *Proceedings of the Fourth International Conference on Innovative Computing Technology* (Luton, United Kingdom, 13-15 August 2014), IEEE Computer Society, pp. 31–36.

[9] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Energy optimization for task scheduling in distributed systems by an artificial bee colony approach. In *Sixth World Congress on Nature and Biologically Inspired Computing (NaBIC)* (Porto, Portugal, 30-31 July 2014), A. Madureira et al., Ed., IEEE Computer Society, pp. 127–132.

[10] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Multiobjective small-world optimization for energy saving in grid environments. *The Computer Journal* (2014), 1–14, doi: 10.1093/comjnl/bxu045.

[11] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Optimización energética basada en el comportamiento de la colonia de abejas para la planificación de trabajos en la grid. In *Actas de las XXV Jornadas de Paralelismo (JP 2014)* (Valladolid, Spain, 17-19 September 2014), pp. 109–116.

[12] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Optimización multiobjetivo para la planificación de tareas en la grid a través de la tormenta de ideas. In *Actas de las XXV Jornadas de Paralelismo (JP 2014)* (Valladolid, Spain, 17-19 September 2014), pp. 117–123.

[13] Arsuaga-Ríos, M., and Vega-Rodríguez, M. A. Swarm approach based on gravity for optimizing energy savings in grid systems. *Journal of Heuristics* (2014), 1–27, doi: 10.1007/s10732-014-9253-2.

[14] Arsuaga-Ríos, M., Vega-Rodríguez, M. A., and Prieto-Castrillo, F. Evaluation of multiobjective swarm algorithms for grid scheduling. In *Intelligent Systems Design and Applications (ISDA)* (Córdoba, Spain, 22-24 November 2011), 11th IEEE International Conference, pp. 1104–1109.

[15] Arsuaga-Ríos, M., Vega-Rodríguez, M. A., and Prieto-Castrillo, F. Multi-objective artificial bee colony for scheduling in grid environments. In *Swarm Intelligence (SIS)* (France, 11-15 April 2011), IEEE Symposium, pp. 206-212.

[16] Arsuaga-Ríos, M., Vega-Rodríguez, M. A., and Prieto-Castrillo, F. Meta-schedulers for grid computing based on multi-objective swarm algorithms. *Applied Soft Computing 13*, 4 (2013), 1567–1582.

[17] Bartle, R. G. *The Elements of Integration and Lebesgue Measure.* Wiley Classics Library. Wiley, 2011.

[18] Bodenstein, C. Heuristic scheduling in grid environments: Reducing the operational energy demand. In *Economic Models and Algorithms for Distributed Systems*, D. Neumann, M. Baker, J. Altmann, and O. Rana, Eds., Autonomic Systems. Birkhäuser Basel, Basel, 2010, pp. 239–256.

[19] Boudet, V. Heterogeneous task scheduling: a survey. Research Report RR-6895, INRIA, France, 2001.

[20] Buyya, R. Gridsim. http://www.buyya.com/gridsim/, 2001. Visited on 2014-06-01.

[21] Buyya, R., and Murshed, M. Gridsim: a toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *Concurrency and Computation: Practice and Experience 14* (2002), 1175–1220.

[22] Buyya, R., Murshed, M., and Abramson, D. A deadline and budget constrained cost-time optimisation algorithm for scheduling task farming applications on global grids. In *Int. Conf. on Parallel and Distributed Processing Techniques and Applications* (Las Vegas, USA, 24-27 June 2002), CSREA Press, pp. 2183–2189.

[23] Castro, C., Crawford, B., and Monfroy, E. A genetic local search algorithm for the multiple optimisation of the balanced academic curriculum problem. In *Cutting-Edge Research Topics on Multiple Criteria Decision Making*, Y. Shi, S. Wang, Y. Peng, J. Li, and Y. Zeng, Eds., vol. 35 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg, Chengdu/Jiuzhaigou, China, 21-26 June 2009, pp. 824–832.

[24] Chakraborty, P., Das, S., Roy, G. G., and Abraham, A. On convergence of the multi-objective particle swarm optimizer. *Information Sciences 181*, 8 (2011), 1411–1425.

[25] Coello, C., Gary, L., and van Veldhuizen, D. A. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Genetic and Evolutionary Computation. Springer-Verlag New York, 2006.

[26] CompuGreen. The green 500. http://www.green500.org, 2007. Visited on 2014-06-01.

[27] Côté, P., Wong, T., and Sabourin, R. A hybrid multi-objective evolutionary algorithm for the uncapacitated exam proximity problem. In *5th International Conference on Practice and Theory of Automated Timetabling* (Pittsburgh, PA, USA, 18-20 August 2004), E. Burke and M. Trick, Eds., vol. 3616 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 294–312.

[28] Deb, K. *Multi-Objective Optimization using Evolutionary Algorithms*. John John Wiley & Sons, 2001.

[29] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. A fast elitist multi-objective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation 6*, 2 (2002), 182–197.

[30] The cloud computing and distributed systems (clouds) laboratory. http://www.cloudbus.org/intro.html, 2007. Dept. of Computer Science and Software Engineering at University of Melbourne. Visited on 2014-06-01.

[31] Diaz, C. O., Guzek, M., Pecero, J. E., Bouvry, P., and Khan, S. U. Scalable and energy-efficient scheduling techniques for large-scale systems. In *Proceedings of the 2011 IEEE 11th International Conference on Computer and Information Technology* (Washington, DC, USA, 31 August - 2 September 2011), CIT '11, IEEE Computer Society, pp. 641–647.

[32] Dolz, M. F., Fernández, J. C., Iserte, S., Mayo, R., Quintana, E. S., Cotallo, M. E., and Díaz, G. Energysaving cluster experience in cetaciemat. In *5th Iberian Grid Infrastructure Conference* (Santander, Spain, 8-10 Junio 2011), IBERGRID, Ed., Enabling Grids for E-sciencE (egee), pp. 39–50.

[33] Du, H., Wu, X., and Zhuang, J. Small-world optimization algorithm for function optimization. In *Advances in Natural Computation*, L. Jiao, L. Wang, X. Gao, J. Liu, and F. Wu, Eds., vol. 4222 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Xi'an, China, 24-28 September 2006, pp. 264–273.

[34] Durillo, J., Nae, V., and Prodan, R. Multi-objective workflow scheduling: An analysis of the energy efficiency and makespan tradeoff. In *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on* (Delft, 13-16 May 2013), IEEE Computer Society, pp. 203–210.

[35] Durillo, J., Nae, V., and Prodan, R. Multi-objective energy-efficient workflow scheduling using list-based heuristics. *Future Generation Computer Systems 36* (2014), 221–236.

[36] Durillo, J., and Prodan, R. Multi-objective workflow scheduling in amazon ec2. *Cluster Computing 7*, 2 (2013), 169–189.

[37] Durillo, J., Prodan, R., and Fard, H. Moheft: A multi-objective list-based method for workflow scheduling. In *Proceedings of the 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)* (Taipei, 3-6 December 2012), CLOUDCOM '12, IEEE Computer Society, pp. 185–192.

[38] EGEE-Project. glite - lightweight middleware for grid computing. http://glite.cern.ch/, 2008. Visited on 2014-06-01.

[39] Entezari-Maleki, R., and Movaghar, A. A probabilistic task scheduling method for grid environments. *Future Gener. Comput. Syst. 28*, 3 (2012), 513–524.

[40] Fard, H., Prodan, R., Durillo, J., and Fahringer, T. A multi-objective approach for workflow scheduling in heterogeneous environments. In *CCGRID* (Ottawa, ON, 13-16 May 2012), IEEE Computer Society, pp. 300–309.

[41] Foster, I., and Kesselman, C. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[42] Foster, I., Kesselman, C., and Tuecke, S. The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications 15*, 3 (2001), 200–222.

[43] GridTalk. A greener way? grids and green computing. http://www.gridtalk.org/Documents/gridsandgreen.pdf, 2009. Visited on 2014-06-01.

[44] GRyCAP-UPV. Clues: cluster energy saving (for hpc and cloud computing). http://www.grycap.upv.es/clues/eng/index.php, 2010. Visited on 2014-06-01.

[45] Hamta, N., Ghomi, S. F., Jolai, F., and Shirazi, M. A. A hybrid pso algorithm for a multi-objective assembly line balancing problem with flexible operation times, sequence-dependent setup times and learning effect. *International Journal of Production Economics 141*, 1 (2013), 99 – 111.

[46] Hernández, C. J. B., Sierra, D. A., Varrette, S., and Pacheco, D. L. Energy efficiency on scalable computing architectures. In *Proceedings of the 2011 IEEE 11th International Conference on Computer and Information Technology* (Washington, DC, USA, 31 August - 2 September 2011), CIT '11, IEEE Computer Society, pp. 635–640.

[47] INFN, ELSAG-DATAMAT, and CESNET. glite wms : Workload management system. http://web.infn.it/gLiteWMS/, 2008. Visited on 2014-06-01.

[48] Ismayilova, N. A., Sagir, M., and Gasimov, R. N. A multiobjective faculty-course-time slot assignment problem with preferences. *Mathematical and Computer Modelling 46*, 7-8 (2007), 1017 – 1029.

[49] Karaboga, D., and Basturk, B. Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. In *IFSA '07: Proceedings of the 12th international Fuzzy Systems Association world congress on Foundations of Fuzzy Logic and Soft Computing* (Cancun, Mexico, 18-21 June 2007), P. Melin et al., Ed., vol. 4529 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 789–798.

[50] Karaboga, D., and Basturk, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *J. of Global Optimization 39*, 3 (2007), 459–471.

[51] Kennedy, J., and Eberhart, R. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks* (Perth, WA, 27 November - 1 December 1995), vol. 4, IEEE, pp. 1942–1948.

[52] Khan, S. U. A goal programming approach for the joint optimization of energy consumption and response time in computational grids. In *28th IEEE International Performance Computing and Communications Conference* (Scottsdale, AZ, 14-16 December 2009), IEEE Computer Society, pp. 410–417.

[53] Khan, S. U. A multi-objective programming approach for resource allocation in data centers. In *International conference on parallel and distributed processing techniques and applications (PDPTA)* (Las Vegas, USA, 13-16 July 2009), CSREA Press, pp. 152–158.

[54] Khan, S. U., and Ahmad, I. A cooperative game theoretical technique for joint optimization of energy consumption and response time in computational grids. *IEEE Trans. Parallel Distrib. Syst. 20*, 3 (2009), 346–360.

[55] Khare, V., Yao, X., and Deb, K. Performance scaling of multi-objective evolutionary algorithms. In *Evolutionary Multi-Criterion Optimization* (Faro, Portugal, 8-11 April 2003), C. Fonseca, P. Fleming, E. Zitzler, L. Thiele, and K. Deb, Eds., vol. 2632 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 376–390.

[56] Kleinberg, J. The small-world phenomenon: an algorithm perspective. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing* (New York, NY, USA, 2000), STOC '00, ACM, pp. 163–170.

[57] Kruskal, W. H., and Wallis, W. A. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association 47*, 260 (1952), 583–621.

[58] Kundu, D., Suresh, K., Ghosh, S., Das, S., Panigrahi, B., and Das, S. Multi-objective optimization with artificial weed colonies. *Information Sciences 181*, 12 (2011), 2441–2454.

[59] Lee, Y. C., and Zomaya, A. Y. Energy conscious scheduling for distributed computing systems under different operating conditions. *IEEE Trans. Parallel Distrib. Syst. 22*, 8 (2011), 1374–1381.

[60] Lee, Y.-H., Leu, S., and Chang, R.-S. Improving job scheduling algorithms in a grid environment. *Future Gener. Comput. Syst. 27*, 8 (2011), 991–998.

[61] Lei, D. Multi-objective production scheduling: a survey. *The International Journal of Advanced Manufacturing Technology 43*, 9-10 (2009), 926–938.

[62] Levene, H. Robust tests for equality of variances. In *Contributions to probability and statistics essays in honor of Harold Hotelling*, I. Olkin et al., Ed. Stanford Univ. Press., 1960, pp. 278–292.

[63] Li, J., Burke, E. K., Curtois, T., Petrovic, S., and Qu, R. The falling tide algorithm: A new multi-objective approach for complex workforce scheduling. *Omega 40*, 3 (2012), 283 – 293.

[64] Li, X., Zhang, J., Wang, S., Li, M., and Li, K. A small world algorithm for high-dimensional function optimization. In *Proceedings of the 8th IEEE international conference on Computational intelligence in robotics and automation* (Daejeon, Korea, 15-18 December 2009), CIRA'09, IEEE Press, pp. 55–59.

[65] Lindberg, P., Leingang, J., Lysaker, D., Khan, S. U., and Li, J. Comparison and analysis of eight scheduling heuristics for the optimization of energy consumption and makespan in large-scale distributed systems. *The Journal of Supercomputing 59*, 1 (2012), 323–360.

[66] Liu, W., Li, H., Du, W., and Shi, F. Energy-aware task clustering scheduling algorithm for heterogeneous clusters. In *Proceedings of the 2011 IEEE/ACM International Conference on Green Computing and Communications* (Washington, DC, USA, 4-5 August 2011), GREENCOM '11, IEEE Computer Society, pp. 34–37.

[67] Loukil, T., Teghem, J., and Fortemps, P. A multi-objective production scheduling case study solved by simulated annealing. *European Journal of Operational Research 179*, 3 (2007), 709 – 722.

[68] Lovasz, G., Berl, A., and De Meer, H. Energy-efficient and performance-conserving resource allocation in data centers. In *Proc. of the COST Action IC0804 on Energy Efficiency in Large Scale Distributed Systems - 2nd Year* (Budapest, May 2011), IRIT, pp. 31–35.

[69] Mansouri, S. A., Gallear, D., and Askariazad, M. H. Decision support for build-to-order supply chain management through multiobjective optimization. *International Journal of Production Economics 135*, 1 (2012), 24 – 36.

[70] Mao, W., Yan, G., Dong, L., and Hu, D. Model selection for least squares support vector regressions based on small-world strategy. *Expert Syst. Appl. 38*, 4 (2011), 3227–3237.

[71] Milgram, S. The small world problem. *Psychology Today 2*, 1 (1967), 60–67.

[72] Mobasher, A. *Nurse Scheduling Optimization in a General Clinic and an Operating Suite*. PhD thesis, University of Houston, 2012.

[73] Moudani WE, Cosenza CAN, C., de Coligny, M., and Mora-Camino, F. A bi-criterion approach for the airlines crew rostering problem. In *Evolutionary Multi-Criterion Optimization*, E. Zitzler, L. Thiele, K. Deb, C. Coello Coello, and D. Corne, Eds., vol. 1993 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Zurich, Switzerland, 7-9 March 2001, pp. 486–500.

[74] Pinedo, M. L. *Scheduling: Theory, Algorithms, and Systems*, 4th ed. Prentice-Hall, 2012.

[75] Pitsoulis, L. S., and Resende, M. G. C. Greedy randomized adaptive search procedures. In *Handbook of Applied Optimization* (2001), Oxford University Press, pp. 168–183.

[76] Pop, F. Optimization of resource control for transitions in complex systems. *Mathematical Problems in Engineering 2012* (2012), 12.

[77] Rashedi, E., Nezamabadi-pour, H., and Saryazdi, S. Gsa: A gravitational search algorithm. *Information Sciences 179*, 13 (2009), 2232 – 2248.

[78] Sheskin, D. *Handbook of Parametric and Nonparametric Statistical Procedures*, 5th ed. Chapman & Hall/CRC Press, New York, NY, USA, 2011.

[79] Shi, Y. Brain storm optimization algorithm. In *Proceedings of the Second international conference on Advances in swarm intelligence - Volume Part I* (Chongqing, China, 12-15 June 2011), Y. Tan et al., Ed., vol. 6728, Springer Berlin Heidelberg, pp. 303–309.

[80] Shi, Y. An optimization algorithm based on brainstorming process. *International Journal of Swarm Intelligence Research* 2, 4 (2011), 35–62.

[81] Silva, A., and Burke, E. K. A tutorial on multiobjective metaheuristics for scheduling and timetabling. In *Multiple Objective Meta-Heuristics* (2004), X. Gandibleux, M. Sevaux, K. Sörensen, and V. T'Kindt, Eds., vol. 535 of *Lecture Notes in Economics and Mathematical Systems*, Springer Berlin Heidelberg.

[82] Silva, A., Burke, E. K., and Petrovic, S. An introduction to multiobjective metaheuristics for scheduling and timetabling. In *Metaheuristic for Multiobjective Optimisation, Lecture Notes in Economics and Mathematical Systems* (2004), X. Gandibleux et al., Ed., vol. 535 of *Lecture Notes in Economics and Mathematical Systems*, Springer Berlin Heidelberg, pp. 91–129.

[83] Strogatz, S. H. Exploring complex networks. *Nature 410*, 6825 (2001), 268–276.

[84] Sulistio, A., Poduval, G., Buyya, R., and Tham, C. On incorporating differentiated levels of network service into gridsim. *Future Generation Computer Systems 23*, 4 (2007), 606–615.

[85] Talukder, A. K. M. K. A., Kirley, M., and Buyya, R. Multiobjective differential evolution for scheduling workflow applications on global grids. *Concurrency and Computation: Practice and Experience. 21*, 13 (2009), 1742–1756.

[86] Topcuoglu, H., Hariri, S., and Wu, M.-Y. Performance-effective and low-complexity task scheduling for heterogeneous computing. *Parallel and Distributed Systems, IEEE Transactions on 13*, 3 (2002), 260–274.

[87] Tsai, M.-Y., Chiang, P.-F., Chang, Y.-J., and Wang, W.-J. Heuristic scheduling strategies for linear-dependent and independent jobs on heterogeneous grids. In *Grid and Distributed Computing*, T.-h. Kim, H. Adeli, H.-s. Cho, O. Gervasi, S. S. Yau, B.-H. Kang, and J. G. Villalba, Eds., vol. 261 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg, Island, Korea, 8-10 December 2011, pp. 496–505.

[88] Tsuchiya, T., Osada, T., and Kikuno, T. Genetics-based multiprocessor scheduling using task duplication. *Microprocessors and Microsystems 22*, 3-4 (1998), 197 – 207.

[89] Wang, L., von Laszewski, G., Dayal, J., and Wang, F. Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with dvfs. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing* (Melbourne, VIC, 17-20 May 2010), CCGRID '10, IEEE Computer Society, pp. 368–377.

[90] Wang, X., Cai, S., and Huang, M. A small-world optimization algorithm based and abc supported qos unicast routing scheme. In *Proceedings of the 2007 IFIP international conference on Network and parallel computing* (Dalian, China, 18-21 September 2007), K. Li, C. Jesshope, H. Jin, and J.-L. Gaudiot, Eds., vol. 4672 of *NPC'07*, Springer Berlin Heidelberg, pp. 242–249.

[91] Watts, D. J., and Strogatz, S. H. Collective dynamics of 'small-world' networks. *Nature 393*, 6684 (1998), 440–442.

[92] Wieczorek, M., Hoheisel, A., and Prodan, R. Towards a general model of the multi-criteria workflow scheduling on the grid. *Future Generation Computer Systems 25*, 3 (2009), 237–256.

[93] Wieczorek, M., Prodan, R., and Fahringer, T. Scheduling of scientific workflows in the askalon grid environment. *ACM SIGMOD Record 34*, 3 (2005), 56–62.

[94] Wu, M.-Y., and Gajski, D. D. Hypertool: A programming aid for message-passing systems. *IEEE Transactions on Parallel and Distributed Systems 1*, 3 (1990), 330–343.

[95] Xhafa, F., and Abraham, A. Computational models and heuristic methods for grid scheduling problems. *Future Generation Computer Systems 26*, 4 (2010), 608 – 621.

[96] Xiong, J., Xing, L., and Chen, Y. Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. *International Journal of Production Economics 141*, 1 (2013), 112–126.

[97] Yang, X.-S. Firefly algorithms for multimodal optimization. In *SAGA*, O. Watanabe and T. Zeugmann, Eds., vol. 5792 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Sapporo, Japan, 26-28 October 2009, pp. 169–178.

[98] Yannibelli, V., and Amandi, A. Project scheduling: A multi-objective evolutionary algorithm that optimizes the effectiveness of human resources and the project makespan. *Engineering Optimization 45*, 1 (2013), 45–65.

[99] Ye, G., Rao, R., and Li, M. A multiobjective resources scheduling approach based on genetic algorithms in grid environment. In *Grid and Cooperative Computing Workshops, 2006. GCCW '06. Fifth International Conference on* (Hunan, October 2006), IEEE Computer Society, pp. 504–509.

[100] Yu, J., Kirley, M., and Buyya, R. Multi-objective planning for workflow execution on grids. In *Proceedings of the 8th IEEE/ACM International Conference on Grid Computing* (Washington, DC, USA, 2007), GRID, IEEE Computer Society, pp. 10–17.

[101] Zeng, B., Wei, J., Wang, W., and Wang, P. Cooperative grid jobs scheduling with multi-objective genetic algorithm. In *Parallel and Distributed Processing and Applications*, 29-31 august ed., vol. 4742 of *Lecture Notes in Computer Science*. Springer Berlin Heidelber, Niagara Falls, Canada, 2007, pp. 545–555.

[102] Zhao, H., and Sakellariou, R. An experimental investigation into the rank function of the heterogeneous earliest finish time scheduling algorithm. In *Euro-Par 2003 Parallel Processing*, H. Kosch, L. Böszörményi, and H. Hellwagner, Eds., vol. 2790 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Klagenfurt, Austria, 26-29 August 2003, pp. 189–194.

[103] Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P. N., and Zhang, Q. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation 1*, 1 (2011), 32 – 49.

[104] Zitzler, E., Deb, K., and Thiele, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation 8*, 2 (2000), 173–195.

[105] Zitzler, E., Laumanns, M., and Thiele, L. Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems* (Athens, Greece, 19-21 September 2001), International Center for Numerical Methods in Engineering, pp. 95–100.

[106] Zitzler, E., and Thiele, L. Multiobjective optimization using evolutionary algorithms - a comparative case study. In *Parallel Problem Solving from Nature* (Amsterdam, The Netherlands, 27-30 September 1998), A. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds., vol. 1498 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 292–301.

[107] Zitzler, E., and Thiele, L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation 3*, 4 (1999), 257–271.