Adapting Scientific Workflow Structures Using Multi-Objective Optimisation Strategies

IRFAN HABIB, Centre for Complex Cooperative Systems, FET, University of the West of England, Bristol, UK
ASHIQ ANJUM, School of Computing and Mathematics, University of Derby, Derby, UK
RICHARD MCCLATCHEY, Centre for Complex Cooperative Systems, FET, University of the West of England, Bristol, UK
OMER RANA, School of Computer Science & Informatics, Cardiff University, UK

Scientific workflows have become the primary mechanism for conducting analyses on distributed computing infrastructures such as grids and clouds. In recent years, the focus of optimisation within scientific workflows has primarily been on computational tasks and workflow makespan. However, as workflow-based analysis becomes ever more data intensive, data optimisation is becoming a prime concern. Moreover, scientific workflows can scale along several dimensions: (i) number of computational tasks, (ii) heterogeneity of computational resources, and the (iii) size and type (static vs. streamed) of data involved. Adapting workflow structure in response to these scalability challenges remains an important research objective. Understanding how a workflow graph can be restructured in an automated manner (through task merge for instance), to address constraints of a particular execution environment is explored in this work, using a multi-objective evolutionary approach. Our approach attempts to adapt the workflow structure to achieve both compute and data optimisation. The question of when to terminate the evolutionary search in order to conserve computations is tackled with a novel termination criterion. The results presented in this paper, demonstrate the feasibility of the termination criterion and demonstrate that significant optimisation can be achieved with a multi-objective approach.

Categories and Subject Descriptors: C.2.4 [**Distributed Systems**]: Distributed applications

General Terms: Multi-objective Optimisation, Evolutionary Computing, Scientific Workflows

Additional Key Words and Phrases: Multi-objective Optimisation, Evolutionary Computing, Termination criteria, Hypervolume, Workflow planning.

1. INTRODUCTION

Many real world problems are multi-objective in nature. Therefore, they consist of the optimisation of multiple criteria which may often be mutually conflicting. In these problems a global optimum, where all objective functions can be optimised in unison, often cannot be found because a tradeoff between various conflicting objectives must be sought. The solutions that have an optimal tradeoff between the various objectives (in the absence of any preference for any specific objective) are regarded as the solutions to these problems. They are often termed as the pareto-optimal or non-dominated solutions. These solutions are such that any improvement in one objective will necessarily lead to a deterioration in the other. Identifying these solutions in an automated manner, taking account of the particular structure of the problem domain, is a key requirement in many real world applications.

Workflows have emerged as the primary mechanism for describing and enacting complex e-Science analyses on distributed infrastructures such as Grids [Deelman et al. 2009] and also require optimisation across multiple, often conflicting, objectives. Examples of such scientific workflows include the analysis of data such as MRI scans, high energy physics detector data or data from Earth Observation sensors, achieved through a series of computations [Barga and Gannon 2007] that are exposed within the workflow graph. Scientific workflows continue to increase in complexity – both in terms of the data volumes they must process and the various types of computational capability they must combine. This complexity is therefore multi-dimensional [Gil et al. 2007]; scientific workflows need to scale: (i) in the number of computations they per-

form; (ii) in the size of data they manage and generate and the resources they consume. Optimising workflow enactment along these computation and data dimensions is often critical for successful analysis and usability. Various researchers have pointed out that advances beyond the current state-of-the-art are required to address these scalability challenges. New techniques are required to represent these workflows, to optimise them, to manage their enactment and to detect failures [Gil et al. 2007].

Many existing optimisation strategies target compute optimisation (or workflow makespan), although there has also been interest in data optimisation recently to support the increasingly more data-intensive scientific workflows [Deelman 2007]. A prime limitation of state of the art approaches for workflow optimisation is that they either are not automated, requiring manual user involvement for facilitating optimisation or are not adaptive, therefore performing best effort optimisations targeting either compute or data optimisation. Both compute and data optimisation of scientific workflows is considered in this work, focusing on the often conflicting requirements to optimise both objectives. e-Science workflows share common characteristics in that they are both complex in terms of the computations they handle and the amount of data they compute, however the structural characteristics of the workflow graph can impact the types of optimisations that can subsequently be carried out [Ramakrishnan and Plale 2010]. An automated approach enables the optimisation strategy to directly modify the workflow structure in order to yield data and compute optimisation (although the degree to which this can be achieved may differ across different types of scientific workflows).

To achieve compute optimisation, computations within a workflow must be distributed in order to benefit from parallelism. On the other hand, to achieve data efficiency computations must be localised in order to limit expensive network transfers. The use of a multi-objective meta-heuristic to optimise scientific workflows is described in this paper and evaluated through a number of real world scientific workflows – focusing on the CIVET [Ad-Dab'bagh et al. 2006] workflow in particular. We describe how an automated approach could be used to modify the structure of a workflow graph taking account of both computational and data transfer requirements, using a multi-objective optimisation strategy.

The domain of multi-objective meta-heuristics has been an active area of research [Talbi 2009] and various successful applications have been reported. For instance, several multi-objective evolutionary approaches have been used to optimise distributed computing capabilities such as scheduling [Huang and Zhu 2009]. However their use in the optimisation of scientific workflows has so far not been explored. Since the compute and data performance may be dependant on various factors, the search space of all possible optimised workflow plans may be large. An evolutionary meta-heuristic, being a stochastic population-based search algorithm, enables the simultaneous exploration of a search space as members of a population can be randomly distributed across the search space. Moreover, the genetic operations of mutation and crossover can enable the fine-grained control of the balance between exploitation (the ability to leverage characteristics of known solutions) and exploration (the ability to explore new parts of the search space). Multi-objective evolutionary algorithms (MOEAs) regarded as state-of-the-art include the Non-dominated Sorting Genetic Algorithms II (NSGA-II) [Deb et al. 2000], Strength Pareto Evolutionary Algorithm 2 (SPEA2) [Zitzler et al. 2001], Indicator based Evolutionary Algorithm (IBEA) [Zitzler and Künzli 2004] and HyPE [Bader 2010].

An open research question in evolutionary computation is the determination of the point during an evolutionary search at which the candidate solution set starts to converge. Continuing an evolutionary search beyond this point yields ever less significant optimisation. Therefore, it is of interest to terminate the search once optimisation

starts to level off. Many existing approaches, such as [Guerrero et al. 2009; Trautmann et al. 2008; Rudenko and Schoenauer 2004], use an array of indicators to determine *search stagnation*. We have explored the use of an online adaptive termination criterion that is based on a single comprehensive indicator. The approach provides adaptive parameters which a decision maker can tune to adjust the behaviour of the approach. The parameters may be configured to get either more optimal solutions at the cost of increased evaluations or more efficient solutions in a timely manner, with varying levels of quality.

To evaluate the effectiveness of the approach, results for a neuroimaging workflow, CIVET [Ad-Dab'bagh et al. 2006] are presented – part of the neuGRID project [Redolfi et al. 2009]. The neuGRID project is an effort to develop a general-purpose Grid-based computing infrastructure to support research into neuro-degenerative diseases, e.g. Alzheimers' disease. The CIVET workflow is a general-purpose neuroimaging pipeline which generates up to 4000% more data than it consumes and in a workstation with a modern dual core CPU it can take around 9 hours to process one brain. With the standard CIVET pipeline the analysis of the ADNI dataset [Jack Jr. et al. 2008], consisting of 8,000 images, may amount to 86 months of computation and generate data in multiple terabytes. The neuGRID project deployed the CIVET pipeline, among other neuroimaging workflows, in order to identify particular biomarkers, such as the cortical thickness, to detect the onset of neuro-degenerative diseases. The results presented in this paper indicate that significant compute and data optimisation was achieved for CIVET after the application of the approach detailed in this paper.

This paper is structured as follows: in Section 2 the state-of-the-art in workflow optimisation approaches is described. This section also provides a survey of related work covering various termination criteria proposed for multi-objective evolutionary algorithms. The multi-objective nature of e-Science workflows is highlighted in Section 3. The meta-heuristic being proposed and how the approach has been implemented in SPEA2 is described in Section 4. The SPEA2 meta-heuristic was chosen because it has been shown to have a good performance on several benchmark problems [Zitzler et al. 2000] – used within evolutionary search algorithms. Moreover, the SPEA2 meta-heuristic is an *elitist* multi-objective evolutionary algorithm, since it maintains an external archive of non-dominated solutions during the evolutionary search. An elitist approach is computationally efficient because only the changes in the archive of non-dominated solutions must be tracked to determine convergence, thereby avoiding the need to consider an entire population. In contrast, other multi-objective evolutionary algorithms like the IBEA are non-elitist, therefore may require more evaluations in order to determine convergence. The environment used for experiments is described in Section 5 and results and discussion are presented in Section 6. Concluding remarks are provided in Section 7.

2. RELATED WORK

Current approaches used for workflow optimisation are described in this section. Generally, workflow optimisation can be carried out in several stages of the scientific workflow life-cycle [Deelman et al. 2009]. These stages include: the composition phase (where the capability required within the workflow is identified), the instantiation phase (where the services needed to enact the workflow are identified), a mapping stage (where workflow services are mapped to physical infrastructure resources) and an execution stage (where the workflow plan is enacted by a local resource management system). The focus of the workflow optimisation in this study is on the mapping stage of the workflow life-cycle. The greatest potential for workflow optimisation is at the mapping stage. Prior to the mapping stage, all optimisation is related to the selection of the executable codes and manual optimisation of the graph structure. Af-

ter the mapping stage, all optimisation approaches are execution-level approaches. These approaches are specific to the type of distributed infrastructure being used. Distributed infrastructures are currently undergoing a transformation as grid computing and cloud computing infrastructure begin to merge . Approaches developed at the mapping stage, can be made relevant to future distributed infrastructures as well because they are not tied to a specific underlying infrastructure. Execution-level approaches on the other hand may not be relevant for the optimisation of future workflows. Therefore, the focus of the workflow optimisation approach is on the optimisation at the mapping stage of the workflow life-cycle.

In the mapping stage the workflow instance is mapped on to the underlying computing infrastructure which will execute the workflow [Deelman et al. 2003]. The mapping of the workflow from the abstract specification to the concrete specification is also termed workflow planning and several approaches have been proposed [Krauter et al. 2002], in systems such as Pegasus [Deelman et al. 2005], Karajan [von Laszewski and Hategan 2005] and Askalon [Prodan 2007]. It is useful to note that the mapping phase does not involve the subsequent execution or scheduling of workflow tasks on the resources – as this is undertaken by a local resource manager. The mapping phase therefore primarily involves finding suitable candidate resources on which tasks may be run and subsequently delegates the execution of such tasks to another system.

Significant amount of work has been done on mapping-level compute optimisation approaches and various data optimisation strategies at the mapping phase have been proposed. However, approaches that focus on both have not been proposed. Attempting to optimise both is complex as it involves conflicting tradeoffs, as will be explored in this paper. Task reduction has been one of the most widely used approaches to improving computation efficiency and is achieved by reducing the granularity of the workflow in order to save latencies associated with each job submission.

Generally, these can be divided into two further categories which include user driven approaches and automated approaches. The other widely used approach centres on predicting possible run times of tasks on particular resources based on historical data and using this data to map tasks to resources – we refer to these as Automated Scheduling based approaches.

Automated task clustering on the other hand is carried out by the workflow management system, an example being the Pegasus [Deelman et al. 2005] system. In automated task clustering, a workflow management system groups mutually independent tasks in a workflow based on pre-defined user specified parameters. These parameters include size of the cluster or the number of groups/clusters to consider, etc. A limitation of this approach is that the workflow management system generates the workflow plan without any consideration of the types of tasks being clustered. The workflow planner may therefore cluster coarse-grained tasks and data intensive tasks while ignoring fine-grained tasks. Consequently, this approach requires informed parameter selection by a user.

Most of these limitations are addressed by User-driven task reduction approaches[von Laszewski and Hategan 2005; Deelman et al. 2005]. In this approach the clustering strategy is specified by a user. The limitations of this approach, however, include the loss of parallelism in the workflow, as a topological ordering on the sequential execution of tasks within a user-specified cluster is assumed.

An extension of the automated task clustering approach is the dynamic task clustering, as implemented in the Falkon project [Raicu et al. 2007]. The Falkon project uses the SwiftScript [Zhao et al. 2007] language to specify the scientific analysis to be carried out (i.e. the workflow to be enacted). Unlike other projects, where the workflow structure is known *a-priori*, in Falkon the workflow is generated dynamically. As the SwiftScript specified workflow is executed, the workflow structure is dynamically gen-

erated and execution commences as soon as the first root node of the workflow graph is available. When multiple independent tasks are created these are clustered and submitted as a single task to the distributed computing infrastructure. The primary limitation of this approach, because a SwiftScript program is dynamically evaluated, is its lack of global knowledge about the workflow graph structure and the tasks being clustered – thereby leading to a potentially inefficient workflow plan.

Existing scientific workflow data optimisation approaches are limited to the execution-level stage of the workflow life-cycle and are deeply integrated with the underlying infrastructure. For instance, the data diffusion approach [Raicu et al. 2008] enables the enactment of data intensive workflows in a data efficient manner. However this approach requires deep integration with the underlying computing infrastructure which includes: the use of worker node-level caches, a custom resource broker and a custom scheduler which supports the data aware scheduling provided by the approach. Similarly, researchers have developed dedicated data placement services [Chervenak and Schuler 2007; Kosar and Livny 2004]. These services dynamically replicate data and move data based on the requirements of the workflow tasks. These data placement services minimise the time spent by workflow tasks waiting for input data.

Park and Humphrey [Park and Humphrey] identify that although obstacles to workflow efficiency are often found within the application, such as inherently limited parallelism because of the workflow definition, often reduced performance also arises due to the workflow engine that maps the abstract workflow to underlying resources in an inefficient manner. This is particularly relevant for applications that involve large data transfers between workflow tasks, where data location and link bandwidth is used to determine how to move large files (utilising the highest capacity links). They make use of a token bucket-based data throttling framework. In their approach, the model of computation of a workflow is data parallelism, consisting of different parallel branches transforming datasets. These datasets need to be transferred between workflow nodes with the key restriction being the limited control available over arrival time and rate of data transfer between nodes.

Our approach, unlike Pegasus (compared to both automated and user-driven clustering modes) does not require user involvement in the planning of the workflow. Similarly, it attempts to optimise a workflow plan for both compute and data optimisation unlike the dynamic clustering approach implemented in Falkon which performs opportunistic dynamic clustering. Our approach is a pure mapping-level approach therefore it is infrastructure agnostic unlike the data optimisations implemented by Pegasus and Falkon [Raicu et al. 2008]. However, as our approach carries out a multi-objective search for optimised workflow plans, it is not suitable for workflows that can change dynamically or are often revised. Such changes/revisions will impact the structure of the workflow and the performance characteristics of the previously optimised plans may change. Therefore, we recommend the use of this approach for standardised e-Science workflow that are not subject to change.

2.1. Current Approaches for terminating Multi-objective Evolutionary Meta-heuristics

An online termination criterion, such as the proposed in this paper, attempts to determine when to terminate a search during an evolutionary search. An offline approach on the other hand determines when it was appropriate to terminate a search after an evolutionary search has been conducted. Different types of online approaches-based have been developed, these include approach based on differential evolution [Ghosh et al. 2012; Xue et al. 2003]. Differential evolutionary approaches have popularly been applied to continuous problems. Therefore, the focus of the online termination approach in this paper is the detection of convergence based on multi-objective indicators which can be applied to discrete search spaces. One of the first *online* mechanism for an multi-

objective evolutionary termination criterion was proposed by Rudenko and Schoenauer [Rudenko and Schoenauer 2004]. They suggested a technique for determining search stagnation based on the stability of the *maximum crowding distance* [Deb et al. 2000], also termed *spread* of solutions. They demonstrated the applicability of this termination criterion to the NSGA-II meta-heuristic which uses the crowding distance in its selection criterion. It is an open research question whether the stability of the maximum crowding distance can be observed in other multi-objective meta-heuristics which do not use the crowding distance as a selection mechanism [Deb et al. 2000]. The use of multi-objective indicators to detect stagnation is popular. Marti et al. introduced a multi-objective gradient-based approach [Martí et al. 2007] which deployed a Kalman filter with the mutual dominance ratio indicator. This approach was extended by [Guerrero et al. 2009] where a Kalman filter was applied to fuse information from both the Hypervolume indicator [Purshouse 2003] and the Additive ϵ indicator [Zitzler et al. 2003] to determine convergence and diversity of solutions. In this approach a final global stopping decision was made based on the behaviour of a mutual dominance ratio and the fused Hypervolume and Additive ϵ indicators.

Recently, Wagner and Trautmann [Trautmann et al. 2008] proposed a Hypervolume-based online convergence mechanism designed to be used for the S-Metric Selection Evolutionary Multiobjective Algorithm (SMS-EMOA) [Beume et al. 2007]. The SMS-EMOA meta-heuristic is a multi-objective meta-heuristic which uses the Hypervolume indicator in its selection approach during the evolutionary search. Our termination criterion is also based on the Hypervolume indicator, however it is not designed for a specific meta-heuristic; to demonstrate usage, its application to the SPEA2 meta-heuristic is provided. Moreover, it can be tuned by a decision maker to achieve a tradeoff between more optimal solutions at the cost of increased evaluations (i.e. greater computational time) or more efficient solutions with lower solution quality. The motivation for selecting this indicator is discussed in Section 4.

3. MULTI-OBJECTIVE SCIENTIFIC WORKFLOW OPTIMISATION

Achieving both compute and data optimisation is a challenging undertaking in many existing scientific workflows. For instance, a lower turn-around time is critical for time-sensitive e-Science workflows, while data efficiency is crucial when the e-Science workflow must deal with large amounts of data, especially when such data needs to be moved across public networks with limited bandwidth and when buffer/storage capacity is limited at the destination. The reason both cannot be achieved in a single plan is due to the fact that different, often conflicting considerations are required to formulate data-efficient and compute-efficient workflow plans.

Formally, a directed acyclic graph (DAG) based workflow can be represented by $G = (T, E)$, where $T = \{t_1, t_2, ..., t_N\}$ is a set of tasks, up to N. The set E is the collection of edges indicating the dependency and precedence constraints between tasks. During execution of a workflow a specific amount of data is transferred between dependent tasks, represented as a NxN matrix, D. Hence D_{ij} denotes the amount of data required to be transferred from task t_i to task t_j. If two tasks have no dependency then $D_{ij} = 0$. Grid resources can be defined as a set of hosts, $H = \{h_1, h_2, ..., h_M\}$ where M is the total number of hosts.

Additionally the interconnection between hosts is represented by two MxM matrices B and L^n where B_{ij} is the bandwidth between two hosts h_i and h_j, while L^n_{ij} is the network latency from host h_i to host h_j. Since there is no need for transferring data within the same host we assume that B_{ii} and $L^n_{ii} = 0$. If task t_i is the direct precedent of task t_j, and if t_i is being executed on host h_m, and t_j on host h_n, then the time required for transferring the output data of task t_i from h_m to h_n is called the transmission

time, denoted as $DT(t_i(m), t_j(n))$, shown in Equation 1:

$$\mathrm{DT}\left(t_i(m), t_j(n)\right) \leftarrow L^{\mathrm{n}}_{\mathrm{mn}} + \frac{D_{\mathrm{ij}}}{B_{\mathrm{mn}}} \tag{1}$$

From Equation 1, if $m = n$, $DT(t_i(m), t_j(m)) = 0$. The average transmission time between t_i and t_j is shown in Equation 2.

$$\overline{\mathrm{DT}}\left(t_i, t_j\right) \leftarrow \bar{L^{\mathrm{n}}} + \frac{D_{\mathrm{ij}}}{\bar{B}} \tag{2}$$

Here, $\bar{L^{\mathrm{n}}}$ is the average network latency and $\bar{B}$ is the average bandwidth among hosts. In order to determine the complete time spent on transferring data between tasks during the lifetime of a workflow we consider the time taken to conduct all data transfers between tasks along the critical path of the workflow. The critical path of the workflow is considered as the smallest number of activities required to complete a workflow plan. However, these activities may also have dependencies on tasks that are not in the critical path, therefore the execution of these tasks may be delayed by a queue waiting time (also referred to as synchronisation time between tasks), denoted as L^s_i for the task t_i. Based on Equation 2, the total time spent transferring data from tasks is defined by:

$$\begin{gathered}\mathrm{DT}(G) \leftarrow \sum_{i,j}^{n} \overline{\mathrm{DT}}\left(t_i, t_j\right) + L^s_j \\ \text{where } \forall t_i, t_j \in \mathrm{CriticalPath}(G)\end{gathered} \tag{3}$$

$ET(t_i, h_n)$ is the estimated execution time of task t_i on host h_n. Based on this, the average estimated execution time of t_i can be calculated as follows:

$$\overline{\mathrm{ET}}\left(t_i\right) \leftarrow \frac{\sum_{n=1}^{M} \overline{\mathrm{ET}}\left(t_i, h_n\right)}{M} \tag{4}$$

As specified earlier M is the total number hosts. Based on Equations 3 and 4 the total workflow turn-around time is defined as shown in Equation 5. The term $n\bar{L}$ represents all the latencies incurred by each task from submission to execution and excludes the queue waiting time latency L^s.

$$\begin{gathered}\mathrm{WFT}(G) \leftarrow \sum_{x=1}^{n} \overline{\mathrm{ET}}\left(t_x\right) + \mathrm{DT}(G) + n\bar{L} \\ \text{where } \forall t_x \in \mathrm{CriticalPath}(G)\end{gathered} \tag{5}$$

Equation 5 identifies the factors that affect the workflow turn-around time. The first term represents the ideal workflow turn-around time, which is the beginning of the execution of the first root node of the workflow and the end of the execution of the leaf node in the workflow. However the ideal workflow turn-around time cannot be achieved because every task submission to a resource carries a cost in the form of a multitude of latencies. There are a number latencies associated with every such submission [Nerieri et al. 2006]. Formally,

$$ET_{real}(t_i) \leftarrow \overline{\mathrm{ET}}\left(t_i\right) + \bar{L} \tag{6}$$

Latencies most severely impact workflow tasks that are fine-grained or where $\bar{L} \geq \overline{\mathrm{ET}}\left(t_i\right)$. In this case the $ET_{real}(t_i)$ may be more than twice that of $\overline{\mathrm{ET}}\left(t_i\right)$. For tasks where $\overline{\mathrm{ET}}\left(t_i\right)$. For instance, 54.7% of the tasks in the CIVET workflow are fine-grained in nature and have a runtime that is less than 10 times than the mean latency observed in a production environment[Lingrand et al. 2008]. This implies that the turn-around time for a single task may be 10 times larger than its runtime. Hence if fine-grained computations are clustered together or eliminated in a workflow a significant improvement in the workflow turn-around time may be observed.

3.1. Optimising for Data efficiency

Improving data efficiency of a workflow involves reducing or eliminating latencies associated with data transfers. In the DAG workflow model, an edge represents a specific data transfer. We have used the term $\overline{\mathrm{DT}}\,(t_i, t_j)$ to define a data transmission event in the workflow between tasks t_i and t_j. Expression 7 defines the upper and lower bounds for the data efficiency of a workflow DAG, where the worst data efficiency would be achieved if data has to be moved for every edge. This situation would occur if a workflow is scheduled in such a manner that each task in the workflow is executed on a different Grid node. In this case before the execution of a task can occur all dependent data has to be staged to the appropriate Grid node.

$$\sum_{i=1,j=1}^{n} \mathrm{DT}\,(t_i(h_m), t_i(h_m)) \leq DataEfficiency \leq \sum_{i=1,j=1}^{n} \overline{\mathrm{DT}}\,(t_i, t_j) \tag{7}$$

On the other hand, the best case would be achieved if all data transfers are local, and are performed at a single node. We assume that the latency of copying a data set from one node to itself is 0. This situation would occur if a scheduler schedules all nodes of a task to a single Grid node. In this scenario no data latency would be incurred. However this is clearly an impractical solution since the compute efficiency will be severely impacted. All independent jobs in the workflow that could be executed in parallel will be executed on a single host. Hence a trade-off must be considered between compute and data efficiency. The objectives are mutually conflicting. Data efficiency is achieved by localising computations while compute efficiency is achieved by distributing computations in order to leverage parallelism.

3.2. Towards a multi-objective approach

The CIVET workflow has an average runtime of 9.6 hours and a data footprint of 403.75 MB per brain scan. The data footprint is generally defined as the amount of data consumed and produced during the lifetime of a workflows' execution. The CIVET workflow may appear to have a low data footprint, however it is data intensive because it generates 4000% more data than it consumes per brain scan processed. In order to achieve compute and data efficiency a multi-objective approach must be considered since they involve mutually conflicting objectives. Therefore, the formal specification of the multi-objective problem is as follows. Let x represent a decision vector of a workflow and let the set X be the decision space. The objective vector for a candidate solution is represented as y, and the objective space is the set Y. There are two objectives in this optimisation problem: turn-around time, denoting the compute efficiency of the workflow and data efficiency which is denoted by the amount of data transfers among workflow tasks during the execution of the workflow. Both objectives are represented as the functions $f_{compute}$ and f_{data}, respectively. Let $e_{compute}$ and e_{data} represent the constraints for both compute and data efficiency. The set of undesirable solutions for this optimisation problem consists of all candidate solutions which have a worse compute and data efficiency than an unoptimised executable workflow instance. Formally, the problem is defined as:

$$\begin{aligned} \textit{minimise} \quad & y = \quad f(x) = (f_{compute}(x), f_{data}(x)) \\ \textit{subject to} \quad & e(x) = \; (e_{compute}(x), e_{data}(x)) \\ \textit{where} \quad & x \epsilon X \\ & y \epsilon Y \end{aligned} \tag{8}$$

The mapping of a workflow to the decision vector defines the operations that can be conducted to optimise a workflow. Section 4 will detail the decision vector mapping

used in this work. The process of optimising a scientific workflow using an evolutionary meta-heuristic involves the evaluation of hundreds or thousands of candidate workflow plans – which may be prohibitively expensive (computationally) to undertake. Therefore, an approach is proposed which attempts to detect convergence based on the Hypervolume indicator, discussed in Section 2, and terminates the search. The termination criteria used in this work and its application to the SPEA2 meta-heuristic is presented in Section 4.3.1.

4. META-HEURISTIC FOR OPTIMISING SCIENTIFIC WORKFLOWS

This section describes the approach used for the optimisation of scientific workflows using the multi-objective meta-heuristic SPEA2. The candidate solution representation is introduced and a novel termination criterion is described, along with evaluations of the termination criteria with results of this analysis applied to the CIVET workflow.

4.1. Candidate Solution Representation

In order to conduct an evolutionary search, a suitable candidate solution must be formulated for the application of genetic operations such as crossover, mutation and selection. The candidate solution representation adopted in this work is based on the task clustering approach. Section 2 introduced the task clustering approach as it is implemented in state-of-the-art workflow optimisation projects. Moreover, as has been previously established, different types of tasks need to be clustered to achieve the objectives adequately. For instance, fine-grained tasks need to be clustered to achieve compute optimisation, while tasks that require a significant amount of data from a single predecessor task can be clustered together to achieve data efficiency. Consequently, by selecting a representation based on the task clustering approach, the evolutionary search in this study is the search for pareto-optimal clustering strategies for workflows. This representation is formulated using a vector-based problem encoding, in evolutionary computation this is called a chromosome. An element of a chromosome is termed an allele. The allele in our adopted representation is a vector of two elements.

Two possible automated task clustering strategies have been identified in literature [Singh et al. 2008]. These include collapsing a set of independent tasks into a single cluster or bundling independent tasks into a specific number of clusters. Hence, the decision vector used in the meta-heuristic is a nested vector of size n where n represents the number of levels in a graph. Each level in the DAG represents a set of independent tasks. Formally, if the set D represents the decision vector, each element in the decision vector is defined by Equation 9.

$$D_i \longleftarrow \left[s \in \{'c', 'b'\}, f \in \{1 \in \mathbb{Z} \wedge 1 \leq f \leq |L_i|\}\right] \tag{9}$$

The symbols c and b represent the clustering strategy to apply at that specific level, i, of the workflow. The granularity of the clustering strategy is defined by factor f. The values associated with this factor range from 1 to the number of tasks at level i, denoted by $|L_i|$. For instance, if the number of tasks at a level is 5, a decision value of $c5$ translates to a single five node cluster (5 tasks are collapsed into a single cluster), while a value of $b5$ translates to five one node clusters (5 tasks are bundled into 5 clusters). Figure 1 depicts the use of a decision vector on a workflow graph visually.

4.2. Population Sizing

Determining the size of the population is a critical parameter within an evolutionary algorithm. Ideally, the search should explore all possible solutions in order to determine all pareto-optimal solutions, but this cannot be achieved in a computationally tractable time for most problems. However selecting a population size that is too small does not leave enough room for exploration and hence the search can prematurely

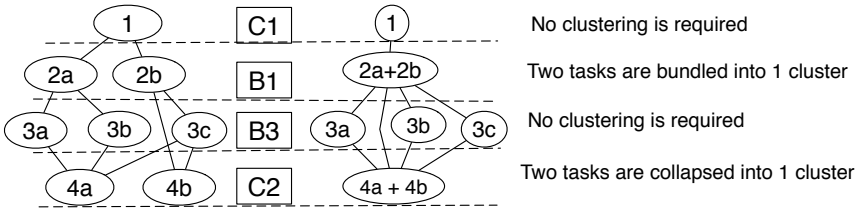


Fig. 1: Visual Representation of the Decision Vector

stagnate. Several researchers have explored this problem of optimal population sizing [Grefenstette 1986; Schaffer et al. 1989; Goldberg et al. 1991]. The approach that has been adopted in this paper is the principle proposed in [Reeves 1993]. The principle states that the population size should enable access to the entire search space by means of crossover only. This requirement can only be satisfied if every single decision vector value is represented in the population.

The complete size of the search space is represented by Expression 10. The set L represents number of tasks per level of a workflow, with the number of tasks at level n being L_n. The term $S_k^{L_n}$ represents the Stirling number of the second kind [Abramowitz and Stegun 1972] and is used to calculate the number of ways n elements can be partitioned into k non-empty sets. The total number of clustering plans that can be applied to a single level in a workflow is the sum of all partitions that can be generated of L_n tasks with k set blocks. The product of the total number of clustering plans per level yields the total size of the search space for a workflow.

$$N \leftarrow \prod_{n=1}^{|L|} \left(\sum_{k=1}^{L_n} S_k^{L_n} \right) \tag{10}$$

In order to determine the minimum population size required to have 99.9% confidence that all decision vector values are represented in the population, Expression 11 was derived and used. This expression is based on the work presented in [Reeves 1993]. The vector L represents the number of clustering strategies that can be applied at a single level of the workflow DAG. The length of L is equal to the number of levels in a DAG. Every value in the decision vector corresponds to a single level in the DAG and n represents the required population size. It is of note here, that the approach outlined here is only applicable workflow DAGs that are static in nature, as the total number of clustering plans that can be generated for such a workflow is always constant. On the other hand, workflows that are generated dynamically, as in the case of a SwiftScript workflow [Raicu et al. 2007], the required population size has to be evaluated again after every dynamic instantiation of the workflow as the structure of the workflow may have changed. The workflows considered in this study are static DAG workflows.

$$p_q = \prod_{i \in L}^{|L|} \left\{ \frac{i! S_i^n}{i^n} \right\} \tag{11}$$

According to the approach, the size of the search space for the CIVET workflow is 5.16672×10^{11}, which represents the total number of clustering strategies that can be applied to the workflow. On the other hand the minimum population size required to have a 99.9% confidence that all decision vector values are represented in the population is 68.

4.3. Meta-heuristic Algorithm

The meta-heuristic is shown in Algorithm 1 and follows the pattern of a standard evolutionary algorithm. Let P_t represent the population of candidate solutions at generation t. The terms P_t^D, P_t^O and P_t^F refer to the decision vectors, objective vectors and fitness values for the population at generation t, respectively. In the beginning of the algorithm the population of candidate workflow clustering solutions, denoted as P_0^D, is initialised, after which the meta-heuristic iterates within a loop until the termination conditions become valid. Initially the objective vector for each candidate solution is evaluated within the loop, followed by the calculation of the fitness of each candidate solution.

The fitness calculation procedure is followed by the update of the external archive which contains $\overline{N}$ number of non-dominated solutions which have been discovered by the meta-heuristic up to this point. The termination criterion is evaluated from the third generation of the evolutionary search, because a change in the Hypervolume indicator is calculated over the last three generations. The termination criterion is checked as soon as an archive of non-dominated solutions for the generation t,$\overline{P}_t^F$, has been determined. The termination criterion is described in Algorithm 2. The final steps within the loop prepare the next generation of candidate solutions. At first, candidate solutions for the next generation are selected using the binary tournament approach, followed by the application crossover operator on a subset of the population determined by crossover probability ρ_c. Finally, a mutation operator is applied on a subset of the population, as determined by the probability of mutation ρ_m.

Algorithm 1 SPEA2 meta-heuristic Algorithm Main Loop

Input :
N, Size of the population
$\overline{N}$, Size of the external set
T, Max. number of generations
ρ_c, Crossover probability
ρ_m, Mutation probability
α, Change threshold
n, Generation count threshold
Returns :
$\overline{P}$, Set of non-dominated solutions of size $\overline{N}$

procedure SPEA2($N, \overline{N}, T, \rho_c, \rho_m$)
  $t \leftarrow 0$
  $P_0^D \leftarrow initPopulation()$
  while $(t \leq T) \wedge (termCondition \neq true)$ **do**
    $P_t^O \leftarrow evalObjectives(P_t^D)$
    $[P_t^F, \overline{P}_t^F] \leftarrow evalFitness(P_t, \overline{P}_t)$
    $\overline{P}_{t+1} \leftarrow UpdateArchive(P_t, \overline{P}_t, \overline{N})$
    if $t \geq 3$ **then**
      $termCondition \leftarrow termCheck(t, \overline{P}_t, n, \alpha)$
    end if
    $P_{t+1} \leftarrow Selection(P_{t+1})$ ▷ Binary Tournament
    $P_{t+1} \leftarrow Crossover(P_{t+1}, \rho_c)$
    $P_{t+1} \leftarrow Mutation(P_{t+1}, \rho_m)$
    $t \leftarrow t + 1$
  end while
  return $\overline{P}_t$ ▷ Final Archive Set
end procedure

4.3.1. Termination Approach. The termination criterion is based on the Hypervolume indicator, originally proposed and employed in [Zitzler et al. 2001] to quantitatively compare the outcomes of different multi-objective evolutionary algorithms. Any approximate pareto set with a higher Hypervolume indicator value is considered to be *better* than one with a lower Hypervolume value with respect to a reference point. Often the reference point is also termed as the nadir point, since it refers to the point in the objective space which represents the worst values for all objectives. The Hypervolume indicator can be used to capture both measurements of the convergence of a pareto front and the spread or diversity of the pareto front with respect to a reference front. In contrast, other indicators measure only convergence or diversity such as the generational distance [Veldhuizen and Lamont 1998], the inverse generational distance [Veldhuizen and Lamont 1998], the Additive ϵ indicator and the crowding distance indicator [Brockhoff and Zitzler 2007]. Therefore, often multiple indicators must be combined to provide an accurate picture of the pareto front, while a single Hypervolume measure may provide the same information. Due to this property the Hypervolume indicator has been used extensively to drive the evolutionary search. Algorithms such SMS-EMOA [Beume et al. 2007], IBEA [Zitzler and Künzli 2004] and HyPE [Bader 2010] use the Hypervolume indicator directly in the selection mechanism. Changes in the indicator are used in this work as the basis for the termination criterion.

For the termination procedure, the Hypervolume indicator value associated with the approximate pareto fronts of two generations is compared. The approximate pareto front at generation t is denoted as $\overline{P}_t$. It is assumed that $\overline{P}_t \succeq \overline{P}_{t-1}$. In order to calculate the Hypervolume of $\overline{P}_{t-1}$, the maximum values for all objectives in $\overline{P}_t$ are used as the nadir point. In order to determine a change, the Hypervolume of $\overline{P}_{t-2}$ is calculated against the same nadir points from $\overline{P}_t$. Therefore, formally:

$$\Delta_{HV}(\overline{P}_t, \overline{P}_{t-1}, \overline{P}_{t-2}) \leftarrow \frac{I_{HV}(\overline{P}_{t-1}, \overline{P}_t) - I_{HV}(\overline{P}_{t-2}, \overline{P}_t)}{I_{HV}(\overline{P}_{t-2}, \overline{P}_t)} \tag{12}$$

The procedure is described in Algorithm 2. A list of past Hypervolume changes is maintained, denoted as I_Δ. The element i in the list is denoted as I^i_Δ. During a search, if the size of I_Δ equals the user specified generation count threshold n then the following conditional is invoked (line 3). If the average Hypervolume change over the past n generations is below the user specified change threshold α then the termination condition, $termCondition$, is set and the meta-heuristic terminates in this generation. However, if the average change in I_Δ is not below α, then the oldest element, I^0_Δ is removed from the list I_Δ. Later, in line 5 the change in Hypervolume, $\Delta(\overline{P}_t, \overline{P}_{t-2})$, over the last two generations is added to the I_Δ.

5. EXPERIMENTAL SETUP

Table I: Hardware Specifications

	CPU	RAM	Storage	Network	OS
3x Server	2x Quad-Core AMD Opteron 2.4 Ghz	32 GB	500GB	1 Gb	Ubuntu 8.04
21x Virtual Machines (Deployed on the Servers)	1 Core of AMD Opteron 2.4 Ghz	1 GB	5GB	1Gb	Ubuntu 8.04

Algorithm 2 Termination Condition Handling Procedure

Input :
n, Generation count threshold
I_Δ, Ordered tuple of change in I_{HV} indicator values where I^i_Δ is the element i of I_Δ
α, Change threshold for I_Δ
$\overline{P}_t$ external set at generation t
Returns :
$termCondition$ Sets the termination condition

1: **procedure** TERMCHECK($t, \overline{P}_t, n, \alpha$)
2: **if** $|I_\Delta| = n$ **then**
3: $(\frac{\sum_{i=1}^{|I_\Delta|} I^i_\Delta}{|I_\Delta|} \leq \alpha) \rightarrow (\text{termCondition} \leftarrow true) \wedge \neg(\Delta I_\Delta \leq \alpha) \rightarrow (I_\Delta \leftarrow I_\Delta - I^0_\Delta)$
4: **end if**
5: $I_\Delta \leftarrow I_\Delta + \Delta_{HV}(\overline{P}_t, \overline{P}_{t-1}, \overline{P}_{t-2})$
6: **end procedure**

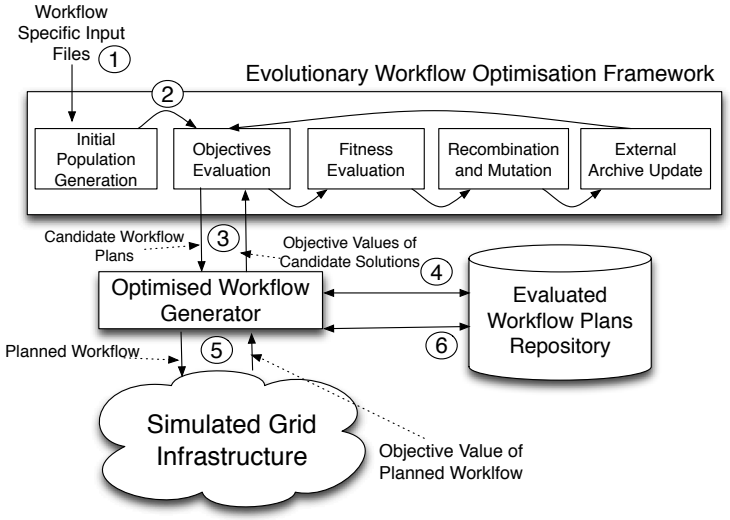


Fig. 2: Overview of the Architecture

The simulation framework used to implement the meta-heuristic and to optimise scientific workflows is shown in Figure 2. The framework consists of three components. The first component is the Evolutionary Workflow Optimisation Framework (EWOF), developed in the meta-heuristic framework jMetal[Durillo et al. 2010], that implements the approach. The second component is the Optimised Workflow Generator (OWG), which is a component that receives a candidate solution from the EWOF and generates the optimised workflow instance. The optimised workflow instance is then executed in the Simulated Grid Infrastructure (or SGI).

The flow of control in the framework is specified in Figure 2. In step (1) the EWOF uses the workflow graph structure (along with other data files) to initialise the workflow specific decision vectors (i.e. whether to use clustering or bundling, as described in Section 4) and to generate the initial population. In step (2) after the initial population has been generated the main loop of SPEA2 meta-heuristic commences. At first the objective values for each individual candidate solution are evaluated. In or-

der to perform this evaluation the OWG is invoked, as depicted in step (3). The OWG at first determines if the clustering strategy, denoted by the decision vector, has already been evaluated, as shown in step (4). If the clustering strategy has been evaluated then the objective vector obtained in the previous evaluation is returned to the EWOF. Otherwise, the OWG uses the task runtime, data set size and the workflow graph in addition to the decision vector to formulate an optimised workflow instance. The optimised workflow instance is executed on a simulated Grid infrastructure (using SimGrid [Casanova et al. 2008]), generated by the SGI component as shown by step (5) based on an infrastructure platform file required by SimGrid. The objective values obtained from the execution are returned to the EWOF via the OWG. The OWG stores the results in the Evaluated Workflow Plans Repository as shown by step (6). After all the objective values for each decision vector are obtained, the main loop of the meta-heuristic continues. The following are input files required by the EWOF.

(1) The *Workflow Graph* is the representation of the workflow DAG specified in the DOT language [Graphviz 2010]. It is used for setting the size of the decision vectors and generate the candidate solutions.
(2) The *Task Runtime Data* file contains execution time information of each individual task of the workflow. The task runtime information is used by the simulated Grid infrastructure to determine the time to run a workflow.
(3) The *Wf Data Set Sizes* file contains data volumes required or generated by the workflow. The data set sizes file is used by the simulated Grid infrastructure for the execution of the workflow.
(4) The *Infrastructure Platform File* represents the Grid infrastructure that is to be simulated. It contains the number of hosts in the Grid, the power of each host, which is specified in floating point operations per second (FLOPS). The file also defines the network topology, the bandwidth, and the latency of each link in the topology. In this work, the power of each specific host was equal to the FLOPS rating of the machine on which the runtimes of each individual task was retrieved.

The distributed experiment setup is summarised in Table I. A total of three servers were used to run 21 virtual machines. The virtualisation hypervisor used was Xen 3.2. The simulation framework developed is not sensitive to the load of a server. SimGrid uses a mathematical model to calculate the runtime of a workflow, therefore in jMetal no time-sensitive calculations are performed which could impact the accuracy of the results. In order to evaluate if the performance of the meta-heuristic is impacted by the balance of mutation and crossover, two sets of evaluations will be performed for the CIVET workflow. One evaluation will be performed where the balance of the crossover and mutation probabilities is towards the crossover operation. In this configuration the probabilities are ρ_m=0.5 and ρ_c=0.9. The evolutionary search in this instance is termed the *exploration focused* configuration. The other configuration will be mutation centric. This configuration will have the probabilities ρ_m=0.9 and ρ_c=0.5. Due to the higher mutations in this instance, this is termed the *exploitation focused* configuration. These two configuration will be compared to assess their impact on workflow efficiency. The t-test will be used to determine any significant trends between the two configurations. The SPEA2 population size for the CIVET Workflow was 68, as calculated by the approach outlined in Section 4.2 and the archive size is 20.

The compute complexity of the simulation framework depends on both the underlying computing power available and the population size of the workflow. The population size of the workflow determines the number of evaluations that will be carried out per generation of the evolutionary algorithm. For the CIVET workflow the average time to calculate a single generation is 3.1 minutes.

6. RESULTS

This section presents the results of the evaluations. Experimental analysis was carried out by applying various thresholds against the standard multi-objective problem set of the WFG family [Huband et al. 2006]. The WFG family cover a variety of different

types of search spaces – thereby enabling them to be used as a basis for comparing various evolutionary search algorithms (in terms of the Hypervolume indicator and an assessment of the associated pareto fronts). The search space can vary in complexity from a uni-modal landscape (for WFG2), multi-model and concave (for WFG4), to degenerate and multi-modal landscape with a parameter dependent bias and a concave geometry (for WFG9). Conclusions drawn from these experiments were used to specify the termination thresholds for the optimisation of scientific workflows. The authors recognise that that real-world problem being targeted is a combinatorial optimisation problem while WFG consists of problems that are numerical optimisation problems. Since diverse multi-objective problem sets do not exist that feature diverse combinatorial optimisation problems, therefore, in order to objectively measure the performance of the termination approach and make informed decisions about the value of the termination thresholds the authors have used the WFG toolkit to evaluate the termination criterion.

Combinations of the change threshold of 1%,0.5% and 0.25% were applied with the generation count thresholds of 10,20 and 50 to optimise the WFG family of multi-objective problems. The following observations were made.

— *A small generation threshold value for n can lead to premature termination of the evolutionary search.* The worst performance for several problems of the WFG family were registered with the generation threshold value of 10. During an evolutionary search, the progress of the search may plateau for some generations. At this plateau, changes to the fitness of the pareto front may be minimal. Therefore, the termination criteria may terminate the search when such a plateau is reached. Consequently a suitably large value for the generation threshold, n is suggested by the results.
— *A small change threshold, α, can lead to premature termination of the evolutionary search.* It was observed in several problems, most notably WFG4, WFG5, WFG6, WFG7, WFG8 and WFG9, no results were recorded for the configuration of α=0.25 and n=50, implying that convergence was never achieved. Analysis of the evolutionary search for these problems however suggested that convergence had been achieved and the search was not making any progress towards the ideal pareto front. Therefore, a higher change threshold value is suggested by the results.

The results suggest that a high value for the generation threshold is required. Therefore, for subsequent evaluations a generation threshold of 150 has been used. In terms of the change threshold, α, three configurations were used in these evaluations. The least successful configuration is α=0.25. The average termination generation of the configuration with α=0.5 and n=50 was 319.38 ± 365.26 (a high variance implies significant differences between the WFG problem sets), while the average termination generation with the configuration α=1.0 and n=50 was 60.55 ± 8.94. On the other hand the difference in terms of performance is 32%. Experimental evaluations with the configuration α=1.0 and n=150 indicated that it was competitive for the entire WFG problem set. In other words, the performance of the termination criterion with this configuration was not significantly different from the results achieved with the best configuration for each individual problem set. Statistical significance was determined via a Kruskal-Wallis analysis of variance test using a p-value of 0.05. Therefore, the termination thresholds of α=1.0 and n=150 have been applied to the optimisation of scientific workflows.

6.1. Optimisation of the CIVET Neuroimaging Workflow

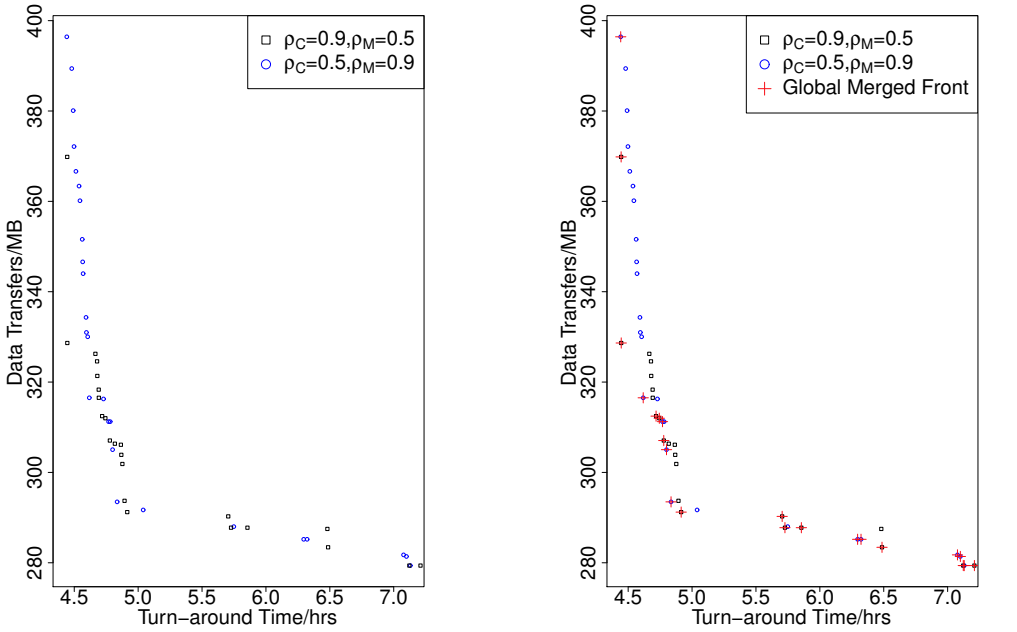
This section presents the results of optimising a real world workflow (CIVET [Habib et al. 2009]) using the meta-heuristic presented in this paper. This workflow is both compute and data intensive and generates 4000% more data than it consumes. The standard turn-around time of the CIVET workflow in the simulated Grid infrastructure used for this study is 10.54 hours and the data footprint is 405.80MB. This performance will be the benchmark against which the optimised workflows are evaluated.

The results are presented as follows: at first, the global approximate pareto optimal front obtained will be presented, and calculated by determining the non-dominated solutions discovered across all 20 runs of the meta-heuristic. Afterwards the performance achieved for the most compute-efficient, data-efficient and compute and data efficient workflow plans are presented.

6.1.1. Global Pareto Fronts. Figure 3a depicts the global pareto fronts achieved for the CIVET experiments. The square represents the case when the balance between exploration and exploitation (as explained in Section 5) was set towards exploration and the circle represents the converse case. We can observe that the geometric shape of the pareto front is largely convex in form. The solutions at the bottom right are the solutions that are most data efficient, as they have the lowest data transfers during the lifetime of the workflow. The solutions at the top left are solutions that are most compute efficient, as they have the lowest workflow turn-around time.We can observe in Figure 3a that the optimisation achieved is significant. The most compute efficient solution discovered has a workflow turn-around time of 4.44 hours, which is 57.87% more efficient than the unplanned CIVET instance. The most data efficient solution discovered had a data footprint of 279.38MB, which is 31.15% less than the standard unplanned workflow instance. Hence, the most data efficient workflow plan is least compute efficient, while the most compute efficient workflow plan is least data efficient.

Several workflow plans were also discovered that have a better balance between compute and data efficiency. These solutions are clustered around the lower left corner of the plot. A typical pareto-optimal solution lying in this cluster for instance includes the solution where the workflow turn-around time is 4.77 hours. This solution is 54% more efficient then the standard unplanned CIVET workflow, however at the same time it is 4.7% less efficient than the most compute efficient solution in the pareto-front. The compute efficiency difference of 4.7% may appear to be marginal, however it translates to a 2.8 minutes difference in the turn-around time of the workflow for a single brain scan. For the ADNI dataset which consists of 8000 images, this translates into an additional 6.2 days of processing. The same solution has a data footprint of 326.3 MB, which is 23.3% lower than the standard unplanned CIVET instance, but 10.2% higher than the data footprint of the most data efficient workflow plan. However this data footprint is 15.8% lower than the most compute efficient workflow plan. If a neuroimaging researcher is interested in measuring the cortical thickness of a brain scan in a timely yet data efficient manner, then a workflow plan that lies in the middle cluster is suitable. While for larger data analysis, the most compute efficient workflow plan may be selected. Otherwise, if network traffic is to be minimised at the cost of compute efficiency, then a data efficient workflow plan may be selected.

We can observe that the pareto front achieved with exploration-centered configuration has a greater spread. While the pareto front achieved for exploitation-centered configuration is largely clustered between the data transfer values of 300 MB and 320 MB and a workflow turn-around time of 4.8 hrs. However, as we can observe from Figure 3b the contribution of the non-dominated solutions from both configurations are largely equal. The quality of solutions discovered by the exploitation centric configuration is better than those from the exploration centric configuration. For instance, the most compute efficient solution discovered by the exploration centric configuration has a workflow turn-around time of 4.4 hrs and a data footprint of 398 MB, compared to a workflow turn-around time of 4.45 hrs and a data footprint of 372 MB. The difference is of only 3 minutes, or 0.04% of the workflow turn-around time of the standard workflow. While the difference in terms of the data footprint is 26MB, or 6.4% of the data footprint of the standard workflow. Therefore, the compute efficient solution dis-



(a) Merged Non-dominated Fronts achieved for Individual Configurations

(b) Superimposed Global Non-dominated Front

Fig. 3: Global Non-dominated Solutions obtained for the CIVET workflow across all experiments

covered by the exploitation centric configuration is better even though it has 3 minutes more workflow turn-around time than the most compute intensive solution discovered by the other configuration.

6.1.2. Performance of the Termination Criterion. This section will focus on analysing the performance of the termination criterion used in the meta-heuristic. The average lost performance, as measured by the change in Hypervolume against the global pareto front, at the termination generation and the end of the evolutionary search was 1.66% $\pm$ 1.35%. The average termination generation for both exploitation and exploration configurations combined was 273 $\pm$ 32.77. Statistical analysis of individual configurations indicated the differences between the configurations were insignificant.

In order to judge how much performance, in terms of the actual workflow turn-around time and data footprint of the candidate workflow plans, was compromised we will compare the average most compute efficient solutions at the termination generation with the average most compute efficient solutions at the end of the evolutionary search. The same analysis has been performed for the average most data efficient and for the compute and data efficient solutions for each configuration. The results are presented in Table II. We can generally observe that the differences between the two configurations (exploration-centric and exploitation-centric) at the termination generation are statistically insignificant.

We can observe that the greatest difference in the data footprint for the most data efficient workflow plans, when comparing their termination generation and the end of the evolutionary search, occurs for the exploitation-centric configuration which is 3.35MB. This constitutes a loss of just 0.8% when compared to the data footprint of the

unplanned CIVET instance. In terms of the most compute-efficient solutions, the greatest differences for the workflow turn-around time occurs for the exploration-centric configuration which is 0.04 hrs, which constitutes a loss of 0.38%, when compared to the workflow turn-around time of the unplanned CIVET instance. Given the results we can conclude that some optimisation does occur after the termination thresholds become valid, however this optimisation is insignificant and in most cases amounts to less than 1% change in the compute or data efficiency when compared to the unplanned CIVET instance.

Table II: Comparison of the data footprint and workflow turn-around time

Type	Data Footprint/MB			Turn-around Time/hr	
	Config	Term. Gen.	Gen. 500	Term. Gen.	Gen. 500
Most Data Efficient	Exploration-centric	285.81 ± 1.64	283.19 ± 1.39	7.49 ± 0.26	7.57 ± 0.2
	Exploitation-centric	286.57 ± 1.38	283.22 ± 1.36	7.34 ± 0.32	7.36 ± 0.32
	Config	Term. Gen.	Gen. 500	Term. Gen.	Gen. 500
Most Compute Efficient	Exploration-centric	374.28 ± 8.71	374.30 ± 8.82	4.55 ± 0.02	4.51 ± 0.02
	Exploitation-centric	371.04 ± 6.74	376.39 ± 6.45	4.55 ± 0.02	4.53 ± 0.01
	Config	Term. Gen.	Gen. 500	Term. Gen.	Gen. 500
Most Compute & Data Efficient	Exploration-centric	313.44 ± 4.17	310.44 ± 4.65	4.85 ± 0.05	4.87 ± 0.06
	Exploitation-centric	314.77 ± 4.55	309.46 ± 3.59	4.81 ± 0.42	4.84 ± 0.04

6.1.3. Summary of CIVET Results. The results indicate that significant optimisation was achieved for the CIVET workflow. The worst average compute efficient solution discovered at the termination generation had a workflow turn-around time of 4.55 hrs and a data footprint of 374.28 MB. When compared to the performance of the workflow for a standard CIVET instance, an optimisation of 131% was achieved in terms of the workflow turn-around time. In terms of the data footprint an optimisation of 11% was achieved. The worst average data efficient solution discovered at the termination generation had a workflow turn-around time of 7.34 hrs and a data footprint of 286.35 MB. This represents an optimisation of 42% for the workflow turn-around time when compared to the standard CIVET instance, and an optimisation of 41% in terms of the data footprint. For the most data and compute efficient workflow plans, the exploration configuration on average had better workflow turn-around times at 4.81 hrs, but with a worse footprint at 314.77 MB. For the worst average compute and data efficient solution, the workflow had a turn-around time of 4.85 hrs and a data footprint of 313.44 MB. When compared to the most compute efficient solution the worst average compute and data efficient solution was 6.1% worse than the best average compute efficient solution and 8.8% worse than the best average data efficient solution. The results also indicate that termination criterion was able to successfully detect search stagnation and terminate the search with a minor loss in the fitness of the candidate solutions. The worst compromised performance for the workflow turn-around time amounted to a loss of 0.38% when compared to the turn-around time of the unplanned CIVET instance. In terms of data efficiency the loss was 0.8%.

7. CONCLUSIONS

We investigated the use of an evolutionary multi-objective meta-heuristic for optimising scientific workflows for distributed infrastructures. The study was motivated by the fact that e-Science workflows, which are currently one of the most widely used approaches for undertaking analysis on distributed computing infrastructure, are growing in scale and complexity. As described in this paper and highlighted by numerous

researchers, this complexity is multi-dimensional. Various approaches to workflow optimisation have been proposed that focus on optimising scientific workflows for one objective only, however the focus in this work has been to propose a method that can be used for both compute and data efficiency simultaneously. This paper also explored the use of an adaptive Hypervolume-based termination criterion for detecting search stagnation. The approach is relevant to real world applications, specifically those where the evaluation of objectives may be computationally expensive and an extensive evolutionary search may be un-feasible.

Results for the optimisation of a real world workflow, the neuroimaging CIVET workflow, comparing benefit to the currently used version were presented. The experiments were setup to evolve the set of candidate solutions for 500 generations. The performance of the most compute optimal, data optimal, as well as compute and data optimal solutions were compared to the performance of the most compute optimal, data optimal and compute and data optimal solutions at the time when the termination thresholds were valid. The maximum loss of compute performance due to the termination criterion was 0.8% in terms of the turn-around time of the most compute efficient solution at the 500th generation compared to the most compute efficient solution present at the time when the termination thresholds became valid. This loss equals a time of 1.8 minutes. The maximum loss of data efficiency on the other hand was 1.4% equaling 0.98 MB. The average termination generation on the other hand was 276. Given that the meta-heuristic was run for 500 generations, no significant optimisation was achieved after the point where the termination thresholds became valid. Therefore, no significant optimisation was lost. The results indicate that a multi-objective approach is feasible for the optimisation of scientific workflows and significant performance gains can be achieved by the application of this approach. We have also described how our presented approach can generalise to other workflows – such as Inspiral search, Epigenomics and Cybershake.

REFERENCES

ABRAMOWITZ, M. AND STEGUN, I. 1972. Stirling Numbers of the Second Kind. In *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover Publications, 824 – 825.

AD-DAB'BAGH, Y., EINARSON, D., LYTTELTON, O., MUEHLBOECK, J.-S., MOK, K., IVANOV, O., VINCENT, R. D., LEPAGE, C., LERCH, J., FOMBONNE, E., AND EVANS, A. C. 2006. The CIVET Image-Processing Environment: A Fully Automated Comprehensive Pipeline for Anatomical Neuroimaging Research. *12th Annual Meeting of the Organization for Human Brain Mapping (OHBM)*.

BADER, J. M. 2010. *Hypervolume-Based Search for Multiobjective Optimization: Theory and Methods*. CreateSpace Independent Publishing Platform, Paramount, CA.

BARGA, R. AND GANNON, D. 2007. Scientific versus business workflows. In *Workflows for e-Science*, I. J. Taylor, E. Deelman, D. B. Gannon, and M. Shields, Eds. Springer London, 9–16.

BEUME, N., NAUJOKS, B., AND EMMERICH, M. 2007. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research 181,* 3, 1653 – 1669.

BROCKHOFF, D. AND ZITZLER, E. 2007. Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods. In *IEEE Congress on Evolutionary Computation, 2007. CEC 2007*. IEEE, 2086 –2093.

CASANOVA, H., LEGRAND, A., AND QUINSON, M. 2008. SimGrid: A Generic Framework for Large-Scale Distributed Experiments. In *Proceedings of the Tenth International Conference on Computer Modeling and Simulation*. IEEE Computer Society, Washington, DC, USA, 126–131.

CHERVENAK, A. L. AND SCHULER, R. 2007. A Data Placement Service for Petascale Applications. In *Proceedings of the 2nd International Workshop on Petascale Data Storage*. PDSW '07. ACM, New York, NY, USA, 63–68.

DEB, K., AGRAWAL, S., PRATAP, A., AND MEYARIVAN, T. 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Parallel Problem Solving from Nature PPSN VI*, M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. Merelo, and H.-P. Schwefel, Eds. Lecture Notes in Computer Science Series, vol. 1917. Springer Berlin / Heidelberg, 849–858.

DEELMAN, E. 2007. Looking into the Future of Workflows: The Challenges Ahead. In *Workflows for e-Science*, I. J. Taylor, E. Deelman, D. B. Gannon, and M. Shields, Eds. Springer London, 475–481. 10.1007/978-1-84628-757-228.

DEELMAN, E., BLYTHE, J., GIL, Y., KESSELMAN, C., MEHTA, G., VAHI, K., BLACKBURN, K., LAZZARINI, A., ARBREE, A., CAVANAUGH, R., AND KORANDA, S. 2003. Mapping Abstract Complex Workflows onto Grid Environments. *Journal of Grid Computing 1,* 1, 25–39.

DEELMAN, E., GANNON, D., SHIELDS, M., AND TAYLOR, I. 2009. Workflows and e-Science: An overview of workflow system features and capabilities. *Future Generation Computer Systems 25,* 5, 528 – 540.

DEELMAN, E., SINGH, G., HUI SU, M., BLYTHE, J., GIL, A., KESSELMAN, C., MEHTA, G., VAHI, K., BERRIMAN, G. B., GOOD, J., LAITY, A., JACOB, J. C., AND KATZ, D. S. 2005. Pegasus: a framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming Journal 13*, 219–237.

DURILLO, J. J., NEBRO, A. J., AND ALBA, E. 2010. The jMetal Framework for Multi-Objective Optimization: Design and Architecture. In *CEC 2010*. Lecture Notes in Computer Science Series, vol. 5467. Springer Berlin / Heidelberg, Barcelona, Spain, 4138–4325.

GHOSH, S., DAS, S., VASILAKOS, A., AND SURESH, K. 2012. On Convergence of Differential Evolution Over a Class of Continuous Functions With Unique Global Optimum. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 42,* 1, 107 –124.

GIL, Y., DEELMAN, E., ELLISMAN, M., FAHRINGER, T., AND FOX, G. 2007. Examining the Challenges of Scientific Workflows. *COMPUTER 40,* 12, 24–35.

GOLDBERG, D. E., DEB, K., AND CLARK, J. H. 1991. Genetic Algorithms, Noise, and the Sizing of Populations. *COMPLEX SYSTEMS 6*, 333–362.

GRAPHVIZ. 2010. DOT language http://www.graphviz.org/doc/info/lang.html.

GREFENSTETTE, J. 1986. Optimization of Control Parameters for Genetic Algorithms. *IEEE Trans. Syst. Man Cybern. 16,* 1, 122–128.

GUERRERO, J. L., GARCIA, J., MARTI, L., MOLINA, J. M., AND BERLANGA, A. 2009. A stopping criterion based on kalman estimation techniques with several progress indicators. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*. GECCO '09. ACM, New York, NY, USA, 587–594.

HABIB, I., ANJUM, A., BLOODSWORTH, P., AND MCCLATCHEY, R. 2009. Neuroimaging analysis using grid aware planning and optimisation techniques. In *E-Science Workshops, 2009 5th IEEE International Conference on*. IEEE Computer Society, 102 –109.

HUANG, S. AND ZHU, Y. 2009. NSGA-II based grid task scheduling with multi-QoS constraint. In *Proceedings of the 2009 Third International Conference on Genetic and Evolutionary Computing*. WGEC '09. IEEE Computer Society, Washington, DC, USA, 306–308.

HUBAND, S., HINGSTON, P., BARONE, L., AND WHILE, L. 2006. A review of multiobjective test problems and a scalable test problem toolkit. *Evolutionary Computation, IEEE Transactions on 10,* 5, 477 –506.

JACK JR., C. R., BERNSTEIN, M. A., FOX, N., AND THOMPSON, P. 2008. The Alzheimer's disease neuroimaging initiative (ADNI): MRI methods. *Journal of Magnetic Resonance Imaging 27,* 4, 685–91.

KOSAR, T. AND LIVNY, M. 2004. Stork: Making data placement a first class citizen in the grid. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*. ICDCS '04. IEEE Computer Society, Washington, DC, USA, 342–349.

KRAUTER, K., BUYYA, R., AND MAHESWARAN, M. 2002. A taxonomy and survey of grid resource management systems for distributed computing. *Software: Practice and Experience 32,* 2, 135–164.

LINGRAND, D., MONTAGNAT, J., AND GLATARD, T. 2008. Modeling the latency on production grids with respect to the execution context. In *CCGRID '08: Proceedings of the 2008 Eighth IEEE International Symposium on Cluster Computing and the Grid*. IEEE Computer Society, Washington, DC, USA, 753–758.

MARTÍ, L., GARCÍA, J., BERLANGA, A., AND MOLINA, J. M. 2007. A cumulative evidential stopping criterion for multiobjective optimization evolutionary algorithms. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. GECCO '07. ACM, New York, NY, USA, 911–911.

NERIERI, F., PRODAN, R., FAHRINGER, T., AND TRUONG, H.-L. 2006. Overhead analysis of grid workflow applications. In *GRID '06: Proceedings of the 7th IEEE/ACM International Conference on Grid Computing*. IEEE Computer Society, Washington, DC, USA, 17–24.

PARK, S.-M. AND HUMPHREY, M. Data throttling for data-intensive workflows. In *22nd IEEE International Symposium on Parallel and Distributed Processing (IPDPS), Miami, Florida USA, April 14-18*. IEEE Computer Society.

PRODAN, R. 2007. Specification and runtime workflow support in the ASKALON Grid environment. In *Sci. Program.* Vol. 15. IOS Press, Amsterdam, The Netherlands, The Netherlands, 193–211.

PURSHOUSE, R. C. 2003. On the evolutionary optimisation of many objectives. Ph.D. thesis, Department of Automatic Control and Systems Engineering The University of Sheffield, UK.

RAICU, I., ZHAO, Y., DUMITRESCU, C., FOSTER, I., AND WILDE, M. 2007. Falkon: a Fast and Light-weight tasK executiON framework. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*. SC '07. ACM, New York, NY, USA, 43:1–43:12.

RAICU, I., ZHAO, Y., FOSTER, I. T., AND SZALAY, A. 2008. Accelerating large-scale data exploration through data diffusion. In *Proceedings of the 2008 international workshop on Data-aware distributed computing*. DADC '08. ACM, New York, NY, USA, 9–18.

RAMAKRISHNAN, L. AND PLALE, B. 2010. A multi-dimensional classification model for scientific workflow characteristics. In *Wands '10: Proceedings of the 1st International Workshop on Workflow Approaches to New Data-centric Science*. ACM, New York, NY, USA, 1–12.

REDOLFI, A., MCCLATCHEY, R., ANJUM, A., ZIJDENBOS, A., MANSET, D., BARKHOF, F., SPENGER, C., LEGRE, Y., WAHLUND, L.-O., BARATTIERI, C., AND FRISONI, G. B. 2009. Grid Infrastructures for Computational Neuroscience : the neuGRID Example. In *Future Neurology*. Vol. 6. Future Science Group Publishers, 703–722.

REEVES, C. R. 1993. Using genetic algorithms with small populations. In *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 92–99.

RUDENKO, O. AND SCHOENAUER, M. 2004. A steady performance stopping criterion for pareto-based evolutionary algorithm. *In The 6th International Multi-Objective Programming and Goal Programming Conference*.

SCHAFFER, J. D., CARUANA, R. A., ESHELMAN, L. J., AND DAS, R. 1989. A study of control parameters affecting online performance of genetic algorithms for function optimization. In *Proceedings of the third international conference on Genetic algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 51–60.

SINGH, G., SU, M.-H., VAHI, K., DEELMAN, E., BERRIMAN, B., GOOD, J., KATZ, D. S., AND MEHTA, G. 2008. Workflow task clustering for best effort systems with Pegasus. In *Proceedings of the 15th ACM Mardi Gras conference*. MG '08. ACM, New York, NY, USA, 9:1–9:8.

TALBI, E.-G. 2009. *Metaheuristics: From Design to Implementation*. Wiley Publishing.

TRAUTMANN, H., LIGGES, U., MEHNEN, J., AND PREUSS, M. 2008. A convergence criterion for multiobjective evolutionary algorithms based on systematic statistical testing. In *Proceedings of the 10th international conference on Parallel Problem Solving from Nature: PPSN X*. Springer-Verlag, Berlin, Heidelberg, 825–836.

VELDHUIZEN, D. A. V. AND LAMONT, G. B. 1998. Multiobjective evolutionary algorithm research: A history and analysis. In *Technical Report TR-98-03*. Department of Electrical and Computer Engineering, Air Force Institute of Technology, Ohio.

VON LASZEWSKI, G. AND HATEGAN, M. 2005. Workflow concepts of the java cog kit. *Journal of Grid Computing 3*, 239–258. 10.1007/s10723-005-9013-5.

XUE, F., SANDERSON, A., AND GRAVES, R. 2003. Pareto-based multi-objective differential evolution. In *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*. Vol. 2. IEEE Press, 862 – 869 Vol.2.

ZHAO, Y., HATEGAN, M., CLIFFORD, B., FOSTER, I., VON LASZEWSKI, G., NEFEDOVA, V., RAICU, I., STEF-PRAUN, T., AND WILDE, M. 2007. Swift: Fast, reliable, loosely coupled parallel computation. In *Services, 2007 IEEE Congress on*. IEEE Computer Society, 199 –206.

ZITZLER, E., DEB, K., AND THIELE, L. 2000. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation 8,* 2, 173–195.

ZITZLER, E. AND KÜNZLI, S. 2004. Indicator-based selection in multiobjective search. In *in Proc. 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII*. Springer, 832–842.

ZITZLER, E., LAUMANNS, M., AND THIELE, L. 2001. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Tech. rep., ETH Zurich.

ZITZLER, E., THIELE, L., LAUMANNS, M., FONSECA, C. M., AND FONSECA, V. G. D. 2003. Performance assessment of multiobjective optimizers: an analysis and review. *Evolutionary Computation, IEEE Transactions on 7,* 2, 117 – 132.