Workflow scheduling with data transfer optimization and enhancement of reliability in cloud data centers

Karima Oukfif, Fatima Boumghar Oulebsir, Samia Bouzefrane, Soumya Banerjee

Workflow scheduling with data transfer optimisation and enhancement of reliability in cloud data centres

Karima Oukfif*

Department of Computer Science,
University of Mouloud Mammeri of Tizi-Ouzou,
Tizi-Ouzou, Algeria
and
LRPE Lab,
USTHB,
Algiers, Algeria
Email: karima.oukfif@gmail.com
*Corresponding author

Fatima Oulebsir-Boumghar

LRPE Lab,
Faculty of Electronics and Computer Science-FEI,
USTHB,
Algiers, Algeria
Email: linda_oulebsir@yahoo.fr

Samia Bouzefrane

CEDRIC Lab,
Conservatoire National des Arts et Metiers,
Paris, France
Email: samia.bouzefrane@lecnam.net

Soumya Banerjee

Birla Institute of Technology,
Mesra, Ranchi, India
Email: dr.soumya@ieee.org

Abstract: Infrastructure as a service (IaaS) clouds offer huge opportunities to solve large-scale scientific problems. Executing workflows in such environments can be expensive in time if not scheduled rightly. Although scheduling workflows in the cloud is widely studied, most approaches focus on two user's quality of service requirements namely makespan (i.e., completion time) and costs. Other important features of cloud computing such as the heterogeneity of resources and reliability must be considered. In this paper, we present a reliability-aware method based on discrete particle swarm optimisation (RDPSO) for workflow scheduling in multiple and

heterogeneous cloud data centres. Our aim is to optimise data transfer time while minimising makespan and enhancing reliability. Based on simulation, our results show a significant improvement in terms of makespan, transferred data and reliability relative to reliability-aware HEFT method (heterogeneous earliest finish time), for the real-world workflows.

Keywords: cloud computing; workflow scheduling; data transfer; reliability; discrete particle swarm optimisation.

Reference to this paper should be made as follows: Oukfif, K., Oulebsir-Boumghar, F., Bouzefrane, S. and Banerjee, S. (xxxx) 'Workflow scheduling with data transfer optimisation and enhancement of reliability in cloud data centres', *Int. J. Communication Networks and Distributed Systems*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Karima Oukfif received her Engineer degree in Computer Engineering from UMMTO (University Mouloud Mammeri of Tizi-Ouzou, Algeria) in 2002 and Magister degree in Computer Science from UMMTO in 2006. Currently, she is Assistant Professor and PhD student at UMMTO and a member of ParIMed Team, LRPE – USTHB. Her main research field is scheduling applications and resource allocation in Cloud Computing.

Fatima Oulebsir-Boumghar is Senior Researcher and Full Professor in Electronic and Computer Sciences at Algiers University of Sciences and Technology (USTHB). She is the Head of ParIMed research team and her main research activities are devoted to parallel computing, complex applications, 3D medical imaging, shape and surface modelling, point-based graphics and computational brain imaging. She was responsible for Control Process and Interactive Rendering (CPRI) PhD program for three years. She obtained her Engineer degree from E.S.E. (Supelec, Paris, 1980) and her PhD in Signal and Image parallel processing from USTHB in 1998, in collaboration with the Laboratory of Parallel Computing (L.I.P) at ENS-Lyon. She collaborates also with LIRIS – U. Lyon2 in 3D discrete geometry, with LIB (UPMC- INSERM) in improving processing image time in the brain dMRI analysis pipeline (Freesurfer) for traumatic brain injury and with Athena-INRIA (Sophia Antipolis) in co-supervising PHD thesis in the brain dMRI field.

Samia Bouzefrane is an Associate Professor and has the accreditation to conduct research (HDR) at the Conservatoire National des Arts et Metiers (CNAM) of Paris. She received her PHD in Computer Science from the University of Poitiers (France) in 1998. After four years at the University of Le Havre (France), she joined in 2002 the CEDRIC Lab of CNAM. She is the co-author of many books (Operating Systems, Smart Cards and Identity Management Systems). Her current research areas cover security in cloud computing and internet of things, resource allocation in cloud computing, and new paradigms for networking such as NFV and NDN.

Soumya Banerjee obtained his Bachelor in Engineering [BE (Hons.)] degree in Computer Science from NIT Nagpur (Maharastra) in 1995 and completed his PhD from BIT Mesra, India. He spend more than two years with Microsoft Research at Seattle, USA as a Summer School Fellow. He was

associated as a Senior Programmer capacity in ISRO and served as a Project Leader (technical) in Cognizant Technology Solution, ICICI InfoTech both in India, Southeast Asia and Europe. He has more than 125 international journal publications including 25 book chapters and 43 international top level conference proceedings published from Elsivier Science, IEEE Transactions, ACM, Springer-Verlag Germany, CRC Press, and Idea Publication USA to his credit covering bio inspired intelligence, soft computing and optimisation, hybrid intelligence, social networking applications and social media, Wiki analysis, machine learning with complex system and evolutionary computing.

1 Introduction

The consistent requirements of workflow applications in data and computing involve a greater performance computing environment in order to perform them in a reasonable time and without failures. The problem of workflow scheduling is NP-hard (Ullman, 1975) and has been widely studied over the years in the context of distributed systems, mainly in grid environments. However, with the emergence of the cloud computing infrastructures, scientific-workflows scheduling still remains a fundamental issue. Cloud computing is a model for enabling on-demand network access to a dynamic shared-resources pool that can be rapidly provisioned (Mell and Grance, 2011) and that makes the satisfaction of some quality of service (QoS) constraints difficult. Several QoS constraints are used in workflow scheduling problems in the context of grids (Chen and Zhang, 2009; Bouali et al., 2015; Oukfif et al., 2015; Kianpisheh et al., 2016) or cloud systems (Chen and Zhang, 2012; Jian et al., 2014; Poola et al., 2016; Rehani and Garg, 2017; Choudhary et al., 2018). Among all these constraints, optimising makespan and cost are the two prime constraints that are broadly studied in Wu et al. (2015). However other QoS constraints, like reliability, can be useful because of their impact in workflow scheduling problems for cloud systems to achieve acceptable performance. In cloud systems, users pay to use cloud resources. Thus they are more demanding on performance and reliability. However, computing resources and networks are not failure free and any type of failure may be crucial to the applications execution.

Using reliable scheduling algorithm can deal with such failures but it is known to be a NP-hard problem (Hakem and Butelle, 2007). This is the reason why many heuristics (Hakem and Butelle, 2007; Doğan and Özgüner, 2005; Dongarra et al., 2007; Fard et al., 2012; Kumar et al., 2018) and meta-heuristics such as genetic algorithm (GA) (Wang et al., 2011; Rehani and Garg, 2017), particle swarm optimisation (PSO) (Chen and Zhang, 2012; Bouali et al., 2015) and ant colony optimisation (ACO) (Chen and Zhang, 2009; Kianpisheh et al., 2016) have been proposed for that issue.

Unfortunately, most studies are based on a reliability model of resources which is not appropriate for real cloud systems and hence neglecting the resources heterogeneity and the network links reliability mostly in multiple cloud data centres. Moreover, most approaches do not consider the extra-time incurred during data transfers between tasks which can influence the performance of the whole application. Indeed, in multiple cloud data centres, resources are deployed on different regions and therefore data transfer times become significant and directly influence the response time of workflow applications. In

fact, cloud services that rely on virtual machines (VMs) are located in several distributed data centres, hence requiring to consider data transfer times and network reliability.

In this work, we propose a reliability-aware strategy based on discrete PSO (DPSO) for workflow scheduling in multiple cloud data centres where reliability is specified jointly by those of computing resources and network links. Our approach pays attention to minimising the workflow completion time, including both the task computation time and the data transfer time, while enhancing the overall reliability of resources and network links. The remainder of this paper is organised as follows. Section 2 reviews the related work, followed by our reliability-aware scheduling algorithm formalisation, and scheduling based on DPSO in Sections 3 and 4, respectively. Our proposed approach is evaluated in Section 5 and Section 6 concludes the paper.

2 Related work

Previous work was done to optimise data transfer and reliability in distributed environments such as grids, and more recently, these two constraints have been re-examined by many researchers for cloud environments.

In this section, we address existing research related to optimising data transfer and enhancing reliability constraints in distributed computing systems such as grid and cloud infrastructures.

In the literature, we note these important works of Hakem and Butelle (2007), Doğan and Özgüner (2005) and Dongarra et al. (2007) which address the problem of optimising reliability considering times of data transfer.

In fact, Hakem and Butelle (2007) used a list scheduling heuristic called BSA (Bi-objective Scheduling Algorithm) for scheduling parallel applications on heterogeneous distributed computing systems which considers not only the makespan but also the application failure probability. Doğan and Özgüner (2005) presented two bi-objective scheduling heuristics for minimising time and enhancing reliability of applications in heterogeneous computing systems. The first one was an improvement of the traditional dynamic-level scheduling (DLS) algorithm (Sih and Lee, 1993) called bi-objective dynamic level scheduling (BDLS). The second one was based on genetic algorithm and is called bi-objective genetic algorithm (BGA) optimising the same criteria as the first one.

A bi-objective scheduling algorithm (Dongarra et al., 2007) which optimises makespan and reliability on heterogeneous systems was presented for independent tasks. The authors demonstrated that tasks should be mapped to the resource so that the product of task time and resource failure rate can be minimised. This product was then used for enhancing the HEFT algorithm (Topcuoglu et al., 2002) to consider reliability giving the reliable-HEFT heuristic.

Chen and Zhang (2009) have considered reliability, time and cost as QoS parameters while scheduling workflows in grids. The proposed algorithm optimised a parameter while meeting constraints of the two others. The algorithm is based on ACO where seven heuristics were defined and selected on pheromone values. Thereafter, the same authors presented a set base version of discrete PSO (S-PSO) for cloud workflow scheduling, taking into account the following users QoS constraints: deadline, budget, and reliability (Chen and Zhang, 2012). These constraints are optimised separately since users are encouraged to specify one preferred QoS as an optimisation objective.

However, in their paper, no cloud model specification was given. In addition, the reliability constraint has not been modelled. It has been just specified by the thresholds (like being no smaller than a user-defined variable *MinReliability*) and thus data transmission reliability was not specified.

In contrast, in the work of Jian et al. (2014), a PSO algorithm for workflow scheduling in cloud to enhance reliability was used where data transmission reliability is considered. But unfortunately, the reliability of the cloud resources and data transmission were expressed as levels that lack precision (values in the interval of [1, 5] to set).

Lee et al. (2010) discussed the effect of the reliability of workflow execution on its cost in distributed computing systems and then they proposed an algorithm called reliability for profit assurance (RPA) for workflow scheduling that incorporates a cost-aware replication scheme to enhance reliability. Task replication approaches are frequently adopted to deal with resource failures. However, replicas created for enhancing reliability are often leading to the waste of resources. The reliability model adopted by the authors did not incorporate communication links reliability.

The reputation-based look-ahead genetic algorithm (LAGA) was presented by Wang et al. (2011) for large-scale distributed systems. LAGA has been optimised in terms of completion time and failure rate. The completion time was used to obtain a task order at each generation, and then the lowest failure rate was taken into account to select a resource in a mutation operation. We note that the LAGA authors focused on intensive computer applications where link reliability and communication time between tasks were not modelled.

In some work, maintaining reliability is in relation to energy consumption. Fard et al. (2012) suggested a multi-objective list scheduling (MOLS) method for workflow execution in distributed computing systems. They used a Pareto optimality to achieve four objectives: makespan, financial cost, energy consumption and reliability.

Guo et al. (2018) proposed an algorithm called Deadline Reliability-Based Fault-Tolerant Scheduling Algorithm for Energy (DRB-FTSA-E) which optimises energy and reliability while meeting the time constraint in a heterogeneous system.

Even recently, the management of system reliability has attracted researchers' attention (Kianpisheh et al., 2016; Wen et al., 2016; Poola et al., 2016; Zhang et al., 2017; Rehani and Garg, 2017; Xiao et al., 2018). Kianpisheh et al. (2016) proposed a constrained-based workflow scheduling algorithm to maximise reliability. They proposed three novel workflow scheduling heuristics which were based on ant colony system. The aggregation of these heuristics minimised violations of time, cost and reliability constraints. For simplicity, the network was assumed to be reliable which is unrealistic in multiple cloud data centres environments. Wen et al. (2016) presented a deploying workflow applications algorithm on federated clouds. Their method is able to evaluate the most reliable scheme that meets application security requirements and that optimises its cost. Poola et al. (2016) used the adaptive and just in-time (AJIT) scheduling algorithm to support fault tolerance. Despite targeting the cloud by their resource model, the network reliability was not modelled with the AJIT approach.

A bi-objective genetic algorithm (BOGA) was suggested by Zhang et al. (2017) to optimise energy consumption and reliability of a parallel application executed on heterogeneous processors of a cluster, but the reliability of the communication's links has not been taken into account. Rehani and Garg (2017) used the NSGA-II meta-heuristic for a multi-objective workflow scheduling in cloud. Their cloud model is a single site where the bandwidth linkage is supposed to be a uniform speed

and links reliability was completely ignored. Xiao et al. (2018) proposed a heuristic called maximising reliability with energy constraint (MREC) for each task of a parallel application on heterogeneous distributed systems. The reliability was expressed based on the failure rate of the processor on which a task was assigned to, but network reliability was not considered despite the fact that the adopted resource model was for distributed systems.

Finally, a lot of research has focused on data transfer regardless of the reliability constraint. For example, Babu and Krishna (2013) proposed an adaptable time-cost heuristic to handle time-basic workflows with least cost. A k-means clustering-based data placement strategy coupled with a multilevel task replication approach is used by Zhang et al. (2015). The aim was to minimise the overall data transfer between multiple data centres of a cloud without taking into account the reliability constraint. Thereafter, Chen et al. (2017) proposed an approach that partitioned graphs in order to execute data-intensive scientific workflow in distributed data centres, and that optimised the overall data transfer cost.

Table 1 Criteria studied by previous prime work and their execution platforms compared to our approach

Paper	*Algorithm*	*Makespan*		*Reliability*		*Platform*
		Ex.	*Comm.*	VM_R	$Link_R$	
Hakem and Butelle (2007)	BSA	✓	✓	✓	✓	DCS
Doğan and Özgüner (2005)	BDLS, BGA	✓	✓	✓	✓	DCS
Dongarra et al. (2007)	Reliable-HEFT	✓	✓	✓		cluster
Chen and Zhang (2009)	ACO	✓		✓		grid
Chen and Zhang (2012)	S-PSO	✓	✓	✓		cloud-S
Lee et al. (2010)	RPA	✓		✓		DCS
Wang et al. (2011)	LAGA	✓		✓		DCS
Fard et al. (2012)	MOLS	✓	✓	✓		grid
Kianpisheh et al. (2016)	ACO	✓	✓	✓		grid
Poola et al. (2016)	AJIT	✓	✓	✓		cloud-S
Zhang et al. (2017)	BOGA	✓	✓	✓		cluster
Rehani and Garg (2017)	NSGA-II	✓		✓		cloud-S
Xiao et al. (2018)	MREC			✓		DCS
Guo et al. (2018)	DRB-FTSA-E	✓	✓	✓		DCS
Our proposal	RDPSO	✓	✓	✓	✓	cloud-M

Table 1 summarises the main previous work compared to our approach (RDPSO). In the category makespan, Ex. and Comm. refer to execution time and communication time respectively. For the category reliability, VM_R and $Link_R$ denote respectively the reliability of VM and the reliability of Links. We note that with the exception of the studies (Chen and Zhang, 2012; Poola et al., 2016; Rehani and Garg, 2017) that were realised in a single data centre cloud (cloud-S), other studies have been performed either in distributed computing systems (DCS), clusters or in a grid platforms while our approach is modelled and validated on multiple cloud data centres (cloud-M). Note that in this table, we only presented the makespan and the reliability constraints. Other constraints, such as cost and energy, of some studies are not presented.

3 Scheduling problem formalisation

In our context, a schedule is defined in terms of application mapping, a resource infrastructure and performance criteria. In this work, the criteria considered are the makespan or completion time (including data transfer times) and reliability.

In the following sub-sections, before formalising these parts, we recall the issue we want to address and the particularities of our context.

3.1 Our context

Our approach is proposed in the context of an IaaS cloud with multiple data centres, where workflow applications have to be scheduled on different data centres for execution. In order to achieve the goal of minimising application completion time while enhancing reliability, we clearly model data transfer times when computing the makespan and integrate the reliability of communication (link reliability) to express the reliability of the resources.

3.2 Application model

A scientific workflow application (w) is modelled as a directed acyclic graph (DAG): $G = (N, E)$, where N is a set of n nodes designating tasks t_i $(1 \leqslant i \leqslant n)$, and E is a set of directed edges. An edge $e(i, j) \in E$ corresponds to a dependence constraint between task t_i and task t_j , in which t_i is an immediate parent task of t_j, and t_j the immediate child task of t_i. A child task cannot be executed until all of its parent tasks are completed. The data matrix, with $n.n$ dimensions, represents the amount of data exchanged between tasks.

3.3 Cloud resource model

In this work, we present a resource model that considers the data transfer times between multiple data centres so that computer resources can be deployed on different data centres or regions. Also, we consider resources reliabilities of both computer resources and network links.

3.3.1 Time model

In IaaS cloud, computer resources are instantiated as virtual machines (VMs). Typically, cloud providers offer multiple types of VM. Each vm_i is defined in terms of its computing capacity CC_{vm_i}. We suppose that for every VM type, we can estimate its computing capacity in floating point operations per second ($FLOPS$). Thus we can estimate the execution time of a task on a given VM type. Besides, we define the computing time ($CT(t_i, vm_p)$) as the estimated execution time of the task t_i in a VM of type vm_p according to its size I_{t_i} likewise defined in floating point operations ($FLOP$). It is calculated as in equation (1).

$$CT(t_i, vm_p) = \frac{I_{t_i}}{CC_{vm_p}} \tag{1}$$

In multiple cloud data centres, VMs are deployed on different regions. Therefore data transfer times become significant and directly influence the response time of the workflow applications. Hence, we consider these times when calculating the workflow's completion time. We consider the number U of data centres in the cloud infrastructure. We define the Transfer Time $TT_{(i,j)}$ between two tasks corresponding to a weight of an edge $(i, j) \in E$ in the application graph (DAG), which corresponds to the time taken to transfer data from task t_i (executed on vm_p lodged in data centre U_a) to task t_j (executed on vm_k lodged in data centre U_b), as in the following equation.

$$TT_{(i,j)} = \frac{data_{ij}}{Transfer_{rate_{(p,k)}}} \tag{2}$$

where $data_{ij}$ is the size of the output data produced by task t_i and transferred to t_j.

Since VMs are located on different data centres, the transfer rates or the bandwidths $Transfer_{rate_{(p,k)}}$ between them are heterogeneous. We note that when two communicating tasks are executed on the same VM, the transfer time is equal to zero and when they run on different VMs hosted in the same data centre, the transfer time is also neglected. Thus, we propose a new model that describes multiple cloud data centres platforms. In our model, the Total Computing Time $TCT(t_i, vm_p)$ of a task in a VM is computed as depicted in equation (3). In this equation, m refers to the number of edges in which t_i is a parent task. The Boolean s_m is equal to 0 whenever t_i and t_j run on the same VM ($p = k$) or 1 otherwise. Also, r_m is equal to 0 whenever vm_p and vm_k are hosted in the same data centre ($U_a = U_b$) or 1 otherwise.

$$TCT(t_i, vm_p) = CT(t_i, vm_p) + \sum_{i=1}^{m} s_m.r_m.TT_{(i,j)} \tag{3}$$

The total computing time $TCT(w)$ of all tasks in the workflow is defined as in equation (4).

$$TCT(w) = \max\{TCT(t_i, vm_p)\} \tag{4}$$

3.3.2 Reliability model

We assume that the failure of VMs and communication links in the system follows a *Poisson* process which is the simplest and most used process that models the occurrence of failures (Zhang et al., 2017). Then we associate a failure rate for each of the VM and communication link. Let λ_p denotes the failure rate by time unit of vm_p and $\lambda_{L_{(k,p)}}$ the failure rate of the communication link $L_{(k,p)}$ between vm_k and vm_p. These failure rates can be derived from the resource's profiling service, file log, and statistical prediction techniques (Tang et al., 2012).

In this model, the reliability of a task t_i is equal to the probability of its successful execution on the VM to which it is assigned, and the probability of successful data transfers to that task from its immediate predecessors. We use the network reliability approximation for data transmission as proposed by Qin and Jiang (2006) and Tang et al. (2012). The reliability $R(t_i, vm_p)$ of t_i executed or mapped on vm_p is computed as follows:

$$R(t_i, vm_p) = e^{-\lambda_p.CT(t_i, vm_p)} \tag{5}$$

Since the communication time between the tasks assigned on the same VM is zero, then the data transfer is considered as failure free. Thereby, the term $R_{(j,i)}$ corresponding to the reliability of the transfer between t_j executed on vm_k and t_i executed on vm_p , can be computed as follows:

$$R_{(i,j)} = e^{-\lambda_{L_{(k,p)}}.TT_{(j,i)}} \tag{6}$$

From equations (5) and (6), the reliability of task t_i, denoted as $R(t_i)$, can be defined by the equation (7), where $pred_{t_i}$ is the set of predecessors tasks of the task t_i.

$$R(t_i) = \prod_{t_j \in pred_{t_i}} R(t_i, vm_p) R_{(i,j)} \tag{7}$$

The reliability of a workflow w is equal to the reliability of all its tasks and can be expressed as follows:

$$R(w) = \prod_{t_i \in N} R(t_i) \tag{8}$$

3.3.3 Criteria of the model

The goal of our scheduling approach is to enhance the reliability and minimise the makespan and data transfers of the workflow application. To maximise the reliability of workflow scheduling, we need to minimise the following reliability index (i.e., the failure factor):

$$R_{index}(w) = \sum_{vm_p \in VM} \sum_{t_i \in N} (\lambda_p.CT(t_i, vm_p) + \lambda_{L_{(k,p)}}.TT_{(j,i)}) \tag{9}$$

The problem can be formally defined as follows: find a schedule S for the workflow (w) with a minimum $TCT(w)$ and a maximum $R(w)$ (thus with a minimum $R_{index}(w)$).

To solve this problem, we propose a strategy based on the PSO meta-heuristic. PSO has fewer parameters than other meta-heuristics as GA or ACO, and it converges promptly.

4 Reliability-aware scheduling based on particle swarm optimisation

In this section, we will first give a brief description of PSO heuristic. Then we will highlight the Reliability-aware Discrete PSO that we have developed and called RDPSO. PSO is a population-based meta-heuristic inspired by social behaviour patterns such as bird flocking and fish schooling. It was introduced by Kennedy and Eberhart in 1995 (Kennedy, 2011), and has been widely utilised. The stochastic optimisation algorithm is based on a concept called a particle. A particle corresponds to an individual (i.e., fish or bird) that has the ability to move or fly through the defined problem space. Each particle is characterised by a position X, a velocity V, and a fitness value. Particles positions are considered as potential solutions to the problem and are evaluated by a fitness function. We seek to optimise this fitness function and enabling it to keep track of

their best position $pbest$ and the global best position $gbest$. Particle's velocity represents the direction and the magnitude of the next movement. It is calculated by considering its actual velocity, $pbest$ and $gbest$ at each step of the algorithm. The movement of particles towards these values is inspired by its actual velocity (weighted by an inertia factor) as well as the gaps between its position and its $pbest$ and the $gbest$ value (weighted by random numbers for the cognitive and the social coefficients respectively).

In each iteration of the PSO algorithm, the fitness function is generated. Then $pbest$ and $gbest$ are updated. After that, the position and the velocity of a particle are updated based on equations (10) and (11) respectively. The PSO algorithm iterates until a stopping criterion is satisfied, which is generally a predefined fitness value estimated to be acceptable or a specified maximum number of iterations. The pseudo code of the PSO algorithm is presented in Algorithm 1.

$$v_i^{k+1} = \omega v_i^k + c_1 r_1 [pbest_i^k - x_i^k] + c_2 r_2 [gbest_i^k - x_i^k] \tag{10}$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \tag{11}$$

where

v_i^k	velocity of particle i at iteration k
v_i^{k+1}	velocity of particle i at iteration $k+1$
ω	inertia weight
x_i^k	current position of particle i at iteration k
$pbest_i$	best position of particle i
$gbest$	position of best particle in a population
x_i^{k+1}	position of the particle i at iteration $k+1$
c_1, c_2	acceleration coefficients
r_1, r_2	random number between 0 and 1.

In this work, we use a discrete variant of PSO named discrete particle swarm optimisation (DPSO) that handles tasks dependencies as achieved in Bouali et al. (2015) and Oukfif et al. (2015).

Every particle position is a possible solution. In our model, a particle has n dimensions that correspond to the n tasks of the workflow. The n tasks must be assigned to m VMs, where $m < n$. The particle position pattern shown in Figure 1 is a possible mapping solution.

Figure 1 A particle mapping

t_1	t_2	t_3	…	t_n
vm_2	vm_1	vm_m	…	vm_1

Algorithm 1 PSO algorithm

```
Data: W: a workflow;
Result: S: a schedule
begin
    Set particle dimension as equal to the number of tasks N;
    foreach particle do
        Initialise positions and velocity randomly;
    end
    foreach particle do
        Initialise pbest and gbest to the current position;
    end
    while stopping criterion is not satisfied do
        foreach particle X_i do
            Get fitness value F(X_i) of the particle using the equation (12);
            if F(X_i) < pbest(X_i) then
                pbest(X_i) = F(X_i)
            end
            if pbest(X_i) < gbest then
                gbest = pbest(X_i)
            end
            Update particle position and particle velocity as in equations (10) and (11);
        end
    end
end
```

Our approach based DPSO aims to optimise makespan and to enhance reliability metrics simultaneously. We called it RDPSO for reliability-aware DPSO. To evaluate the solutions with RDPSO, the fitness function is expressed using both the metrics makespan and reliability.

We used the weighted sum ratios to incorporate makespan and the reliability index into a single objective fitness function as indicated in formula (12). Note that the makespan and the reliability index are normalised in this equation. The normalisation allows to equalise the scale of the compared values. In fact, normalisation is necessary so that the fitness function is not biased by one of the metrics having a high value. In our algorithm, the T_{max} and the $R_{index_{max}}$ values represent the maximum value of makespan and R_{index} metrics respectively found in the previous iteration. The parameters α_1, α_2 are the relative weights assigned to makespan and reliability index respectively according to the user's compromise requirement, such as $\alpha_1 + \alpha_2 = 1$.

$$Fitness = \alpha_1 \frac{TCT}{T_{max}} + \alpha_2 \frac{R_{index}}{R_{index_{max}}} \tag{12}$$

In the PSO algorithm, to evaluate the quality of the solution, the fitness function is estimated after calculating $TCT(w)$ and $R_{index}(w)$ using equations (4) and (9) respectively.

5 Experimental results

In this section, we introduce the experiments realised in order to assess the performance of the proposed approach. We developed a simulator based on CloudSim tool (Calheiros

et al., 2011) to simulate a cloud with multiple data centres platforms. We used various workflows from different scientific domains with different structures and we implemented a standard approach for the comparison. We tested our approach with simulations, but to get closer to the real world, we used VM configurations offered by a public cloud provider and real workflows-based applications.

5.1 IaaS infrastructure settings

We modelled an IaaS cloud offering several heterogeneous data centres (3 or 5) each having three or four different types of VMs. The features of the virtual machine instances are based on the standard of Amazon Elastic Computing Cloud EC2 instance (http://aws.amazon.com/ec2) offerings and are shown in Table 2.

Table 2 Type of VMs used in our experiments

VM type	*Name*	*EC2 vCPU*	*Clock speed (GHz)*
1	m3.medium	1	2.5
2	m3.large	2	2.5
3	m3.xlarge	4	2.5
4	m3.2xlarge	8	2.5

In each data centre, three or five VMs are provisioned. We use the notation $(i)dcx(j)vm$ to indicate the configuration for the platform used for every test where (i) indicates the number of data centres and (j) indicates the maximum number of VMs in each data centre. For example, the notation 3dcx5vm indicates that the platform has three data centres each having five VMs and thus the total number of VMs is 15. Thereby we have four platforms as described in Table 3. The heterogeneity of data centres is considered by using different numbers and types of VMs. On the platforms with five data centres, we use all types of VMs including type 4 which is the most efficient. On platforms with three data centres, we only have the first three types listed in Table 3. For the network configuration, we use heterogeneous transfer rates means between VMs instances set at 500 Mbps to 1 Gbps, and between data centres set at 50 Mbps to 100 Mbps.

Table 3 Data centres, total number and types of VMs for each platform

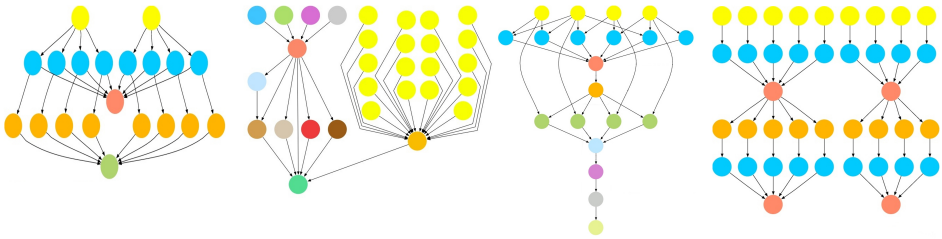
Platform	*Data centres*	*# total of VMs*	*VMs' type*
3dcx3vm	dc1, dc2, dc3	9	1, 2, 3
3dcx5vm	dc1, dc2, dc3	15	1, 2, 3
5dcx3vm	dc1, dc2, dc3, dc4, dc5	15	1, 2, 3, 4
5dcx5vm	dc1, dc2, dc3, dc4, dc5	25	1, 2, 3, 4

The instances of failure probabilities for VMs and for all the network links are set from 10^{-3} to $10^{-4}/h$ (Doğan and Özgüner, 2005; Wang et al., 2011).

5.2 Workflow applications

We use four different real-world scientific workflows. We choose two data intensive workflows (Montage and CyberShake) and two compute intensive workflows (LIGO Inspiral and SIPHT). These applications are published by Pegasus project and well described by Juve et al. (2013).

Figure 2 Scientific workflows, (a) CyberShake (b) SIPHT (c) Montage (d) LIGO Inspiral (see online version for colours)



The CyberShake workflow, is par excellence a data-intensive application. It is an application of seismology used to describe earthquakes by generating synthetic seismograms. Montage, is another data-intensive workflow used in astronomy area, generating custom mosaics of the sky using a set of input images. SIPHT, characterised to be a CPU-intensive application, is used in bioinformatics to manage the process of finding genes encoding RNAs for any bacterial replicon. Finally, the LIGO Inspiral is an application in the field of physics for the detection of gravitational waves. It is considered as a CPU-intensive application. Detailed workflow descriptions are available in DAX format (https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator) and include the DAGs, the sizes of data transferring and the reference execution time [based on Xeon@2.33 GHz CPUs ($cu = 8$)]. Figure 2 shows the basic structures of the four workflows.

To assess the performance of RDPSO, we used four sizes of workflows instances, which are: small (25 or 30 tasks), medium (50 or 60 tasks), large (100 tasks) and extra large (1,000 tasks).

5.3 Algorithms settings

To set up the RDPSO algorithm, we define its parameters as follows: the inertia weight $\omega = 0.9$, the acceleration coefficients $c_1 = c_2 = 2.0$ and the fitness parameters $\alpha 1 = \alpha 2 = 0.5$. During the executions we used 50 particles in 100 to 500 iterations. These values are adopted from similar works (Bouali et al., 2015; Oukfif et al., 2015; Jian et al., 2014). The results are average over 10 trials.

To evaluate our algorithm, we used a version of the HEFT algorithm (Topcuoglu et al., 2002) as a baseline after improving it to take into account reliability tasks as realised by Dongarra et al. (2007). Then we adapted it to the model we proposed in Subsection 3.3 [equation (3)] to support execution in multiple cloud data centres platforms. We call this algorithm RHEFT for reliability-aware HEFT.

The HEFT algorithm operates in two phases:

1 The prioritisation phase where priorities for all tasks are calculated. Then a tasks list is generated by sorting tasks in respect to their descending priorities.

2 The resources selection phase where tasks are assigned to resources that minimise their EFT (earliest execution finish time). The EFT of a task is calculated as the time at which its execution ends, which is, its starting time added to its execution time on a computer resource.

The RHEFT algorithm that we implemented improves HEFT by selecting resources (VMs) that minimises makespan and enhances reliability during resources selection phase. Furthermore communication times are calculated using the equation (3) for the EFT.

5.4 Results and discussions

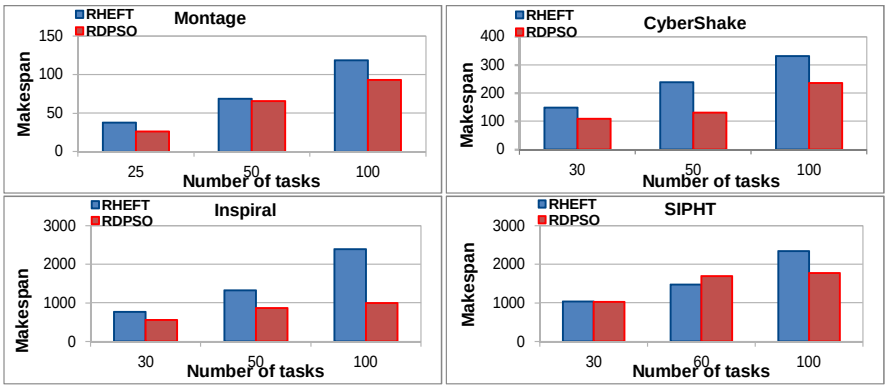
We evaluated our solution under various configurations of platforms (3dcx3vm, 3dcx5vm, 5dcx3vm and 5dcx5vm) and using four well known applications. The results of makespan, transferred data and the overall reliability were compared with those obtained by the RHEFT algorithm. We noted that since the data transfer times depend on the amount of the transferred data, we chose to present the results according to this latter.

5.4.1 Impact of the workflow size

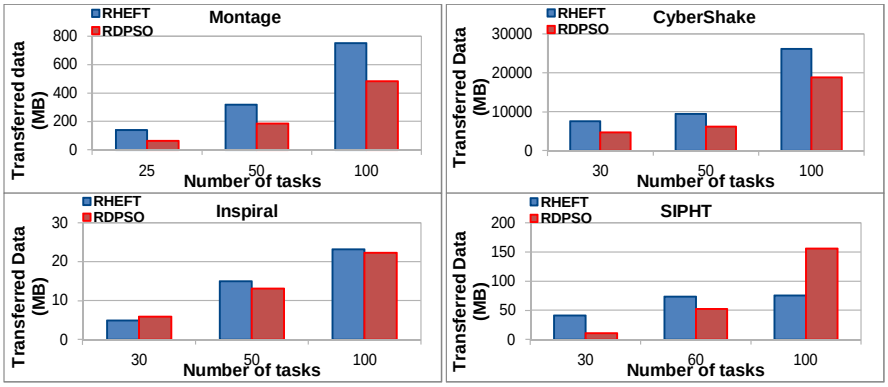
First, our experiments aim to study the performance of our approach while scheduling scientific workflows with increasing complexity. For this purpose, we varied the size of the workflows in a number of tasks. We chose to present the results obtained in 3dcx5vm platform, but they are alike on the all used platforms. The experimental results that are plotted in Figure 3(a), 3(b) and 3(c) represent makespan, amount of transferred data and overall reliability respectively for different number of tasks. We used the four workflow applications with 30 (25 for Montage), 50 (60 for SIPHT) and 100 tasks.

For both RHEFT and RDPSO algorithms, as the number of tasks of each workflow augments, the makespan, increases but is clearly improved by RDPSO [Figure 3(a)]. The schedule generated by RDPSO produced values of makespan that decreased by about 28% for Montage that has 25 tasks and that exceeds 30% for CyberShake that has 30 tasks compared to those obtained with RHEFT. The makespan RDPSO enhancements are very pronounced for all workflows except for SIPHT having 60 tasks for which RHEFT outperforms RDPSO. The parallel structure of these workflows makes that several tasks have the same value of the rank but ordered in a fixed list with HEFT. This is not the case with RDPSO where the execution order of the parallel tasks can change from one iteration to another so as to improve makespan.

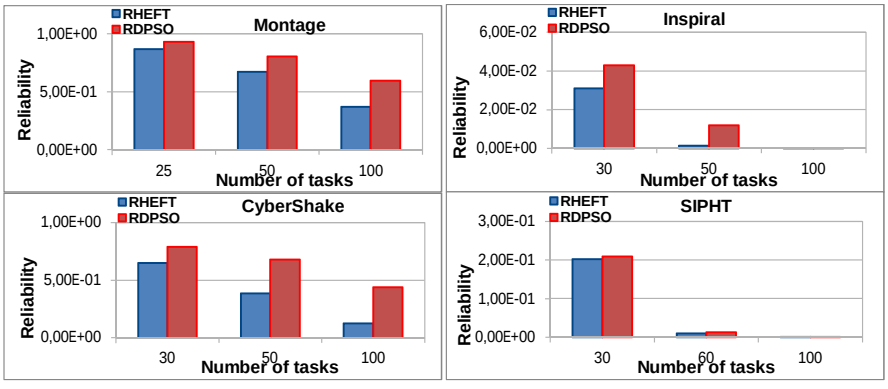
Figure 3 (a) Makespan, (b) amount of transferred data and (c) reliability variations on 3dcx5mv platform for Montage, CyberShake, Inspiral and SIPHT, with 30, 50 and 100 tasks (see online version for colours)



(a) Makespan



(b) Transferred Data



(c) Reliability

The values obtained for transferred data [Figure 3(b)] with Montage and CyberShake workflows increase with respect to the number of tasks since these two workflows are data intensive. These values are reduced with RDPSO compared to those registered with RHEFT. For the Inspiral workflow, the results obtained with RDPSO are closer to those presented by RHEFT. Our approach recorded amounts of transferred data which are reduced compared to RHEFT for SIPHT application having 30 or 60 tasks but it fails to reduce these values for the SIPHT workflow that has 100 tasks.

Concerning reliability [Figure 3(c)], the values decrease while the number of the tasks increases for the two algorithms. Likewise, RDPSO recorded best values compared to RHEFT for Montage, CyberShake and Inspiral workflows.

We summarise in Table 4 the results obtained for workflows that have 100 tasks on 3dcx5vm platform.

Table 4 Comparison of results for workflows having *100* tasks in 3dcx5vm platform

Workflow	*Makespan*		*Transferred data*		*Reliability*	
	RHEFT	*RDPSO*	*RHEFT*	*RDPSO*	*RHEFT*	*RDPSO*
Montage	117.22463	*78.72292*	750.45070	*483.54444*	3.71E-01	*5.97E-01*
CyberShake	331.8782	*235.79208*	2,857.28185	*2,393.98899*	1.27E-01	*4.39E-01*
Inspiral	2,384.8249	*999.06190*	19.38876	*18.49763*	8.37E-08	*2.73E-04*
SIPHT	2,336.69602	*1,857.78642*	75.57317	156.14683	1.23E-04	*1.41E-03*

Table 5 Comparison of results for workflows having *1,000* tasks in 5dcx3vm platform

Workflow	*Makespan*		*Transferred data*		*Reliability*	
	RHEFT	*RDPSO*	*RHEFT*	*RDPSO*	*RHEFT*	*RDPSO*
Montage	515.45287	667.51229	11,066.9285	11,205.40756	4.12E-03	1.51E-04
CyberSHake	1,113.93309	1,164.10414	167,035.5456	*159,361.5927*	1.58E-10	4.09E-12
Inspiral	13,378.0107	*8,134.19217*	309.16844	*303.666252*	3.77E-23	1.76E-64
SIPHT	8,475.70674	16,174.637	1,771.13884	*1,533.76228*	7.57E-21	8.77E-51

Using extra large workflows, the results obtained are not improved by RDPSO in all cases because of the widening of the meta-heuristic search space due to the large dimension of the particles (Table 5). Regarding the makespan, the values recorded for the Montage and CyberShake applications with RDPSO are higher but remain close to those found by RHEFT. The makespan is improved only for the Inspiral workflow. Transferred data is decreased in most cases, but overall reliability has not been improved.

5.4.2 Impact of the characteristics of the platform

In this subsection, we show the effect of resource characteristics and their number on the studied criteria (makespan, transferred data and reliability). Workflows with 100 tasks (Montage-100, CyberShake-100, Inspiral-100 and SIPHT-100) are selected to examine the impact of the number of VMs and data centres on the makespan, the amount of transferred data and the reliability with both approaches. Figures 4(a), 4(b) and 4(c) represent the results obtained on different platforms. We observe that from

platforms with three data centres (3dcx3mv and 3dcx5mv) to those with five data centres (5dcx3mv and 5dcx5mv), the makespan decreases considerably with the increase of the number of VMs or their types are more efficient [Figure 4(a)]. And we know that using more VMs allows us to take advantage of parallelism, mainly in our case as all the workflows used have a partial parallel structure. The makespan values recorded on platforms with three data centres were decreased on platforms with five data centres by about 40% for CyberShake, by more than 45% for Inspiral and about 50% for Montage and SIPHT applications. The results obtained with the four workflows confirmed that RDPSO has the lowest makespan than RHEFT in most cases. For example, the improvement performed by RDPSO compared to RHEFT is about 25% for Montage, 28% for CyberShake, and 25% to 50% for Inspiral on platforms composed of three data centres.

The amount of transferred data [Figure 4(b)] increases with the number of data centres. We explain this by the fact that the tasks of the workflow are needed to be run in more data centres which implies more transfers. Comparing the values obtained on the platforms 3dcx5vm and 5dcx3vm, although these two platforms have the same number of VMs (15 vms), the transferred data quantities are not equivalent since in 5dcx3vm we have to make more transfers between more data centres than in 3dcx5vm platform. The results remain better with RDPSO for Montage and CyberShake applications. The improvement reaches 35% for Montage on 3dc5vm and 33% for CyberShake on 5dcx5vm. We notice that concerning the platforms 3dcx3vm and 3dcx5vm, despite dealing with three data centres in each one, tasks are assigned to more VMs in the 3dcx5vm platform in each data centre than in 3dcx3vm platform, so data transfers decease more considerably in 3dcx5vm than in 3dcx3vm for CyberShake and Montage applications. The decrease of the transferred data is less emphasised for Inspiral workflow and exceptionally not optimised for SIPHT application which has a complex structure.

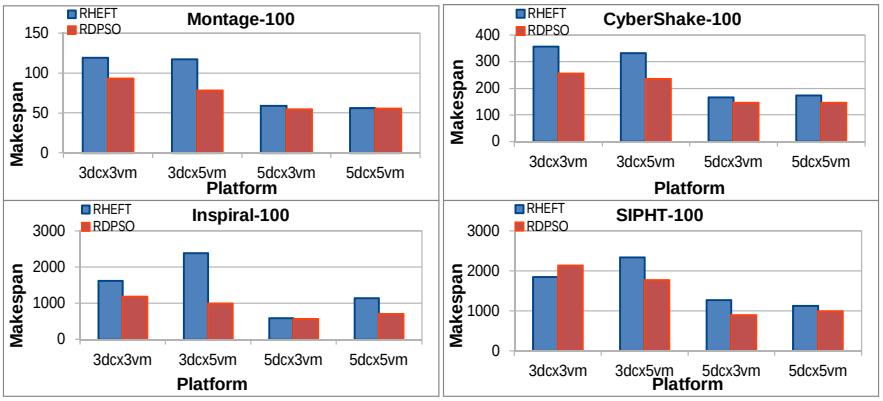
The overall reliability of workflows with 100 tasks [Figure 4(c)] does not fluctuate significantly through the different platforms for Montage and CyberShake workflows unlike Inspiral and SIPHT workflows. This is due to the fact that these latter are computing intensive and the value of the reliability is proportional to the computing time. For all the workflows, the reliability increased with RDPSO when compared to RHEFT mainly for Montage and Cybershake applications.

5.4.3 Details of transfers

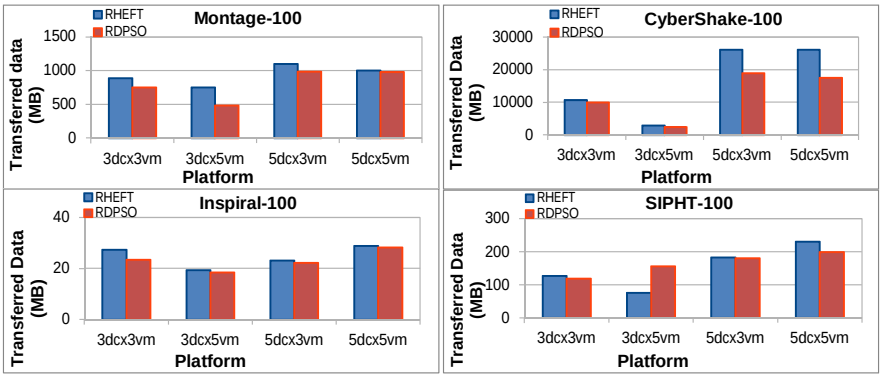
In this subsection, we analysed data transfers between different data centres within the same platform to show how the transferred data are optimised by RDPSO compared to RHEFT.

We chose to detail the transfers made during scheduling workflows having 100 tasks and 1000 tasks on the 5dcx3vm platform.

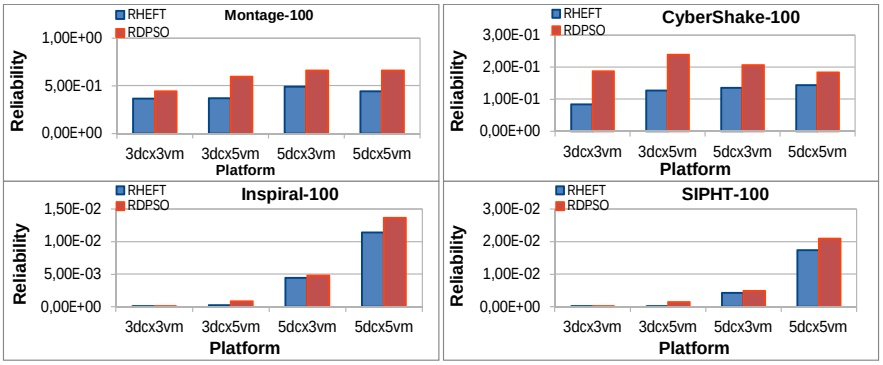
Figure 4 (a) Makespan, (b) transferred data and (c) reliability variations on different platforms for Montage-100, CyberShake-100, Inspiral-100 and SIPHT-100 workflows (see online version for colours)



(a) Makespan



(b) Transferred Data



(c) Reliability

Figure 5 Transfers details on 5dcx3vm for workflows having 100 tasks (see online version for colours)

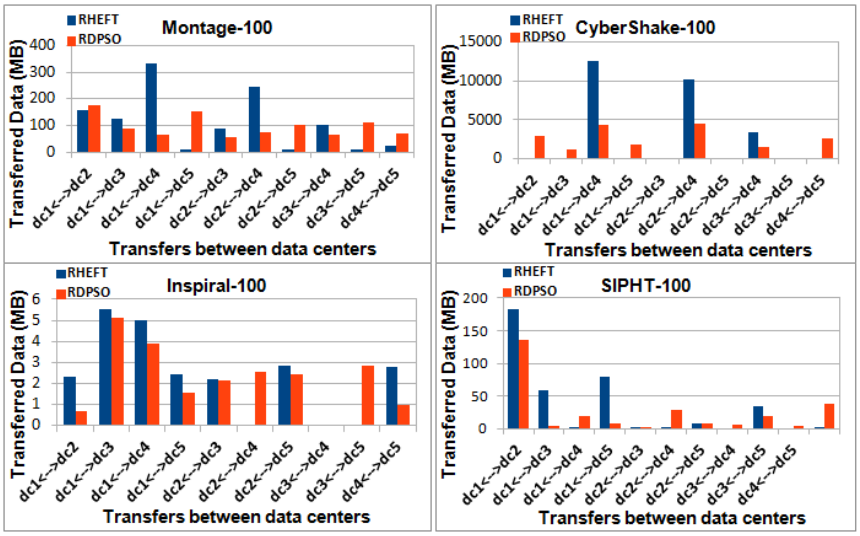


Figure 6 Transfers details on 5dcx3vm for workflows having 1,000 tasks (see online version for colours)

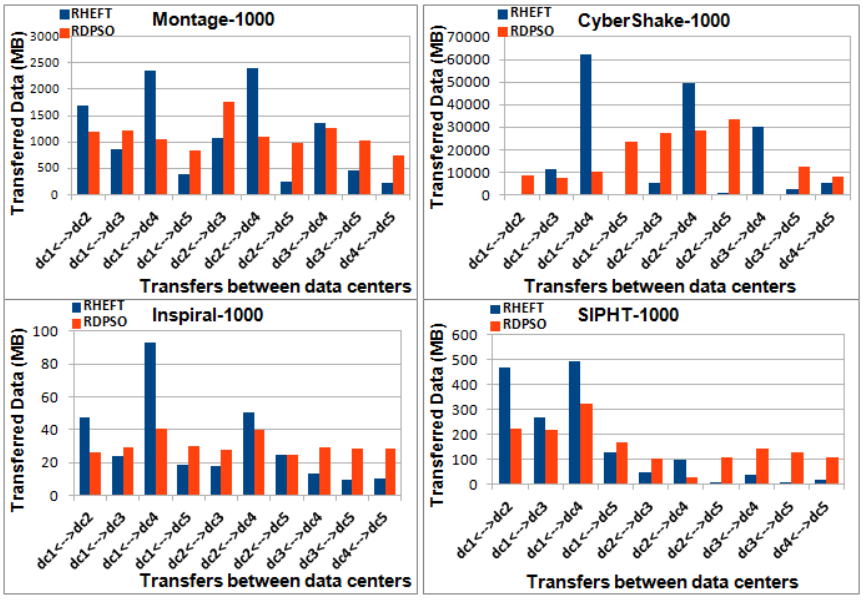


Figure 5 plots transfers details for workflows having 100 tasks between each pair of data centres in the 5dcx3vm platform. The notation ($dci \longleftrightarrow dcj$) means transfers between data centre i and data centre j. We discover that RDPSO avoids the large peaks of data transfers recorded by RHEFT carried out between $dc1$ and $dc4$ ($dc1 \longleftrightarrow dc4$), and between $dc2$ and $dc4$ ($dc2 \longleftrightarrow dc4$) for Montage and CyberShake workflows. In the

case of Inspiral and SIPHT workflows, data transfers achieved by RDPSO are closer to those performed by RHEFT.

Finally, our results are confirmed on a large scale concerning data transfers for extra large workflows using the Montage-1000, CyberShake-1000, Inspiral-1000 and SIPHT-1000 applications, and achieved on 5dcx3vm platform (Figure 6). The results for the Montage and CyberShake applications again show that RDPSO avoids the large peaks of transfers recorded by RHEFT carried out between $dc1$ and $dc4$ ($dc1 \longleftrightarrow dc4$), and between $dc2$ and $dc4$ ($dc2 \longleftrightarrow dc4$). Our approach avoids the significant transfers between $dc1$ and $dc4$ for Inspiral application and between $dc1$ and $dc2$ and between $dc1$ and $dc4$ for SIPHT application.

6 Conclusions and future work

The makespan still remains the most important metric to optimise when scheduling applications. Particularly, in multiple cloud data centres platforms, this metric is degraded due to data transfer times. In addition, most of the studies assume that links are homogeneous with uniform bandwidth and that there is no VM failure during the execution of a task or that failures of the network links are ignored. In our contribution, we presented the RDPSO, a scheduling approach based on PSO. We aimed to minimise the makespan and the data transfer times while enhancing the reliability's workflow on multiple cloud data centres. We also consider resources heterogeneity, VMs failures and links failures.

We evaluated our algorithm using real-world workflows and presented the results obtained by our heuristic in opposition to RHEFT. The results show that RDPSO outperforms the RHEFT algorithm. The makespan is improved by RDPSO in most cases and the overall reliability is enhanced. Our approach realised transfers with less amount of data than the RHEFT algorithm even using extra large workflows.

As future work, we plan to expand our proposed approach to investigate additional constraints such as cost and energy consumption in multiple cloud data centres. We would like to consider variable costs to access resources that change across multiple data centres. Moreover, according to the needs of users and providers, actual data centres must be low-energy systems. Thereby, we will extend our approach to operate in a bi-level (Sinha et al., 2018) by optimising resource allocation with corresponding resource availability and reliability and by reducing energy consumption for workflows scheduling. Finally, the resource model we proposed can be easily extended to federated clouds and the experimental results can be perfected by comparing our PSO-based algorithm with a GA-based approach.

References

Babu, L.D. and Krishna, P.V. (2013) 'Versatile time-cost algorithm (VTCA) for scheduling non-preemptive tasks of time critical workflows in cloud computing systems', *International Journal of Communication Networks and Distributed Systems*, Vol. 11, No. 4, pp.390–411.

Bouali, L., Oukfif, K., Bouzefrane, S. and Oulebsir-Boumghar, F. (2015) 'A hybrid algorithm for DAG application scheduling on computational grids', *International Conference on Mobile, Secure and Programmable Networking*, pp.63–77, Springer.

Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A. and Buyya, R. (2011) 'CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms', *Software: Practice and Experience*, Vol. 41, No. 1, pp.23–50.

Chen, J., Zhang, J. and Song, A. (2017) 'Efficient data and task co-scheduling for scientific workflow in geo-distributed datacenters', *2017 Fifth International Conference on Advanced Cloud and Big Data (CBD)*, pp.63–68, IEEE.

Chen, W-N. and Zhang, J. (2009) 'An ant colony optimization approach to a grid workflow scheduling problem with various QoS requirements', *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 39, No. 1, pp.29–43.

Chen, W-N. and Zhang, J. (2012) 'A set-based discrete pso for cloud workflow scheduling with user-defined QoS constraints', *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp.773–778, IEEE.

Choudhary, A., Gupta, I., Singh, V. and Jana, P.K. (2018) 'A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing', *Future Generation Computer Systems*, Vol. 83, No. C, pp.14–26.

Doğan, A. and Özgüner, F. (2005) 'Biobjective scheduling algorithms for execution time-reliability trade-off in heterogeneous computing systems', *The Computer Journal*, Vol. 48, No. 3, pp.300–314.

Dongarra, J.J., Jeannot, E., Saule, E. and Shi, Z. (2007) 'Bi-objective scheduling algorithms for optimizing makespan and reliability on heterogeneous systems', *Proceedings of the Nineteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, pp.280–288, ACM.

Fard, H.M., Prodan, R., Barrionuevo, J.J.D. and Fahringer, T. (2012) 'A multi-objective approach for workflow scheduling in heterogeneous environments', *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2012)*, pp.300–309, IEEE Computer Society.

Guo, T., Liu, J., Hu, W. and Wei, M. (2018) 'Energy-aware fault-tolerant scheduling under reliability and time constraints in heterogeneous systems', *International Conference on Intelligent Computing*, pp.36–46, Springer.

Hakem, M. and Butelle, F. (2007) 'Reliability and scheduling on systems subject to failures', *International Conference on Parallel Processing, 2007. ICPP 2007*, pp.38–38, IEEE.

Jian, C., Tao, M. and Wang, Y. (2014) 'A particle swarm optimisation algorithm for cloud-oriented workflow scheduling based on reliability', *International Journal of Computer Applications in Technology*, Vol. 50, Nos. 3–4, pp.220–225.

Juve, G., Chervenak, A., Deelman, E., Bharathi, S., Mehta, G. and Vahi, K. (2013) 'Characterizing and profiling scientific workflows', *Future Generation Computer Systems*, Vol. 29, No. 3, pp.682–692.

Kennedy, J. (2011) 'Particle swarm optimization', *Encyclopedia of Machine Learning*, pp.760–766, Springer, New York, NY, USA,.

Kianpisheh, S., Charkari, N.M. and Kargahi, M. (2016) 'Reliability-driven scheduling of time/cost-constrained grid workflows', *Future Generation Computer Systems*, Vol. 55, No. C, pp.1–16.

Kumar, H., Chauhan, N.K. and Yadav, P.K. (2018) 'A task allocation model for minimising system cost and maximising reliability of distributed computing system', *International Journal of Communication Networks and Distributed Systems*, Vol. 20, No. 2, pp.226–243.

Lee, Y.C., Zomaya, A.Y. and Yousif, M. (2010) 'Reliable workflow execution in distributed systems for cost efficiency', *2010 11th IEEE/ACM International Conference on Grid Computing (GRID)*, pp.89–96, IEEE.

Mell, P. and Grance, T. (2011) *The NIST Definition of Cloud Computing*, National Institute of Standards and Technology [online] http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf (accessed August 2016.).

Oukfif, K., Bouali, L., Bouzefrane, S. and Oulebsir-Boumghar, F. (2015) 'Energy-aware dpso algorithm for workflow scheduling on computational grids', *2015 3rd International Conference on Future Internet of Things and Cloud (FiCloud)*, pp.651–656, IEEE.

Poola, D., Ramamohanarao, K. and Buyya, R. (2016) 'Enhancing reliability of workflow execution using task replication and spot instances', *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, Vol. 10, No. 4, p.30.

Qin, X. and Jiang, H. (2006) 'A novel fault-tolerant scheduling algorithm for precedence constrained tasks in real-time heterogeneous systems', *Parallel Computing*, Vol. 32, No. 5, pp.331–356.

Rehani, N. and Garg, R. (2017) 'Meta-heuristic based reliable and green workflow scheduling in cloud computing', *International Journal of System Assurance Engineering and Management*, Vol. 9, No. 4, pp.1–10.

Sih, G.C. and Lee, E.A. (1993) 'A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures', *IEEE Transactions on Parallel and Distributed systems*, Vol. 4, No. 2, pp.175–187.

Sinha, A., Malo, P. and Deb, K. (2018) 'A review on bilevel optimization: from classical to evolutionary approaches and applications', *IEEE Transactions on Evolutionary Computation*, Vol. 22, No. 2, pp.276–295.

Tang, X., Li, K., Qiu, M. and Sha, E.H-M. (2012) 'A hierarchical reliability-driven scheduling algorithm in grid systems', *Journal of Parallel and Distributed Computing*, Vol. 72, No. 4, pp.525–535.

Topcuoglu, H., Hariri, S. and Wu, M-Y. (2002) 'Performance-effective and low-complexity task scheduling for heterogeneous computing', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 13, No. 3, pp.260–274.

Ullman, J.D. (1975) 'NP-complete scheduling problems', *Journal of Computer and System Sciences*, Vol. 10, No. 3, pp.384–393.

Wang, X., Yeo, C.S., Buyya, R. and Su, J. (2011) 'Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm', *Future Generation Computer Systems*, Vol. 27, No. 8, pp.1124–1134.

Wen, Z., Cala, J., Watson, P. and Romanovsky, A. (2016) 'Cost effective, reliable and secure workflow deployment over federated clouds', *IEEE Transactions on Services Computing*, Vol. 10, No. 6, pp.929–941.

Wu, F., Wu, Q. and Tan, Y. (2015) 'Workflow scheduling in cloud: a survey', *The Journal of Supercomputing*, Vol. 71, No. 9, pp.3373–3418.

Xiao, X., Xie, G., Xu, C., Fan, C., Li, R. and Li, K. (2018) 'Maximizing reliability of energy constrained parallel applications on heterogeneous distributed systems', *Journal of Computational Science*, Vol. 26, No. C, pp.344–353.

Zhang, J., Wang, M., Luo, J., Dong, F. and Zhang, J. (2015) 'Towards optimized scheduling for data-intensive scientific workflow in multiple datacenter environment', *Concurrency and Computation: Practice and Experience*, Vol. 27, No. 18, pp.5606–5622.

Zhang, L., Li, K., Li, C. and Li, K. (2017) 'Bi-objective workflow scheduling of the energy consumption and reliability in heterogeneous computing systems', *Information Sciences*, Vol. 379, pp.241–256.