A Robust Scheduler for Workflow Ensembles under Uncertainties of Available Bandwidth

Thiago A. L. Genez*, Luiz F. Bittencourt*, Rizos Sakellariou†, and Edmundo R. M. Madeira*
*Institute of Computing, University of Campinas (Unicamp), Campinas, São Paulo, Brazil
†School of Computer Science, University of Manchester, U.K.
E-mails:{thiagogenez, bit, edmundo}@ic.unicamp.br, rizos@manchester.ac.uk

***Abstract*—Imprecise input data imposes special challenges to workflow scheduling. This paper introduces a robust scheduler based on particle swarm optimisation, called RobWE, which considers uncertainties of available bandwidth when producing schedules for workflow ensembles. The proposed scheduler is also a flexible scheduler since it allows the replacement of its objective function according to the user's needs. The effectiveness of the proposed RobWE scheduler is compared to a non-robust scheduler that does not consider the presence of such uncertainties. Results of simulations considering diverse scenarios, based on several degrees of uncertainty in available bandwidth estimates,characteristics of bandwidth estimations and workflow applications, demonstrate the advantages of the proposed RobWE scheduler.**

***Index Terms*—Workflow Ensembles; Scheduling; Uncertainty;**

I. Introduction

Resources in hybrid clouds, consisting of private and public resources, are normally connected through Internet links whose available bandwidth fluctuates considerably because of the shared communication infrastructure [1]. The use of imprecise information to estimate inter-cloud available bandwidth may cause a scheduler to generate schedules that result in slow execution time and/or poor application performance [2]. Actually, one of the main issues faced in scheduling problems is the quality of the information that is provided as input data to the scheduler to produce a schedule [3]. Uncertainties of available bandwidth can be a result of imprecise measurement and monitoring network tools and/or their incapacity of estimating the real value of the available bandwidth during the application scheduling time [4]. In general, bandwidth estimations will be given in ranges rather than deterministic values [5]. Thus, schedules produced by deterministic schedulers may make decisions that lead to ineffective executions of applications. Among the different type of applications, the focus of this paper is on scientific workflows [6], particularly groups of inter-related workflows, known as *workflow ensembles* [2].

This paper introduces a novel robust scheduler, called RobWE, for workflow ensembles in hybrid clouds. The scheduler is based on particle swarm optimization (PSO) for the consideration of the uncertainties of available bandwidth. The RobWE scheduler is a revised version of the PSO-based scheduler, presented in [2], which does *not* consider the uncertainty introduced using monitoring/measuring tools. As the original PSO-based scheduler neglects such uncertainties, it usually produces poor schedules when the measured bandwidth estimates provided by these tools at scheduling time do not correspond to the real bandwidth experienced at the application execution time. To improve the RobWE scheduler and generate robust schedules under imprecise bandwidth estimations, we employ an adapted version of the mechanism presented in [7]. This mechanism deflates the bandwidth estimates given by the monitoring tools before furnishing them to the scheduler in an attempt to minimize the negative impact of uncertain bandwidth estimates.

The contribution of this paper is a scheduler for workflow ensembles that is able to make scheduling decisions which should be robust in the face of uncertainties stemming from using monitoring/measurement networking tools. The effectiveness of the RobWE scheduler is compared to that of the original PSO-based scheduler which does not consider the uncertainties of available bandwidth. As both schedulers are flexible, i.e., they allow the replacement of its objective function according to the user's needs, three objective functions were evaluated: minimization of the overall makespan, maximization of fairness, and minimization of costs. For all three objective functions evaluated, the results have indicated that the RobWE scheduler undergoes less degradation in its performance than the original PSO-based scheduler as the degree of uncertainty in available bandwidth increases. The present paper focuses on the robustness of the RobWE scheduler for different degrees of uncertainties in available bandwidth. To the best of our knowledge, there is no other proposal in the literature that simultaneously takes into account the scheduling of a set of workflows concurrently (workflow ensembles), the uncertainties in available bandwidth, and different scheduler objective functions.

This paper is organised as follows. Section II presents related work, while Section III reviews the key concepts of this paper. The proposed RobWE scheduler is introduced in Section IV and evaluated in Section V. Results are discussed in Section VI, while final remarks and conclusion are presented in Section VII.

II. Related Work

Imprecision in input information to the scheduler has been approached mostly via reactive techniques, such as dynamic scheduling [8], rescheduling [9], and self-adjusting [10]. Reactive techniques are best suited in situations when the resource availability fluctuates considerably, such as the available bandwidth in inter-cloud links. As reactive mechanisms are based on monitoring of resource utilisation, imprecise information may also be due to the intrusion effect of monitoring tools [4]. In order to avoid the drawbacks of reactive approaches, Batista and Fonseca presented in [5] a proactive approach to deal with the uncertainties in application demand and resource availability when scheduling workflows in grids. They proposed a scheduler based on fuzzy optimization which models the uncertainties of application demands and resource availability as fuzzy numbers. They treat the scheduling problem as an integer linear-programming (ILP) problem since the fuzzy optimisation problem is transformed into an equivalent ILP. In contrast to our proposal, the fuzzy-based scheduler does not handle the scheduling of a set of workflows concurrently on the same set of resources. Also, since the fuzzy-based scheduler only minimizes the makespan of the application, other objective functions cannot be employed.

A proactive mechanism [7], previously introduced by the authors of the present paper, minimizes the negative impact of using uncertain bandwidth estimates in the production of schedules. To reduce the imprecision of bandwidth estimates given by monitoring/measuring tools, the mechanism refines such estimates and furnishes amended values to the scheduler. As an initial evaluation, the effectiveness of the mechanism was only assessed with schedulers from the literature that deal with only one workflow at a time and simply minimize the execution cost while meeting specified deadlines. Results showed that the mechanism increased the effectiveness of such schedulers under such uncertainties since they started to produce schedules with less inaccurate cost estimates, besides increasing the number of solutions within the deadline. Motivated by results found in that previous paper, the scheduler introduced here (RobWE) extends the robustness of our scheduler for workflow ensembles with the employment of an adapted version of this mechanism. In contrast to related work, this paper proposes a scheduler for workflow ensembles that is able to make scheduling decisions which should be robust in the face of the uncertainties as a result of using bandwidth estimates given by unreliable monitoring tools.

III. Background and Key Concepts

A. Workflow Representation

In this paper, a workflow application is represented by a directed acyclic graph (DAG) in which nodes denote computational tasks and edges denote data dependencies between tasks. A scientific application that comprises a set of inter-related workflows is called a *workflow ensemble* [6], [11], [12]. Workflows in an ensemble typically have a similar structure, distinguished only by their sizes (number of tasks), input data, and individual computational task sizes.

B. Hybrid Cloud

A hybrid cloud is composed of resources from a private cloud, assumed free of charge, as well as resources, i.e., virtual machines (VMs), from one or more public cloud providers on a pay-per-use basis. When the application demands require more resources than locally available, the scheduler must decide which VMs should be leased from public cloud providers to compose the hybrid cloud. In this paper, we assume that the organization has a broker in its premises which is responsible for connecting resources from the public clouds to those of the private cloud resources. This broker receives requests for workflow ensemble execution and rents resources from the public clouds based on the decisions of the scheduler.

C. A Flexible Scheduler for Workflow Ensembles based on Particle Swarm Optimization

In this section we describe the flexible workflow scheduler based on particle swarm optimization (PSO) presented in [2]. PSO is composed of a population (called a swarm) of candidate schedule solutions (called particles), which are independently evolved at each iteration of the algorithm. Iteratively, PSO uses a fitness function to select the best solution within the swarm. This scheduler is accomplished by modelling the workflow scheduling problem as a PSO and developing a fitness function that acts as the scheduler's objective function. Due to the possibility to create as many fitness functions as desired, this scheduler is considered *flexible* since it allows the replacement of the objective function according to the user's needs, making the scheduler usable in various cloud scenarios.

1) ***Particle as a Candidate Schedule Solution:*** Let $\mathcal{E} = \{\mathcal{G}_1, \ldots, \mathcal{G}_n\}$ be a DAG (workflow) ensemble and $\mathcal{R}$ be a set of resources. A particle is represented by an array of length k, where k is the total number of tasks contained in the ensemble $\mathcal{E}$. The value assigned to each particle's dimension (array's position) is an integer that represents the index of a computing resource in $\mathcal{R}$. Therefore, each particle of the swarm represents a schedule that allocates $\mathcal{E}$ to $\mathcal{R}$.

2) ***Policies:*** The DAGs represented by the particle are scheduled independently by the scheduler on a "first come, first served" basis. We assume that each DAG in the ensemble is given a numeric priority which indicates how important the DAG to the user is. The highest priority DAG is the first one to be scheduled. Regarding the node ordering within a DAG, this follows the topological order given by the depth-first search algorithm. The scheduler uses a gap search or backfilling technique to schedule tasks on resources [13]. This method takes advantage of the gaps left on resources between scheduled tasks as a result of communication dependencies.

3) ***Fitness Function as a Scheduler's Objective Function:*** Let $f : \mathbb{N}^k \rightarrow \mathbb{R}$ be the fitness function that must be minimized. This function takes a k-dimensional particle as an argument in the form of an array of integers, and produces a real number as output, which indicates the particle's fitness value. The aim of PSO is to find, at each iteration, a particle ρ such that $f(\rho) \leq f(q)$ for all particles q in the swarm. The maximization can be performed by considering the function $f' = -f$ instead. Importantly, the particle only informs on which resource of $\mathcal{R}$ each task of $\mathcal{E}$ is scheduled to be run. Complying with the scheduler's policies, it is the role of the fitness function in calculating the timing of scheduling events.

To show the scheduler's flexibility, three different objective functions were employed: minimization of overall makespan, minimization of costs, and maximization of fairness. For each objective function, one fitness function was developed.

a) Makespan-aware scheduler: To comply with the minimization of the schedule makespan (application execution time), the fitness function f is employed. This function produces as output numbers associated with the overall makespan of a given particle (schedule) ρ. By using this function, the PSO is setup to minimize the resolution of the problem, since its goal is to find particles that represent schedules with the lowest overall makespan. The fitness function f is defined as follows:

$$f(\rho) = \mu(\rho) \tag{1}$$

where $\mu(\rho)$ is the overall makespan of the schedule given by the k-dimensional particle ρ, that is, the start time of the first task of the first DAG in the ensemble $\mathcal{E}$ to the finish time of the last task of the last DAG in the ensemble $\mathcal{E}$.

b) Cost-aware scheduler: To minimize costs, we employ the fitness function g. This function is defined on the basis of the monetary execution cost of a schedule represented by a particle ρ. PSO minimizes the resolution of the problem in this case, since its goal is to achieve particles that represent low-cost schedules. The function g is defined as follows:

$$g(\rho) = \mathcal{C}(\rho), \tag{2}$$

where $\mathcal{C}(\rho)$ is the overall monetary execution cost of the schedule given by ρ, which is defined as follows:

$$\mathcal{C}(\rho) = \sum_{\forall\, r \,\in\, \mathcal{R}} \left(c_r \cdot \lceil t_{r,\rho} \rceil \right) \tag{3}$$

where c_r is the cost per time unit for using the resource $r \in \mathcal{R}$, and $t_{r,\rho}$ is the total time units of usage of r in the schedule given by ρ, where partial units are rounded up.

c) Fairness-aware scheduler: When scheduling several DAGs simultaneously on the same set of resources, each DAG may experience a *slowdown* in its execution, when compared to the runtime of the same DAG if it had the resources on its own. A schedule is considered fair when each DAG of the ensemble experiences equal (or similar) slowdown in its execution. The slowdown value of a DAG $i \in \mathcal{E}$, $\sigma(i)$, in the schedule given by the particle ρ is defined as in [14]: $\sigma_i = {\mu_i^{own}(\rho)}/{\mu_i(\rho)}$, where μ_i^{own} is the makespan of a DAG i if it had the available resources on its own, and μ_i is the makespan of a DAG i when sharing resources along with all other DAGs. It is expected that slowdown values are in the range $[0, 1]$, with values closer to 1 indicating a small slowdown.

To maximize fairness, the fitness function h is employed. This function is defined as the ratio between the lowest (minimum) slowdown and the peak (maximum) slowdown experienced by the DAGs in the schedule. PSO is configured to maximize the resolution of the problem since its goal is to find particles that represent high $h(\rho)$ values, i.e., fairness schedules. The function h is defined as:

$$h(\rho) = \frac{\min\limits_{\forall\, i \,\in\, \mathcal{E}} (\sigma_i)}{\max\limits_{\forall\, i \,\in\, \mathcal{E}} (\sigma_i)} \tag{4}$$

IV. Robustness under Uncertainties of the inter-cloud available bandwidth values

The PSO-based workflow scheduler [2] is not designed to address inaccurate information about the available bandwidth. As this scheduler does not deal with imprecise estimates introduced by the use of monitoring/measuring tools [4], poor schedules are expected to be produced when the measured bandwidth estimates used at scheduling time do not correspond to the actual ones experienced at the execution time. In an attempt to drive the RobWE scheduler in producing robust schedules in the face of such uncertainty, we employ a revised version of the mechanism in [7].

A. Mechanism to handle the Uncertainty of Available Inter-cloud Bandwidth values in the Scheduling Production

The estimate of the available bandwidth may deviate from that measured by an expected percent error p at the execution time of the application [4]. For example, in a scenario in which the expected uncertainty of the available bandwidth is $p = 50\%$, a data transfer that is initially estimated by the scheduler to take 30 seconds, it could take between 15 and 45 seconds during the execution of the workflow. To quantify such uncertainties, the mechanism employs the quality of information model presented in [3]. Let b be the estimated value of the available bandwidth and p be the expected uncertainty of b. According to this model, during the execution of the workflow, the estimate of the inter-cloud available bandwidth may experience a variation from $b - p\%$ to $b + p\%$. Most schedulers in the literature neglect the presence of such variability p, and as a result, the produced schedules may have their performance degraded as a result of large values of p at application runtime.

As a proactive approach to minimize the negative impact of using imprecise information about the available bandwidth, the mechanism in [7] produces a deflating multiplier value that is applied to the measured available bandwidth before being furnished to the scheduler. According to an expected uncertainty p, the mechanism applies a reduction factor $\mathcal{U}$ to the measured inter-cloud bandwidth to prevent inaccurate information from affecting the performance of the execution of the workflow. Building a schedule which takes into account the uncertainty of available bandwidth is expected to give robust schedules. Importantly, this mechanism does not change the scheduling algorithm or the scheduler itself, it is simply a procedure that assists the scheduler in producing robust schedules under imprecise bandwidth estimations.

B. Computation of the Deflating $\mathcal{U}$ Value

Let $\mathcal{H}_{\mathcal{E},n}$ be a data set of previous executions of a specific workflow ensemble type $\mathcal{E}$, where each ensemble is composed

Algorithm 1 Computation of the coefficients a_1, a_2, and a_3

```
1: $\mathcal{H}_{\mathcal{E},n}$ = Database of ensembles $\mathcal{E}$ composed of $n$ inter-related DAGs
2: $f'$ = Fitness function employed in the scheduler $\{f, g, \text{or } h\}$
3: $\mathcal{H}^{f'}_{\mathcal{E},n} = \emptyset$ {Subset for the fitness function $f'$}
4: for all $(b', p')$ pair existing in $\mathcal{H}_{\mathcal{E},n}$ do
5:     $ST = None$ {the selected 7-tuple according to $b'$, $p'$, and $f'$ values}
6:     for all 7-tuples $\langle func, b, p, \mathcal{U}, m, c, j \rangle \in \mathcal{H}_{\mathcal{E},n}$ do
7:         if $f' == f$ then
8:             $ST =$ Select the 7-tuple that $b = b'$, $p = p'$ and $m$ is minimal
9:         else if $f' == g$ then
10:            $ST =$ Select the 7-tuple that $b = b'$, $p = p'$ and $c$ is minimal
11:        else if $f' == h$ then
12:            $ST =$ Select the 7-tuple that $b = b'$, $p = p'$ and $j$ is maximal
13:        end if
14:    end for
15:    $\mathcal{U}'$ = Take the $\mathcal{U}$ value from the selected 7-tuple represented by $ST$
16:    $\mathcal{H}^{f'}_{\mathcal{E},n} = \mathcal{H}^{f'}_{\mathcal{E},n} \cup \langle b', p', \mathcal{U}' \rangle$
17: end for
18: Call multiple linear regression by using the $\mathcal{H}^{f'}_{\mathcal{E},n}$ as an input data set
    and the function $f(x, y) = a_1x + a_2y + a_3$ as a formula to fit the data
19: return the the coefficient values $a_1$, $a_2$, and $a_3$
```

of n inter-related DAGs. Each row of the data set $\mathcal{H}_{\mathcal{E},n}$ is represented by 7-tuples $\langle func, b, \mathcal{U}, p, m, c, j \rangle$, where the first three elements are information used to produce the ensemble's schedule, while the others are obtained from the ensemble's execution. The elements $func$, b and $\mathcal{U}$ are, respectively, the fitness function employed into the RobWE scheduler to produce the schedules, the measured available bandwidth at scheduling time, and the calculated $\mathcal{U}$ used to deflate the respective b value. The fourth element, p, represents the maximum percentage error between b and the observed available bandwidth during the execution of the ensemble $\mathcal{E}$. The observed overall makespan (execution time) of the ensemble is represented by the fifth element m, while the actual (real) cost of this execution is included as the sixth element c. The seventh and last element, j, corresponds to the Jain's fairness index of the ensemble execution, an index developed for resource sharing problems [13]. Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of numbers, the Jain's fairness index of $\mathcal{X}$ is defined as: $\left(\sum_{i=1}^{n} x_i\right)^2 / n\left(\sum_{i=1}^{n} x_i^2\right)$, where the index ranges between $1/n$ (worst case, unfair) and 1 (best case, fair).

The computation of the deflating $\mathcal{U}$-value involves the construction of a specific data subset $\mathcal{H}^{func}_{\mathcal{E},n} \subseteq \mathcal{H}_{\mathcal{E},n}$ for each fitness function $func$ in $\mathcal{H}_{\mathcal{E},n}$. This subset is composed of 3-tuples $\langle b, p, \mathcal{U} \rangle$ and is constructed as follows: for each pair (b, p) in $\mathcal{H}_{\mathcal{E},n}$, one 7-tuple from $\mathcal{H}_{\mathcal{E},n}$ is selected and its $\mathcal{U}$ value is taken to compose the 3-tuple $\langle b, p, \mathcal{U} \rangle$, which is stored in $\mathcal{H}^{func}_{\mathcal{E},n}$. The selection of this 7-tuple, however, is carried out according to the fitness function $func$ in $\mathcal{H}^{func}_{\mathcal{E},n}$:

- If $func$ is the fitness function f, the 7-tuple selected is the tuple that contains the smallest m value (the fastest execution), besides the b and p values
- If $func$ is the fitness function g, the 7-tuple selected is the tuple that contains the smallest c value (the cheapest execution), besides the b and p values
- If $func$ is the fitness function h, the 7-tuple selected is the tuple that contains the highest j value (the fairness execution), besides the b and p values

Each 3-tuple $\langle b, p, \mathcal{U} \rangle$ of the subset $\mathcal{H}^{func}_{\mathcal{E},n}$ means that, when the bandwidth estimate b had an expected uncertainty of up to p and the objective function used to produce the schedules was $func$, the scheduler was able to produce better results when the $\mathcal{U}$ value was used to deflate b. The calculation of the $\mathcal{U}$ value is carried out by a multiple linear regression (MLR) procedure, which uses the subset $\mathcal{H}^{func}_{\mathcal{E},n}$ as input data. The MLR uses the subset to compute the values of the coefficients a_1, a_2 and a_3 in the equation $f(x, y) = a_1x + a_2y + a_3$. In this equation, x is a variable that represents the currently predicted uncertainty in the available bandwidth, y is an independent variable that represents the currently available bandwidth, and $\mathcal{U} = f(x, y)$. The subset construction and the computation of the coefficients a_1, a_2, and a_3 are illustrated in Algorithm 1.

C. Applying the Mechanism

When an ensemble $\mathcal{E}$ is about to be scheduled, the $\mathcal{U}$ value is computed through the p and b values obtained from the monitoring tools [4]. Making $x = p$, the currently predicted uncertainty, and $y = b$ the currently estimated available bandwidth, a percentage reduction factor $\mathcal{U} = f(x, y)$ is computed for the next ensemble schedule. The use of the mechanism requires a set of historical data for training before the multiple linear regression approach can be applied.

V. Evaluation Methodology

A. Workflow Ensembles

Ensembles using workflows available from the workflow generator gallery[1] were used. The gallery is composed of synthetic workflows modelled using patterns taken from real workflow applications [6] such as: LIGO, Montage, and SIPHT. As an initial evaluation, we considered only workflows with 50 tasks (DAG nodes). For each type of application, 100 different workflows were generated differing in the weights of the DAG nodes (computational demands) and DAG edges (communication demands). Using this collection of synthetic workflows, we constructed 500 ensembles for each workflow type, where each ensemble has 10 workflows.

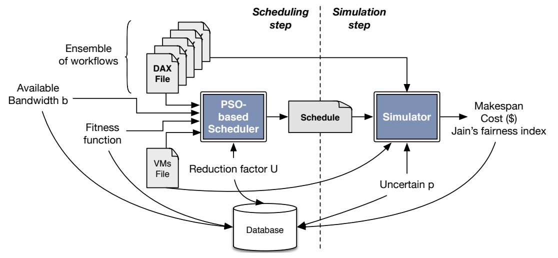
TABLE I
Hybrid cloud scenario

Provider	Type	Cores	Performance per core	Price per unit of time	Number of VMs
A (public)	Small	2	2.0	\$ 0.026	3
	Large	4	4.0	\$ 0.120	1
B (public)	Large	2	4.0	\$ 0.180	2
	X-Large	4	8.0	\$ 0.300	2
C (private)	Small	2	1.0	\$ 0.000	1

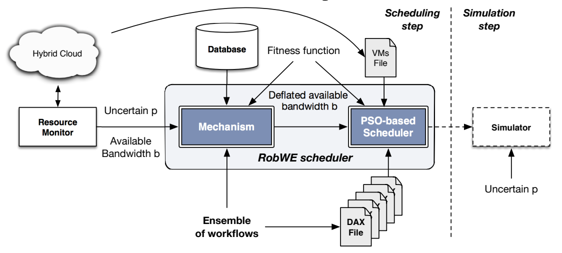
B. Hybrid Cloud Scenario

The hybrid cloud scenario to execute the workflow ensembles is shown in Table I, resulting in a total of 9 heterogeneous virtual machines. Inter-cloud bandwidths from 10 to 80 Mbps (in steps of 10 Mbps) and intra-cloud bandwidths of 100 Mbps were considered. We assume that the intra-cloud bandwidth is greater than the inter-cloud bandwidth, which is a reasonable assumption in real hybrid cloud environments. Lastly, we

[1]https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator



(a) First step



(b) Second and third steps

Fig. 1. Overview of the steps of the experimental evaluation.

assume that the virtual machines (VMs) are exclusively used for the workflow ensemble execution.

C. Experimental PSO Parameters

We used the JSwarm-PSO[2] to conduct the evaluation of the proposed scheduler. The number of particles was set to 500 and the number of iterations was set to 50. The particle's inertia ω was set to 1.5, and it decreases by 1% at each iteration. The particle's acceleration values a_1 and a_2 were set to 0.95, and both also decrease by 1% at each iteration.

D. Experimental Simulator

The data inputs for the proposed scheduler are: DAX files (ensemble), a VMs file, and a fitness function (scheduler's objetive function). Each DAX file contains the description of the workflow in XML format, while the VMs file contains information about the hybrid resources from Table I. After the schedule is produced by the scheduler, it is used as an input to the simulator, which simulates the schedule and calculates the metric values, such as: the overall makespan (ensemble execution time), the slowdown values, the monetary execution costs. However, at simulation time, all tasks of all workflows are re-ranked according to the execution finish time estimated by the produced schedule. The estimated inter-cloud available bandwidth value b is deflated by a factor $\mathcal{U}$, and the deflated value is used by the scheduler to produce the schedules; $\mathcal{U} = 0$ means that no reduction is applied, while $\mathcal{U} = 5$ means that the deflated bandwidth value provided to the scheduler is 95% of b. Importantly, at simulation time, no deflation is applied and the simulator uses a bandwidth estimate taken from the uniformly distributed interval $[b - p\% \,; b + p\%]$.

E. Experimental Setup

The proposed RobWE scheduler is evaluated in three steps. First (Fig. 1a), when the database is created, we assessed, without using the mechanism, how fixed $\mathcal{U}$ factors ameliorate the negative impact of imprecise information about the inter-cloud available bandwidth on the execution of the workflow ensemble. The following fixed bandwidth deflating factors $\mathcal{U} \in \{0, 5, 15, 25, 35, 45, 50\}$ were used according to [7]. For each value of $\mathcal{U}$, the percent error p ranged from 0% to 99%, in steps of 10%. Given a $\mathcal{U}$ value, a percentage error p is introduced by the simulator to derive the makespan, costs, and slowdowns of the ensemble execution, and all values are stored into the database. In this step, each application type is simulated using 500 different ensembles of 10 inter-related workflows each, 3 different fitness functions, 7 fixed $\mathcal{U}$-value, 8 bandwidth values, and 11 different values of p. The database for each application type is produced from an extensive set of $924,000$ simulations.

Secondly, the values of the the coefficients a_1, a_2 and a_3 in the equation $f(x, y) = a_1x + a_2y + a_3$ are computed using Algorithm 1. Lastly (Fig. 1b), to evaluate the effectiveness of the RobWE scheduler, new simulations were executed for each application type and fitness function by using its respective $\mathcal{U} = f(x, y)$. A total of 132,000 simulations were carried out in this step for each application type (500 different ensembles $\times$ 3 different fitness functions $\times$ 8 bandwidth values, and 11 p values).

VI. Results

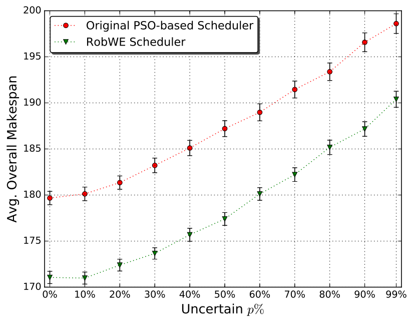
In this section, the effectiveness of the proposed RobWE scheduler is assessed and compared via simulation to that of its deterministic counterpart (the original PSO-based scheduler [2]). This paper focuses on the robustness of the RobWE scheduler for different degrees of uncertainties in available bandwidth estimates when scheduling workflow ensembles using several objective functions. The graphics shown are plotted over 500 simulations with a confidence interval of 95%.

A. RobWE as a Robust Makespan-aware Scheduler

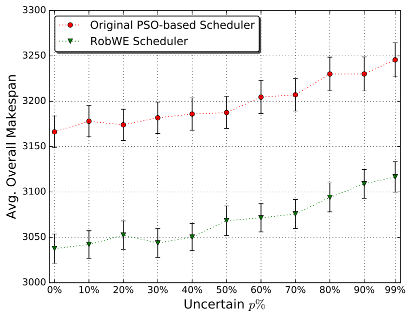
Figures 2a, 2b, and 2c show the average overall makespan for the Montage, LIGO, and SIPHT applications, respectively, as a function of the degree of experienced uncertainty in the available bandwidth estimates. Each graph shows curves corresponding to the results given by the original PSO-based scheduler and the proposed RobWE scheduler. To comply with the objective of minimizing the overall makespan, we employ the fitness function f to drive PSO in both schedulers.

It can be seen in Figure 2 that, for all applications, the RobWE scheduler was able to produce a lower overall makespan than the original PSO-based scheduler regardless of the degree of uncertainty p. For example, when a measured available bandwidth of 30 Mbps was deflated by the $\mathcal{U}$ value with an uncertainty degree of 50%, the average values of the overall makespan produced by the proposed RobWE scheduler for the Montage, LIGO, and SIPHT applications were approximately 7%, 6%, and 10% lower than the values produced by the original PSO-based scheduler, respectively. Even in scenarios with high uncertainties, a remarkable difference between

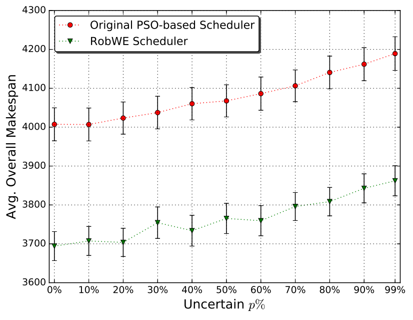
[2]http://jswarm-pso.sourceforge.net/



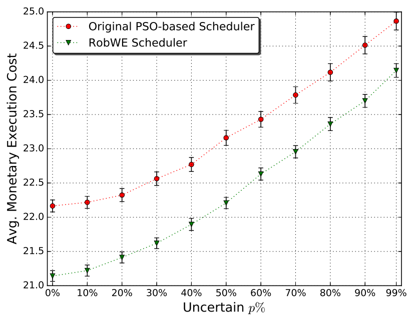
(a) Montage Application



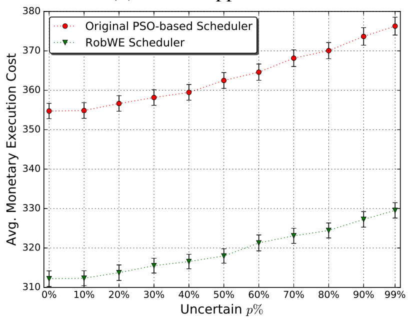
(b) LIGO Application



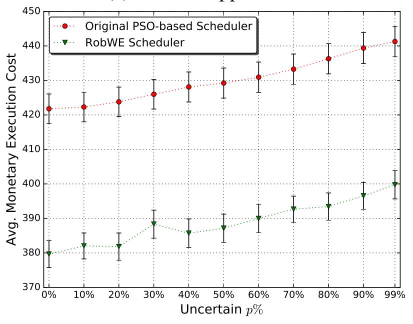
(c) SIPHT Application



(d) Montage Application



(e) LIGO Application



(f) SIPHT Application

Fig. 2. Average values of overall makespan (top) and execution cost (bottom) for 10-workflow ensembles for Montage, LIGO, and SIPHT applications when the measured available bandwidth in inter-cloud links was of 30 Mbps. These results were generated using the fitness function f in both schedulers. For the proposed RobWE scheduler only, the measured available bandwidth of 30 Mbps was deflated by the mechanism and provided as input to the scheduler. The x-axis represents the estimated error p in the measured available bandwidth which varies from 0% to 99%.

both schedulers was found for all applications. For example, for the SIPHT application, the RobWE scheduler reveals an average overall makespan 10% lower than those furnished by the original PSO-based scheduler when the uncertainty $p = 99\%$.

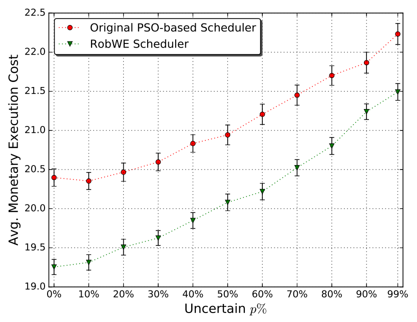
These results give some evidence of the robustness of the RobWE scheduler and its ability to achieve a lower makespan under uncertainties of the available bandwidth, since its performance underwent less degradation in makespan than the original PSO-based scheduler as the degree of uncertainty increases. As evidence that the RobWE scheduler did not result in more expensive execution, Figures 2d, 2e, and 2f show the average execution costs of the evaluation displayed in Figures 2a, 2b, and 2c, respectively. These figures also show that the performance of the proposed scheduler underwent less degradation in execution cost than that of the original PSO-based scheduler. For example, when $p = 50\%$, the RobWE scheduler was able to achieve schedules with reduced overall makespan for the Montage, LIGO, and SIPHT applications; it was also able to reduce costs in 5%, 13%, and 11%, respectively, from those produced by the original PSO-based scheduler. Indeed, the performance of the original scheduler degrades more than the RobWE scheduler as the degree of uncertainty increases.

B. RobWE as a Robust Cost-aware Scheduler

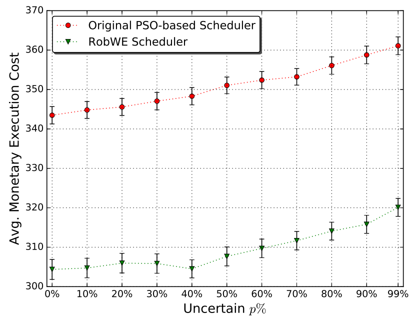
To comply with the objective function of minimizing the execution cost, we employ the fitness function g in the evaluation. Figures 3a, 3b, and 3c show the average monetary execution costs of the evaluation results for the Montage, LIGO, and SIPHT applications, respectively, as a function of the degree of uncertainty experienced in available bandwidth estimates. It is expected that the fitness function h would drive the PSO algorithm in producing cheaper schedules than the previous fitness function f, and indeed, for all applications, this expectation has been achieved for both schedulers, as we can compare with the average costs displayed in Figures 2d, 2e, and 2f. Since the aim of this paper is to discuss the negative consequences of imprecise information about the available bandwidth in the production of schedules and focus on how the RobWE scheduler can reduce such consequences, we do not go further into this analysis. For more details about the differences among fitness functions, please, refer to [2].

When the fitness function g is employed to minimize costs, the RobWE scheduler tends to produce cheaper schedules than the original PSO-based scheduler regardless of the degree of uncertainty p for all applications. For instance, for an uncertainty degree of 30%, the average execution cost produced by the RobWE scheduler for the Montage, LIGO, and SIPHT applications was approximately 6%, 14%, and 10% lower than that produced by the original PSO-based scheduler, respectively. A remarkable difference between both schedulers is also observed in scenarios with high uncertainties. For instance, when $p = 99\%$, the results achieved by the RobWE scheduler were5%, 12%, and 6% cheaper than the results produced by the original PSO-based scheduler.

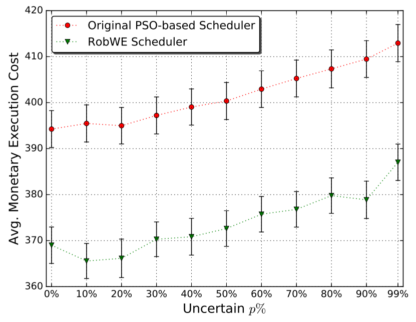
Similar to the previous analyses, the results presented in



(a) Montage Application

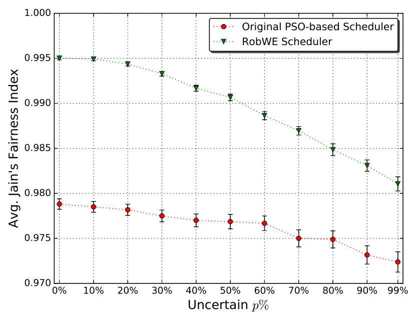


(b) LIGO Application

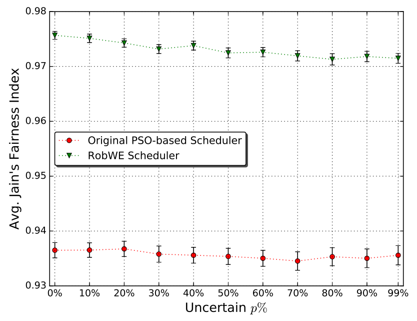


(c) SIPHT Application

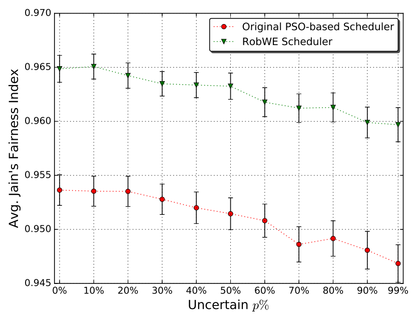
Fig. 3. Average values of execution costs for 10-workflow ensembles for the Montage, LIGO, and SIPHT applications when the measured available bandwidth in inter-cloud links was of 30 Mbps. These results were generated using the fitness function g in both schedulers. For the proposed RobWE scheduler only, the measured available bandwidth of 30 Mbps was deflated by the mechanism and provided as input to the scheduler. The x-axis represents the estimated error p in the measured available bandwidth which varies from 0% to 99%.



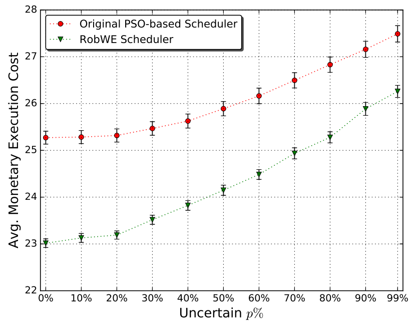
(a) Montage Application



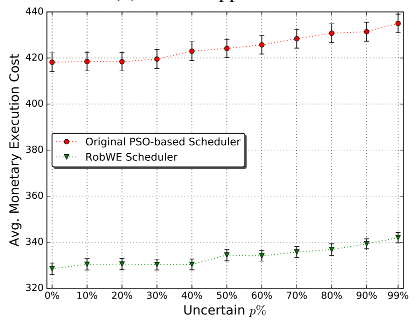
(b) LIGO Application



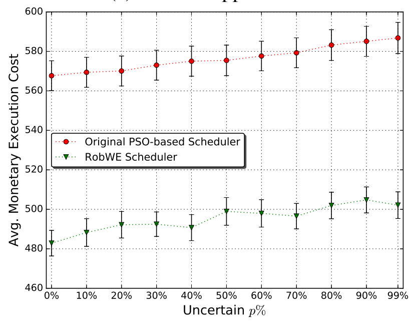
(c) SIPHT Application



(d) Montage Application



(e) LIGO Application



(f) SIPHT Application

Fig. 4. Average values of Jain's fairness index (top) and execution costs (bottom) for 10-workflow ensembles for the Montage, LIGO, and SIPHT applications when the measured available bandwidth in inter-cloud links was 30 Mbps. These results were generated using the fitness function h in both schedulers in an attempt to maximize fairness. For the proposed RobWE scheduler only, the measured available bandwidth of 30 Mbps was deflated by the mechanism and provided as input to the scheduler. The x-axis represents the estimated error p in the measured available bandwidth which varies from 0% to 99%.

this section also reinforce the argument about robustness of the RobWE scheduler in achieving cheaper executions in the presence of uncertainties in the available bandwidth, since its performance also underwent less degradation in cost than the performance of the original PSO-based scheduler as the degree of uncertainty increases. The trend observed in Figure 2, which can also be observed in Figure 3, shows a significant degradation in performance for the original PSO-based scheduler than the performance of the RobWE scheduler.

C. RobWE as a Robust Fairness-aware Scheduler

This section presents the results obtained when using the fitness function h in an attempt to maximize the fairness of the schedules. Figures 4a, 4b, and 4c show the average Jain's fairness index of the evaluation results for the Montage, LIGO, and SIPHT applications, respectively, as a function of the degree of uncertainty experienced in bandwidth estimates. For both schedulers, the function h is able to drive PSO in achieving fairer schedules for all applications, since average values of Jain's fairness index closer to 1 indicate that fair schedules were produced.

Similar to previous results, the performance of the RobWE scheduler had a smaller degradation in fairness than that of the original PSO-based scheduler. For instance, for an uncertainty degree of 20%, the average values of Jain's index produced by the RobWE scheduler for the Montage, LIGO, and SIPHT applications were approximately 3%, 5%, and 2% higher than the values produced by the original PSO-based scheduler, respectively. Although the evidence of smaller fairness degradation by the RobWE scheduler is modestly noticeable, the cost savings for these schedules are remarkably significant, as shown in Figures 4d, 4e, and 4f. For example, for $p = 20\%$, the results achieved by the RobWE scheduler were approximately 9%, 21%, and 16% cheaper than the results produced by the original PSO-based scheduler.

A performance difference between both schedulers is also noticeable in cases with high uncertainties, wherein the RobWE scheduler underwent less degradation in its performance than the original PSO-based scheduler. For a projected uncertainty degree of 99%, for instance, the schedules achieved by the RobWE scheduler were approximately 2%, 4%, and 3% fairer for the Montage, LIGO, and SIPHT applications than the schedules produced by the original PSO-based scheduler, respectively. Regarding the execution costs, these fairer schedules have offered considerable economy savings, approximately 6%, 10% and 11%, respectively. Again, the trend observed in Figures 2 and 3 can also be observed in Figure 4, suggesting that the original PSO-based scheduler underwent more degradation in fairness than the proposed RobWE scheduler in the presence of any degree of uncertainty p in the available bandwidth estimates.

D. Discussion

The difference in performance between both schedulers lies in the fact that, to produce robust schedules, the RobWE scheduler uses a deflated value of the measured available bandwidth according to an expected uncertainty of such measurement, while the original PSO-based scheduler [2] uses the measured available bandwidth as a precise information. The results have indicated that the effectiveness of the proposed RobWE scheduler relies on its ability to cope with a high level of uncertainty. The performance of the original PSO-based scheduler fed by imprecise information underwent significant degradation than the performance of RobWE. As workflow applications, especially those grouped into ensembles, generate huge amounts of data during their executions, the scheduler proposed in this paper seems to be attractive for hybrid cloud environments, where the available bandwidth in inter-cloud links may vary and cannot be estimated precisely in advance.

VII. Conclusion

The performance of workflow execution in hybrid clouds depends on the available bandwidth in the inter-cloud links since this bandwidth determines the data transfer time between tasks residing in different clouds. The available bandwidth in these links can fluctuate considerably since Internet links are shared resources. This fluctuation contributes to imprecision estimates of the available bandwidth, and furnishing such inaccurate estimates as an input to a scheduler usually results in the production of poor schedules.

In this paper, the negative impact of uncertainties on the original PSO-based scheduler was used to justify the development of the RobWE scheduler that employs a proactive approach (mechanism) to schedule workflow ensembles involving uncertainty in inter-cloud available bandwidth. The results show that the performance of the RobWE scheduler undergoes less degradation than that of the original PSO-based scheduler as the degree of uncertainty increases. Although the scheduler introduced here represents a preventive approach for the handling of imprecise information, it was not designed to replace reactive approaches, such as self-adjusting scheduling. The integration of the two approaches seems to be a promising option and is left as future work.

Acknowledgment

The first, second, and fourth authors acknowledge grant #2012/02778-6, São Paulo Research Foundation (FAPESP); the European Commission H2020 programme under grant agreement no. 688941 (FUTEBOL), as well as support from the Brazilian Ministry of Science, Technology and Innovation (MCTI) through RNP and CTIC, CNPq and CAPES.

References

[1] S. Sanghrajka, N. Mahajan, and R. Sion, "Cloud performance benchmark series – network performance: Amazon EC2," Stony Brook University, Tech. Rep., 2011.

[2] T. A. L. Genez, L. F. Bittencourt, R. Sakellariou, and E. R. M. Madeira, "A flexible scheduler for workflow ensembles," in *9th International Conference on Utility and Cloud Computing*, 2016, pp. 55–62.

[3] H. Casanova, A. Legrand, D. Zagorodnov, and F. Berman, "Heuristics for scheduling parameter sweep applications in grid environments," in *9th Heterogeneous Computing Wksh.*, 2000, pp. 349–363.

[4] D. M. Batista, L. J. Chaves, N. L. Fonseca, and A. Ziviani, "Performance analysis of available bandwidth estimation tools for grid networks," *J. Supercomput.*, vol. 53, no. 1, pp. 103–121, Jul. 2010.

[5] D. M. Batista and N. L. da Fonseca, "Robust scheduler for grid networks under uncertainties of both application demands and resource availability," *Computer Networks*, vol. 55, no. 1, pp. 3 – 19, 2011.

[6] I. J. Taylor, E. Deelman, D. B. Gannon, and M. Shields, *Workflows for e-Science: Scientific Workflows for Grids*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[7] T. A. L. Genez, L. Bittencourt, N. Fonseca, and E. Madeira, "Estimation of the available bandwidth in inter-cloud links for task scheduling in hybrid clouds," *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, pp. 1–14, 2015.

[8] G. Allen, D. Angulo, I. Foster, G. Lanfermann, C. Liu, T. Radke, E. Seidel, and J. Shalf, "The cactus worm: Experiments with dynamic resource discovery and allocation in a grid environment," *Journal of High Performance Computing App.*, vol. 15, pp. 345 – 358, 2001.

[9] R. Sakellariou and H. Zhao, "A low-cost rescheduling policy for efficient mapping of workflows on grid systems," *Scientific Programming*, vol. 12, no. 4, pp. 253–262, Dec. 2004.

[10] D. M. Batista, N. L. S. da Fonseca, F. K. Miyazawa, and F. Granelli, "Self-adjustment of resource allocation for grid applications," *Computer Networks*, vol. 52, no. 9, pp. 1762–1781, Jun. 2008.

[11] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Algorithms for cost- and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds," *Future Generator Computer Systems*, vol. 48, pp. 1–18, 2015.

[12] Q. Jiang, Y. C. Lee, and A. Y. Zomaya, "Executing large scale scientific workflow ensembles in public clouds," in *44th International Conference on Parallel Processing (ICPP)*, Sept 2015, pp. 520–529.

[13] L. F. Bittencourt and E. R. M. Madeira, "Towards the scheduling of multiple workflows on computational grids," *Journal of Grid Computing*, vol. 8, no. 3, pp. 419–441, 2010.

[14] H. Zhao and R. Sakellariou, "Scheduling multiple DAGs onto heterogeneous systems," in *15th Heterogeneous Computing Workshop (HCW) in conjunction with 20th International conference on Parallel and distributed processing (IPDPS)*, Washington, DC, USA, 2006.