Future Generation Computer Systems 102 (2020) 307–322

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing

Goshgar Ismayilov, Haluk Rahmi Topcuoglu *

Computer Engineering Department, Marmara University, Istanbul, 34722, Turkey

ARTICLE INFO

Article history:
Received 13 March 2019
Received in revised form 13 July 2019
Accepted 6 August 2019
Available online 12 August 2019

Keywords:
Workflow scheduling
Resource failures
Changing number of objectives
Dynamic multi-objective evolutionary algorithms
Neural networks

ABSTRACT

Workflow scheduling is a largely studied research topic in cloud computing, which targets to utilize cloud resources for workflow tasks by considering the objectives specified in QoS. In this paper, we model dynamic workflow scheduling problem as a dynamic multi-objective optimization problem (DMOP) where the source of dynamism is based on both resource failures and the number of objectives which may change over time. Software faults and/or hardware faults may cause the first type of dynamism. On the other hand, confronting real-life scenarios in cloud computing may change number of objectives at runtime during the execution of a workflow. In this study, we propose a prediction-based dynamic multi-objective evolutionary algorithm, called NN-DNSGA-II algorithm, by incorporating artificial neural network with the NSGA-II algorithm. Additionally, five leading non-prediction based dynamic algorithms from the literature are adapted for the dynamic workflow scheduling problem. Scheduling solutions are found by the consideration of six objectives: minimization of makespan, cost, energy and degree of imbalance; and maximization of reliability and utilization. The empirical study based on real-world applications from Pegasus workflow management system reveals that our NN-DNSGA-II algorithm significantly outperforms the other alternatives in most cases with respect to metrics used for DMOPs with unknown true Pareto-optimal front, including the number of non-dominated solutions, Schott's spacing and Hypervolume indicator.

1. Introduction

Cloud computing is a large-scale heterogeneous and distributed computing infrastructure for the scientific and commercial communities, which provides high quality and low cost services with minimal hardware investments. Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) are among the most popular service layers that cloud computing delivers over the internet. In this paper, we will mostly refer to IaaS, where the customers can access hardware resources, on which the applications can be deployed.

Workflows are the common techniques to construct large scale compute and data intensive applications from different research domains. An application workflow is modelled with a directed acyclic graph where the nodes of the graph are tasks that are interconnected via compute or data resources. The workflow scheduling problem in cloud computing aims to map the tasks of a given application onto available resources [1–4]. It is an NP-complete problem [1], in which the orchestration of task executions is the main concern in order to optimize the objectives specified in QoS.

The periodical workflow scheduling mostly involves with the applications that are submitted to the system periodically, where those applications have been increasingly encountered in different domains. They emerge in physics for gravitational waves yearly [5], in business for fiscal analysis monthly and in meteorology for weather forecasting and storm surge prediction daily or hourly [6]. The workloads of the tasks in the applications may vary in every period according to the amount of data they collect. These unpredictable fluctuations lead to period-based scheduling where the tasks should be rescheduled with respect to their latest workloads in every period.

The concept of dynamism in workflow scheduling that we consider in this study is twofold. The first type of dynamism is transient resource failures over time, where the resources may dynamically join to and leave from the cloud. They may arise in consequence of several events such as software faults (bugs, overflows, etc.) or hardware faults (irregular electric power, hard disk failures, etc.). The other source for dynamism in workflow scheduling problem is changing number of objectives during execution of workflows. Cloud computing confronts with real-life scenarios, where the number of objectives may change over time [7]. For instance, makespan of a workflow may not be taken into the consideration until a workflow with tighter deadline is submitted for execution. It is also emphasized that cloud

* Corresponding author.
E-mail address: haluk@marmara.edu.tr (H.R. Topcuoglu).

https://doi.org/10.1016/j.future.2019.08.012
0167-739X/

computing is one of the subjects that is open for more variety of changing objectives to be considered. The objectives can be ignored or considered by several external factors including the level of power consumption, the level of unbalance of workloads among resources or changing QoS requirements. This dynamism has impact on the selection of optimal workflow scheduling solutions in different periods.

To the best of our knowledge, our paper is among the first attempts to model the workflow scheduling problem in cloud computing as a dynamic multi-objective optimization problem (DMOP) by considering the resource failures and changes in the number of objectives as main sources of dynamism. In this paper, we present a novel prediction based dynamic multi-objective evolutionary algorithm, called NN-DNSGA-II algorithm, that incorporates artificial neural network with the non-dominated sorting genetic algorithm (NSGA-II algorithm). The NN-DNSGA-II algorithm combines the strength of the neural network with dynamic NSGA-II to reveal the change patterns in the optimization environments. It exploits correlation between task–resource pairs and correlation between two successive optimization environments in order to estimate the future positions of optimal solutions.

Additionally, we adapt five dynamic multi-objective algorithms from the literature that do not require prediction, including DNSGA-II-A, DNSGA-II-B, DNSGA-II-HM, DNSGA-II-RI algorithms and dynamic variation of the particle swarm optimization algorithm called DMOPSO algorithm. They are included for types of changes in the scenarios that are not predictable, where they have different types of response mechanisms to cope with the changes. While DNSGA-II-A, DNSGA-II-RI and DMOPSO basically insert random solutions to re-diversify the population, DNSGA-II-B inserts mutated versions of the existing solutions in the population. Finally, DNSGA-II-HM use adaptation of mutation rates in the environment.

An extensive empirical study is carried out to demonstrate the performance of our algorithm, where the optimization objectives include the minimization of *makespan*, *cost*, *energy* and *imbalance*; and the maximization of *reliability* and *utilization*. The resource specifications are based on the of Amazon EC2; and the workflows from Pegasus Workflow Management System are the test beds. The comparisons of the algorithms are carried out by three metrics from the multi-objective optimization problems, which are the number of non-dominated solutions, Schott's spacing and Hypervolume. The experimental evaluation validates that our NN-DNSGA-II algorithm outperforms the adapted ones up to 24 out of 30 cases in varying frequency of changes and up to 35 out of 45 cases in varying severity of changes.

The rest of the paper is organized as follows. Section 2 reviews the related work. In Section 3, we present workflow application model, the architecture for workflow execution and the objectives considered in this study. Dynamic workflow scheduling problem is presented in Section 4, which is followed by the proposed NN-DNSGA-II algorithm and adaptations of five dynamic multi-objective algorithms from the literature. Section 5 presents experimental setup and the definitions of metrics considered in this study. The results and discussions of our empirical study is presented in Section 6; and Section 7 concludes the paper.

2. Related work

Workflow scheduling on distributed resources is an extensively studied NP-complete problem [1]. Although there are a few single objective workflow scheduling problems [8,9], most of the existing research have multi-objective characteristics [1,10,11]. Although some of them linearly combine multiple objectives [12,13], it may not appropriate to the nature of the real-world problem at some certain conditions. One limitation is setting weight parameters with correct values. Additionally, a linear combination of objectives may not be able to converge to the optimal solutions that are at the concave parts of the Pareto-optimal front. For workflow scheduling problem in cloud computing, the most common objectives in the literature include makespan, cost, reliability, energy and utilization.

Dynamic workflow scheduling refers to a special kind of scheduling techniques, that may have basic conceptual differences. For a type of dynamic workflow scheduling problem, the prices of resources may vary over time with respect to the supply and demand of the market. On the other hand, for some problems, information for the workflows including their arrivals, size and characteristics are unknown in advance; and scheduling decisions are made at runtime [14–16]. In this study, we extend the definition of dynamic workflow scheduling problem by considering failures of resources and change in the number of objectives in QoS based on changes in workflow execution requirements.

Regardless of the efforts in the literature to model and improve the reliability of cloud computing, the failures of resources are still challenging. To understand the characteristics of the failures, Iosup et al. [17] assesses the resource availability data in large systems based on long-term traces and provides sufficient statistical background for frequency, duration and size of the failures. To cope with these failures, fault-tolerant scheduling is mainly considered as effective technique, which prevents wastage of money, time and energy in cloud computing to a large extent. Zhu et al. [18] attempts to apply extended primary-backup scheduling technique to real-time scheduling in cloud computing in order to tolerate the faults during workflow execution. On the other hand, Plankensteiner et al. [19] uses task resubmission technique with the proposed resubmission impact for reliable workflow execution while meeting their deadlines in clouds.

The dynamically changing number of objectives in a multi-objective optimization has been rarely considered so far in the literature. First, Chen et al. emphasizes importance of changing number of objectives for clouds, since different scheduling scenarios with more variety of objectives may appear in clouds [7]. It also analyses its potential effects on Pareto-optimal front of the problem and the performance of the algorithms. Chen et al. [20] and Zille et al. [21] make some attempts to apply it to distance minimization and scheduling problems, respectively. However, the existing work on cloud computing remains limited since scenarios involving more than two objectives are not investigated [20]. In addition, the main motivation for dynamically changing objectives in the paper is to find feasible solutions in the environment, where no feasible solutions previously exist. However, it does not consider changing objectives any more after it finds at least one feasible solution.

Cloud computing in general is a stochastic environment, which may have several types of delays and uncertainties. The major delays include provisioning and deprovisioning delays for virtual machines (VMs) while the major uncertainties include variations in task runtimes, resource and network performance, where there are a few papers that address them in the literature [22–24]. However, delays and uncertainties during execution may have negative impacts on the efficiency of the base solutions in case they are considered enough when scheduling. This may cause discrepancy between the expected and actual results of the solutions.

The existing dynamic multi-objective evolutionary techniques in the literature can be classified under several categories with respect to the ways they cope with the dynamism. One of the major categories is the prediction-based models, which has especially drawn much attention in parallel with increasing popularity of machine learning techniques [25–30]. The prediction-based

Table 1
Notations.

Symbol	Definition
$RT(t_i)$	The reference time of task t_i
$ST(t_i, r_k)$	The start time of task t_i on resource r_k
$PT(t_i, r_k)$	The processing time of task t_i on resource r_k
$TT(t_i, t_j)$	The transfer time of task t_i and task t_j
$FT(t_i, r_k)$	The finish time of task t_i on resource r_k
CU_k	Computing unit of resource r_k
C_k	Cost of resource r_k
N_k	Number of billing period that r_k is provisioned
B	System bandwidth
DL	Delay
DI	Degree of imbalance
R	Reliability
E	Energy consumption
τ	Billing period

models in this domain can be used for forecasting the time for environmental change or tracking the moving optimal solutions over time. Liu et al. [29] adapted neural network to predict the change direction of optimal solutions to track them in dynamic environment for a single objective optimization problem, where useful knowledge is extracted from past experiences in previous environments to make predictions in the future environments. On the other hand, Jiang et al. [30] adapted support vector machine (SVM) to NSGA-II algorithm for dynamic multi-objective optimization problem, where the solutions in the past Pareto-optimal fronts are labelled as positive and the rest of the solutions are labelled as negative examples.

The main distinction of our study is that we model dynamic workflow scheduling problem in cloud computing by incorporating two types of dynamism which are failures of resources and changing number of objectives for the presented model. Additionally, we propose a novel prediction-based dynamic multi-objective evolutionary algorithm, NN-DNSGA-II. Additionally, we adapt existing dynamic multi-objective algorithms from the literature. They optimize up to six different objectives including minimization of makespan, cost, energy and imbalance; and maximization of reliability and utilization. The algorithms also take the delays and uncertainties into the account while scheduling the workflows.

3. Workflow scheduling problem in cloud computing

3.1. Application model

A workflow application, $G = (T, D)$, is modelled through a directed acyclic graph, where $T = (t_1, t_2, \ldots, t_n)$ is the set of tasks and $D = \left(d_{ij} \mid t_i, t_j \in T\right)$ is the set of edges among precedence-constrained tasks. The weight of a task is defined by its reference execution time in seconds and the weight of an edge is defined by its output data size in bytes. For a task t_i, $pred(t_i)$ refers the set of its immediate predecessors and $succ(t_i)$ refers the set of its immediate successors. A task with no predecessor is t_{entry} and a task with no successor is t_{exit}. A task can be scheduled to only one cloud resource. However, each resource can execute multiple tasks by processing at most one task at a time. Once a task starts, it cannot be interrupted by any other events.

The periodical workflows are certain types of applications whose tasks arrive at the system periodically, which are the workflows considered in this study. The reference execution times and communicational data size may naturally fluctuate to certain extents based on the amount of information collected at each periodic stage. They follow Gaussian distributions as happened in most real-world cases. Such variations leads to period-based workflow scheduling conception, in which the tasks need to be rescheduled in every separate period with respect to their latest workloads. Table 1 lists all the key notations used through the paper.

3.2. System model

Let $R = (r_1, r_2, \ldots, r_m)$ represent a set of heterogeneous computational resources. The IaaS platform offers a range of types with different combinations of resource attributes such as price and computing unit (CU), which is a relative performance indicator. The resources are fully inter-connected and can contain only one virtual machine (VM) at a time. In this regard, the processing time of task t_i on resource r_k is calculated as follows:

$$PT(t_i, r_k) = \frac{RT(t_i)}{CU_k} \tag{1}$$

The transfer time between two tasks depends on bandwidth, which is identical all over the system; and the intra-resource transmission is neglected in the model. The transfer time between two dependent tasks t_i and t_j over a link with bandwidth B is calculated as follows:

$$TT(t_i, t_j) = \begin{cases} \frac{d_{ij}}{B} & r_k \neq r_{k'} \\ 0 & r_k = r_{k'} \end{cases} \tag{2}$$

In our study, the objectives of the workflow scheduling problem are the minimization of *makespan*, *cost*, *energy* and *imbalance*; and the maximization of *reliability* and *utilization* while satisfying the precedence constraints between tasks. In this section, we present formulation of the four objectives (makespan, cost, utilization and imbalance), while two remaining ones (reliability and energy) are given in the following subsections.

To estimate the values of objectives, the start and finish time of the tasks are first calculated in order as follows:

$$ST(t_{entry}, r_k) = ready(r_k) \tag{3}$$

$$ST(t_i, r_k) = max(ready(r_k)), \max(FT(t_p, r_{k'})) \tag{4}$$

$$FT(t_i, r_k) = ST(t_i, r_k) + PT(t_i, r_k) + \sum_{t_c \in succ(t_i)} TT(t_c, t_i) \tag{5}$$

where $t_p \in pred(t_i)$, t_c is the task from the set of $succ(t_i)$ that is not assigned to the same resource with t_i and $ready(r_k)$ shows when the resource is ready to be executed. A resource is considered as ready for execution when it is provisioned and its boot-up ends if not provisioned at the time; or it is free from any task if provisioned at the time. The *makespan* is the maximum of the finish time of the last executed task as follows:

$$\max_{t_i \in T} (FT(t_i, r_k)) \tag{6}$$

In this paper, we build a complete cost model which considers the cost of task execution, output transmission and output storage. The cost of output transmission between two dependent tasks within the same resource is neglected. The *cost* of execution of a workflow in cloud computing is defined as follows:

$$\left(\sum_{r_k \in R} N_k \times C_k^{execution}\right) + \left(\sum_{d_{ij} \in D} d_{ij} \times C_k^{transfer}\right) + \left(\sum_{d_{ij} \in D} d_{ij} \times time(d_{ij}) \times C_k^{storage}\right) \tag{7}$$

where $C_k^{execution}$, $C_k^{transfer}$ and $C_k^{storage}$ are the execution, transmission and storage prices of the resource r_k per unit time, respectively; and $time(d_{ij})$ is the amount of time the output data is stored, which spans from the first production of output data until the end of workflow.

Through the workflow execution, the resources should be efficiently utilized for better cloud system performance. The *utilization* as ratio of total working times of resources to total serviceable times of resources is defined as follows:

$$\frac{\sum_{t_i \in T} PT(t_i, r_k)}{\sum_{r_k \in R}(\tau - DL_{prv} - DL_{deprv}) \times N_k} \tag{8}$$

where DL_{prv} and DL_{deprv} denote the provisioning and deprovisioning delays, respectively. They are removed from the formulation in order to find the *net utilization*.

The degree of imbalance states the balance of task workloads among all available resources. The degree of imbalance in conjunction with utilization objective may lead to more accurate assessments about the performance of a cloud system, since for a highly utilized system, it is possible to have low degree of imbalance. The *degree of imbalance* of an individual resource r_k is:

$$DI_k = \sum_{t_i \in T} PT(t_i, r_k) \tag{9}$$

where DI_k is the summation of tasks executed on resource r_k. Finally, the degree of imbalance of total system is defined as follows:

$$\frac{DI_{max} - DI_{min}}{DI_{avg}} \tag{10}$$

where DI_{max}, DI_{min} and DI_{avg} represent the maximum, minimum and average degree of imbalance values of individual resources, respectively. Finally, a resource that already starts to be shut down cannot execute tasks any more. In case execution of a task is still ongoing, an extra billing period of that resource necessarily begins. A resource is provisioned only when there exists a ready task that is assigned to that resource. Since price of a resource has to be paid even if its billing period is not fully utilized or there may be other tasks waiting to be executed on that resource in the same billing period, it is shut down only at the end of the billing period. Additionally, we consider the delays for resource provisioning and deprovisioning in this study.

3.2.1. Reliability model

In this study, we quantitatively assess the safety of the cloud computing to failures of resources. It is based on long-term empirical resource availability data presented in a related work [17]. It shows that Weibull is the closest distribution to model the inter-arrival time between failures and the size of failures. The two-parameter Weibull distribution with scale (η) and shape (β) parameters is:

$$p(x) = \frac{\beta x^{\beta-1}}{\eta^{\beta}} e^{-\left(\frac{x}{\eta}\right)^{\beta}} \tag{11}$$

where x is a random variable, which represents a point in time and $p(x)$ represents the failure probability at that point in time. In this paper, we follow Weibull distribution to generate random and statistically independent resource failures. For a single task, the integration of individual failure probabilities at each time point during its execution interval yields its cumulative likelihood of being not successfully completed. So, the reliability of a task can be defined as follows:

$$R(t_i, r_k) = 1 - \int_{ST(t_i, r_k)}^{FT(t_i, r_k)} p(x)dx = e^{-\left(\frac{PT(t_i, r_k)}{\eta}\right)^{\beta}} \tag{12}$$

The multiplicative aggregation of the reliability for each task gives the reliability of overall workflow, which is defined as follows:

$$R(G) = \prod_{t_i \in T} R(t_i, r_k) \tag{13}$$

The multiplicative inverse of reliability, $R(G)$, has been used for ease of use. In periodical workflow scheduling, once resources temporarily fail, they cannot participate into that scheduling period. The duration of failures for recovery is comparably smaller than the inter-arrival time between failures [17]. It can be inferred that the resources that fail in previous scheduling period, are capable of taking part in the next period. In the paper, we assume that the communicational components in the system are reliable.

3.3. Energy model

Our DVFS-based energy model is based on power consumption of complementary metal–oxide semiconductor (CMOS) logic circuits. In the paper, we consider only the dynamic power dissipation which is the most influential term in the model. On this basis, the executional energy for a workflow is the linear integration of the energy required for execution of each task as follows:

$$E^{execution} = \sum_{t_i \in T} K \times f_k^3 \times PT(t_i, r_k) \tag{14}$$

where the supply voltage is expressed as the linear function of frequency, f, in the equation. The cloud resources are assumed to be operating at the highest frequency level ($f = 1.0$) during execution and the lowest frequency level ($f = 0.4$) during the output transmission. The communicational energy for a workflow is also the linear integration of the energy required for each transmission as follows:

$$E^{transfer} = \sum_{t_i, t_j \in T} K \times f_{min}^3 \times TT(t_i, t_j) \tag{15}$$

To save energy in the cloud system, the frequencies of the resources are set to the lowest level when they are idle. The energy spent by idle resources is calculated with the same way for communicational energy, but the durations of idle times of the resources are considered instead of transfer times. The failed resources do not spend any further energy. Altogether, the total energy for a workflow includes the energy for execution, communication, idling and failure.

4. Dynamic workflow scheduling problem

A multi-objective optimization problem (MOP) is an optimization problem that has two or more objectives, where at least two objectives may be in conflict with one another. For a MOP, the set of solutions that are not dominated by any other solutions is called as *Pareto-optimal set* (POS) in decision space and *Pareto-optimal front* (POF) in objective space. *True POF* is not known a priori for this problem. On the other hand, a dynamic multi-objective optimization problem (DMOP) is a MOP, in which constraints, objectives or parameters that may dynamically change over time [31]. Formally, a DMOP can be defined as:

$$\begin{cases} min\ F(x, t) = (F_1(x, t), F_2(x, t), \ldots, F_o(x, t))^T \\ s.t.\ g_i(x, t) \leq 0, h_i(x, t) = 0 \end{cases} \tag{16}$$

where x is the vector of decision variables and F is the set of objectives to be minimized with respect to time t [32].

Workflow scheduling problem becomes more challenging and more realistic when it is addressed in a dynamic environment. Specifically, the first type of dynamism considered in this study for workflow scheduling is the failures of resources. The large-scale systems including clouds are prone to the failures, which may have significant impacts on workflow executions. They may even crash a part or whole of the system. In case of the failures, the set of resources available may change, which lead POS, POF or

both of them to change as well. The target algorithms for dynamic workflow scheduling problem are expected to track and converge new POF before the next failures. In the experimental study, they are exposed to varying frequency and size of resource failures.

Changing number of objectives is the second type of dynamism considered for workflow scheduling in this study. It emerges when workflows arrive at the system with different requirements. A workflow scheduler in cloud should be capable of satisfying changing workflow requirements over time. This dynamism specifically alters the shape of objective space of the problem. An increase in number of objectives leads to the expansion of POF, which deteriorates the diversity of the population [7]. This poses challenges among the solutions in terms of Pareto-domination since it gets harder for a solution to be better than another one in every objective. This prevents algorithms from converging close to the *POF*. On the other hand, a decrease in number of objectives leads to contraction of POF, which creates similar and duplicate solutions [7]. In order to generate test scenarios for this type of dynamism, we use the following generalization of the periodic change in objective size:

$$o(t) = o(t+c) = \begin{cases} m & t = 0 \\ m+k & t \in [a, b] \\ m-k & t \in [b, c] \end{cases} \tag{17}$$

where m is starting number of objectives, k is severity of change, t is time and a, b, c are the certain discrete points in time. The objective size gradually increases from a to b and decreases from b to c.

For both dynamic extensions of workflow scheduling, the problem stays stationary between two successive changes. The effects of both resource failures and changing number of objectives are reflected to the problem at the end of each period. Whenever the problem changes, the old instance of periodic workflow is considered as completed and dismissed. It is replaced by new instance of the same periodic workflow with the new workloads in the new period.

4.1. A new prediction-based dynamic multi-objective evolutionary algorithm

The dynamic workflow scheduling problem is a multi-objective optimization problem with dynamic characteristics that requires algorithms to react to changes effectively and track the changing POF over time. That is why, leading static multi-objective algorithms are not appropriate for this problem. We propose a novel prediction-based dynamic multi-objective evolutionary algorithm, called NN-DNSGA-II. Our algorithm integrates artificial neural network with the non-dominated sorting genetic algorithm (NSGA-II) [33], which is one of the most efficient algorithm in solving static multi-objective problems.

Although there are increasing number of dynamic multi-objective evolutionary algorithms (DMOEAs) in different categories proposed in the literature, prediction-based methods attract the attention of the researchers, which integrates multi-objective evolutionary algorithms with selected machine learning techniques seamlessly [25–30]. By the integration, the corresponding strategy targets to solve DMOP by gathering the history of the past POS information and exploit it to estimate next POS at the problem. It reveals the patterns behind the displacements of POS in two successive scheduling periods and utilizes the estimated new solutions as initialization points for the next scheduling period.

As can be seen in Algorithm 1, NN-DNSGA-II starts with random re-initialization of parent and child populations with N number of individuals for each (line 1). When a change happens in the environment (lines 3–10), the multi-layer feedforward neural network is activated. The nearest solutions between POF_j and POF_{j+1} of two successive environments are paired at first (line 4) and then these pairs are included to the training set (line 5). The Pareto-optimal solutions from the first of the successive environments are treated as input data and the Pareto-optimal solutions from the second of the successive environments are treated as output data. Each input solution can corresponds to only one output solution, which means there is one-to-one relationship between input and output sets. It is clear that the number of paired solutions is equal to the minimum of number of input or output solutions. The pairing process can be seen in Fig. 1, where POF_1 includes input set and POF_2 includes output set.

The pairing of the closest solutions in terms of Euclidean distance in decision space has time-complexity of $O(n^2)$, where n is the number of tasks in the workflow. This complexity leads to very high running times, when scheduling workflows that have 1000 tasks or more. In this paper, we will use the pairing of the closest solutions in terms of Euclidean distance in objective space, which is insensitive to the number of tasks. Instead, it is only interested in number of objectives, which is relatively very small compared with number of tasks.

The neural network goes through the training phase by using the cumulative training set until the given number of epoch is satisfied (line 6). Then, it becomes ready to predict locations of Pareto-optimal solutions. In the prediction process, the most recently found Pareto-optimal solutions is fed to the network as input to generate output solutions for the initial population of the next environment (line 7). The new solutions are replaced with the randomly chosen solutions from the parent population (line 8). At the end, the infeasible solutions are repaired to make them feasible again (line 9).

Our neural network model that is integrated into the optimization environment has three-layered architecture: an input, a hidden and an output layer. The number of neurons in the input and the output layers is equal to the number of tasks in the workflow; and the number of neurons in the hidden layer is equal to twofold of the number of tasks. The neurons in the input layer take the resource numbers as input, which range from zero to maximum number of resources in the system and correspondingly, the neurons in the output layer generate the resource numbers as output in the same range. Mean Absolute Error (MAE) is used as linear loss function to measure the proximity between the predicted and the actual results. In addition, the standard back propagation method is used and the maximum number of epoch is set to ten for training phase.

The steps (lines 11–15) are the standard run of NSGA-II, where it first merges the parent and child population into single population with $2N$ individuals (line 11). This operation ensures the elitism in the environment. The individuals altogether are sorted into different non-domination and crowding density levels with respect to fast non-dominated sorting and crowding distance sorting scheme, respectively (lines 12–13). From $2N$ individuals, the best N ones are selected for parent population of the next generation (line 14). In selection process, the individuals with lower rank are preferred. If they are at the same rank, the ones from less crowded regions are preferred. At the end, the new parent population are used to generate offspring (line 15).

4.2. Adapting dynamic multi-objective evolutionary algorithms

As part of this paper, we also adapt leading dynamic multi-objective evolutionary algorithms from the literature for solving dynamic workflow scheduling problem in cloud computing. Specifically, we consider four dynamic variants of the NSGA-II algorithm (DNSGA-II-A [34], DNSGA-II-B [34], DNSGA-II-HM [35],

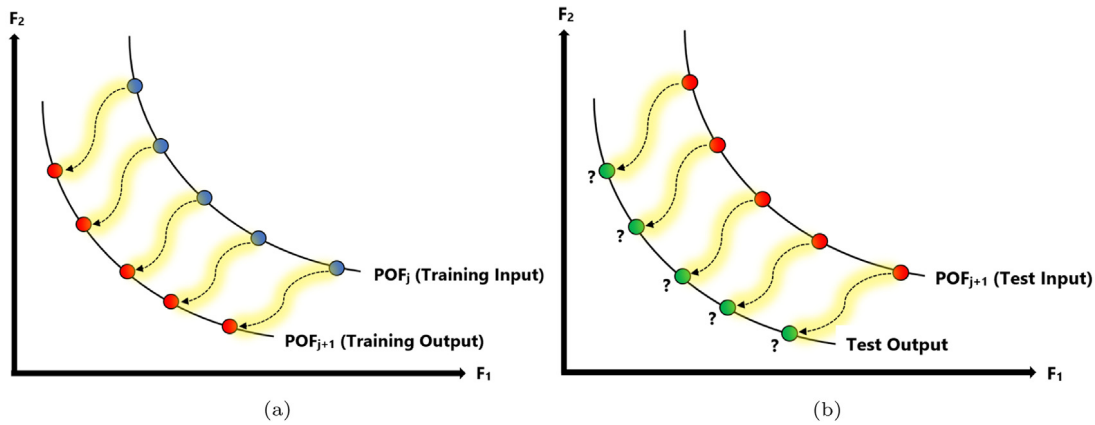

Fig. 1. (a) The optimal solutions in *POF*s of last two successive environments are paired and added to training data. (b) The optimal solutions in the most recently found *POF* is used by NN-DNSGA-II to generate new solutions.

Algorithm 1 NN-DNSGA-II

Require: P : Parent population, Q : Child population, R : Merged population, T : Training Set, N : Neural network-generated solutions
Ensure: POF : Pareto-optimal Front
1: InitializePopulations(P, Q)
2: **while** termination condition not satisfied **do**
3: **if** change happened **then**
4: [T_j.Input, T_{j+1}.Output] = PairOptimalSolutions(POF_j, POF_{j+1})
5: $T = T \cup$ [T_j.Input, T_{j+1}.Output]
6: TrainNeuralNetwork(T)
7: N = TestNeuralNetwork(T_{j+1}.Output)
8: P = ReplaceSolutions(P, N)
9: RepairSolutions(P)
10: **end if**
11: $R = P \cup Q$
12: R_1 = FastNonDominatedSort(R)
13: R_2 = CrowdingDistanceSort(R)
14: P = SelectBestSolutions(R, R_1, R_2)
15: Q = GenerateChildren(P)
16: **end while**

DNSGA-II-RI [36]) and one dynamic variant of the MOPSO algorithm (DMOPSO [37]). They each have different types of response mechanisms to cope with the changes occurred in the optimization environments, where we briefly summarize their details in this section.

- DNSGA-II-A [34] re-initializes certain number of solutions in the population. The solutions that are relocated can track the Pareto-optimal solutions relatively in the distant parts of search space of the new environment. We provide a pseudocode of DNSGA-II-A in Algorithm 2 as a representation for other algorithms in this section. The steps (lines 7–11) belong to classical NSGA-II algorithm. DNSGA-II-A relocates a percentage of population randomly in the search space to respond to the change (line 4).
- DNSGA-II-B [34] replaces a percentage of solutions in the population with the mutated versions of the existing solutions. While completely context-free solutions are added to the population in DNSGA-II-A, contextual relationship among solutions is preserved in DNSGA-II-B.
- DNSGA-II-HM (Hyper-mutation) [35] re-diversifies the population by increasing the mutation rates, which are decreased again after certain number of generations. The high mutation rates distribute individuals in the population to the nearby of their current positions in the search space.
- DNSGA-II-RI (Random immigrants) [36] maintains diversity without explicitly detecting the environmental changes by re-initializing certain number of solutions at each generation.
- DMOPSO [37] is a dynamic and multi-objective extension of the particle swarm optimization, which uses adaptive grid and mutation schemes. As in DNSGA-II-A, it randomly re-initialize certain number of solutions in the population as well when change is happened.

Algorithm 2 DNSGA-II-A

Require: P : Parent population, Q : Child population, R : Merged population, θ : Pre-defined percentage of population
Ensure: POF : Pareto-optimal Front
1: InitializePopulations(P, Q)
2: **while** termination condition not satisfied **do**
3: **if** change happened **then**
4: RelocateSolutions(P, θ)
5: RepairSolutions(P)
6: **end if**
7: $R = P \cup Q$
8: R_1 = FastNonDominatedSort(R)
9: R_2 = CrowdingDistanceSort(R)
10: P = SelectBestSolutions(R, R_1, R_2)
11: Q = GenerateChildren(P)
12: **end while**

It should be noted that all algorithms consider a two-dimensional string representation for encoding the solutions. The first dimension stores the relative execution order of the tasks that preserves the precedence constraints and the second one presents mapping of the tasks to the resources. A random initialization is performed on the solutions, whose lengths are specified by number of tasks in the workflow. The first dimension stores the relative execution order of the tasks and the second one maps the tasks to the resources. The single-point crossover given in [38] is utilized, since it produces valid strings by preserving the precedence constraints between tasks. For a randomly selected task, a mutation operator randomly repositions it at the first dimension; and another one assigns a random resource to it.

5. Experimental study

5.1. Experimental setup

The algorithms are evaluated by using ten real world workflows with approximately 100 and 1000 tasks from Pegasus Workflow Management System [39]. They include Montage from astronomy, CyberShake from geology, Epigenomics from biology, Inspiral from physics and Sipht from bioinformatics. The Fig. 2 symbolically represent their topological structures such as the task pipelines, data distributions and aggregations. The specifications of the resources in Table 2 are compatible with Amazon EC2, US East (Ohio) region [40]. The delays in seconds for resource

Table 2
Instance types and their specifications.

Type	Compute unit (cu)	Price ($)
m4.large	6.5	0.1
m5.xlarge	16	0.192
m5.2xlarge	31	0.384
m5.4xlarge	60	0.768
m5.12xlarge	173	2.304
m5.24xlarge	345	4.608

Table 3
Default parameter settings.

Parameter	Value
Population size	50
Task size	100/1000
Resource size	60
Archive size	50
Frequency of change (η, β)	(13, 12)
Severity of change (η, β)	(2, 2)
Number of runs	30
Stopping criterion (generation)	1000

provisioning and deprovisioning are generated by Gaussian distributions, $N(100, 10)$ and $N(10, 2)$, respectively. They are based on real evidences provided in [41]. The uncertainties for reference task execution times, resource and bandwidth performance are also generated by multiplying the original values with a random Gaussian number from $N(1.0, 0.1)$.

For the evolutionary algorithms, the number of generations reserved for each period is equal to the division of total number of generations by total number of changes. The crossover rate is 100%, the population replacement rate is 10% and finally the mutation rates for task repositioning, resource rescheduling are 1%. Table 3 lists the values for all other default parameters. The logarithmic time-scale has been employed in resource failures for Weibull distribution [17]. The billing period for resources is 60 min and all partial usages are rounded up. The bandwidth is 1 Gbps in the system. The utilization and the imbalance are used with the remaining four objectives only in the experiments to test the algorithms under changing number of objectives.

5.2. Performance metrics

In this paper, we consider three performance metrics to measure the effectiveness of the algorithms to find good non-dominated solutions in dynamic environments, which are *number of non-dominated solutions* (*NS*), *Schott's spacing* (*SS*) and *Hypervolume* (*HV*). It should be noted that those are the metrics used for dynamic multi-objective optimization problems with unknown true *POF*, which are appropriate to our dynamic workflow scheduling problem [42].

Number of non-dominated solutions (*NS*) [42] counts the average of the number of non-dominated solutions found at each generation just before the changes in the environment. It is defined as follows:

$$NS = |POF_{known}| \tag{18}$$

where $|.|$ operator returns the amount of solutions that are parts of the known *POF*. For this metric, it is better for the algorithms to find non-dominated solutions as much as possible. However, this may be misleading when a new solution dominates and discards the solutions already in the known *POF*. From this viewpoint, it is recommended to evaluate the results of this metric along with the results of the other metrics.

Schott's spacing (*SS*) [43] measures how regularly the non-dominated solutions are spread into the known *POF* at each generation just before the changes by considering the distance variance of neighbour solutions. It is defined as follows:

$$SS = \frac{1}{|POF_{known}|}\left[\frac{1}{|POF_{known}|}\sum_{i=1}^{|POF_{known}|}(d_i - \bar{d})\right]^{\frac{1}{2}} \tag{19}$$

where d_i is the Euclidean distance between non-dominated solution i and its nearest neighbour; and $\bar{d}$ is:

$$\bar{d} = \frac{1}{|POF_{known}|}\sum_{i=1}^{|POF_{known}|} d_i \tag{20}$$

The metric prefers the algorithms that generate the known *POF*, in which the non-dominated solutions are well-distributed and not concentrated in certain parts of the objective space.

Hypervolume (*HV*) [44] measures the amount of space covered by non-dominated solutions found at each generation just before the changes, with respect to the pre-defined reference point. The space covered by a single non-dominated solution is defined as follows:

$$HV(i) = \{i' \in O : i \prec i'\} \tag{21}$$

where O represents the objective space and the total space dominated by union of non-dominated solutions is:

$$HV(POF) = \bigcup_{i \in POF} HV(i) \tag{22}$$

For this metric, the fitness values of the scheduling solutions are normalized by the upper bounds found so far in the experimental study for all the workflows. The upper bound for each objective are mostly determined by the large workflows. However, the normalization by such numerically large values hinders the real improvements of the algorithms. That is why, even very small differences among the algorithms in HV metric actually corresponds the very significant improvements in the objectives. After normalization, the reference point for each objective is set to 1.1 [45].

6. Results and discussion

The results of our experimental evaluation are given in two parts. The first subsection presents the performance evaluation of the algorithms for resource failures. The second subsection is for validating performance of our algorithms for changing number of objectives. In order to indicate the significance between the results of the algorithms, the Wilcoxon ranksum test [47] is carried out at the 0.05 significance level for each table that consider workflows with 100 tasks (specifically, Table 4, 6, 8, and 10), where the best results are shown as bold. The results with "+" sign in those tables are significantly outperformed by the best results in bold for the selected case that is considered. As an example, the NS value of NN-DNSGA-II given in Table 4 is marked bold for the Montage workflow when change frequency is 100. In this case, based on the NS values, NN-DNSGA-II statistically outperforms DNSGA-RI and DMOPSO and it outperforms DNSGA-II-HM, the DNSGA-II-A and DNSGA-II-B.

6.1. Measuring effect of resource failures

6.1.1. Varying frequency of changes

In the first experiment of this category, we measure effects of varying change frequency (frequency of resource failures) on the performance of the algorithms. A scheduling period is assigned to 100 generations in high-frequency cases and 500 generations in low-frequency cases. In low cases, we expect to have (1) greater number of non-dominated solutions for *NS*, (2) smaller deviation

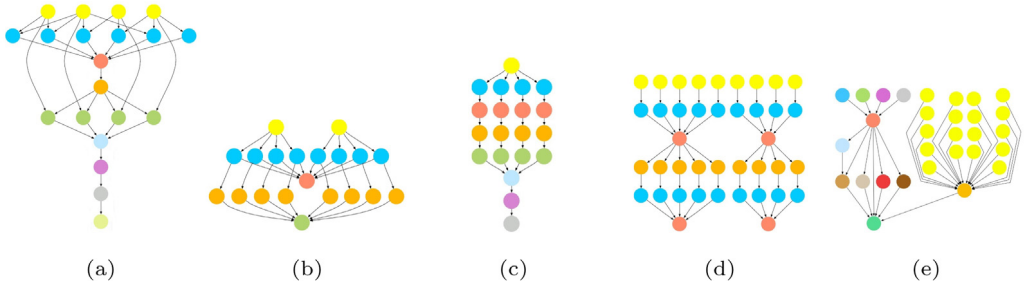

Fig. 2. The structures of Pegasus workflows [25] : (a) Montage, (b) CyberShake, (c) Epigenomics, (d) Inspiral and (e) Sipht.

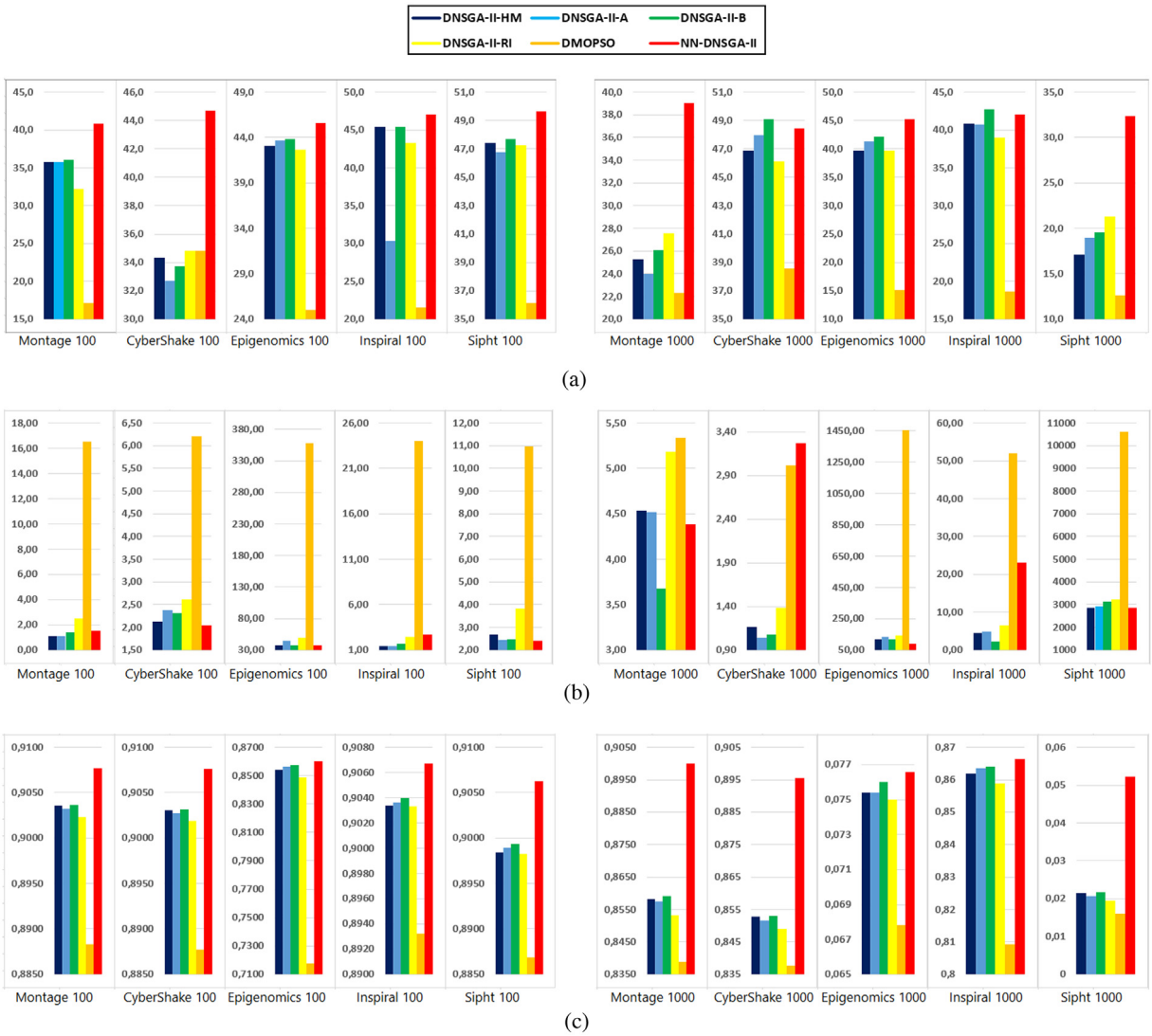

Fig. 3. Performance of algorithms in (a) *NS*, (b) *SS* and (c) *HV* metrics for average resource size.

in distribution of these solutions for *SS* and (3) higher coverage area in objective space for *HV*.

As seen in Tables 4 and 5, the algorithms tend to find more non-dominated well-distributed and well-converged solutions for most of the cases when $f = 500$. It can be inferred that the larger workflows have more impacts on the performances of the algorithms. Compared with other workflows with 100 tasks, Epigenomics 100 is more complex with higher average data size and average execution time. For this workflow, the algorithms have notably poor performances in *SS* and *HV* metrics. The performances of algorithms are better when they deals with the workflows with 100 tasks rather than workflows with 1000 tasks. Among all algorithms, NN-DNSGA-II outperforms the other alternatives, where it provides best *NS* and *HV* values for almost all of the cases. However, there is no algorithm that is superior than the others when *SS* is considered. Overall, NN-DNSGA-II is superior than the other algorithms for 24 out of 30 cases for the workflows with 100 tasks and 19 out of 30 cases for the workflows with 1000 tasks. On the other hand, DMOPSO is the least appropriate algorithm due to its poor performance in every metrics. The particle swarm optimization was initially proposed for continuous space problems and its performance

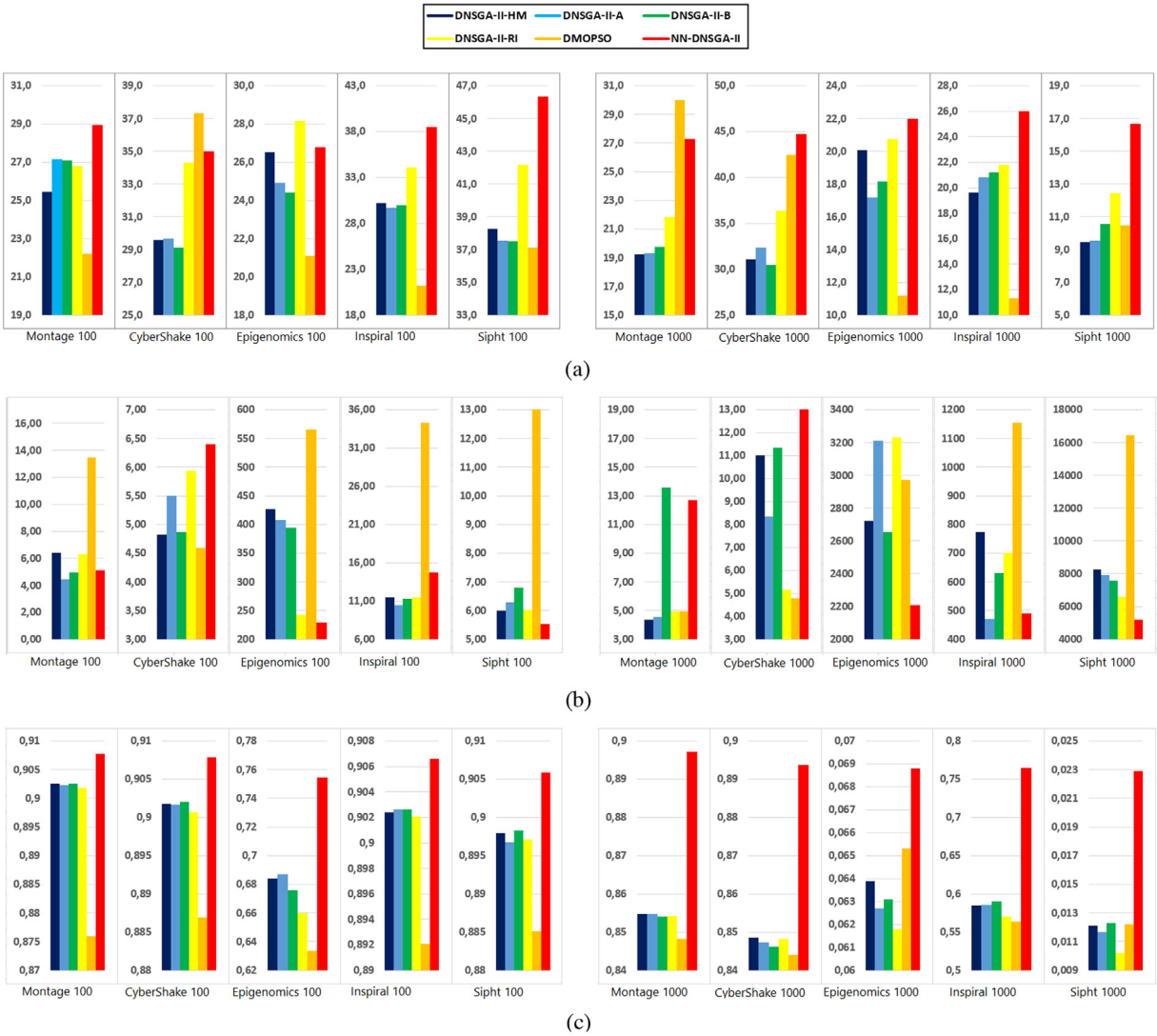

Fig. 4. Performance of algorithms in (a) *NS*, (b) *SS* and (c) *HV* metrics for average uncertainty.

greatly suffers from the discrete problems and cannot satisfy the requirements of the discrete problems.

6.1.2. Varying severity of changes

We investigate the performance variations of the algorithms under different severity of changes (or size of resource failures) in this experiment. The number of resources that fail at every period is 2, 4, 8 in small, medium and high severity cases, respectively.

It can be inferred from the results presented in Tables 6 and 7 that when the landscape of the search space of the problem changes much, the algorithms have difficult times to converge new POF in the next environment. The lack of convergence emerges in the metrics as decrease of *NS* and *HV* values and increase of *SS* values. The algorithms show better performances when the workflow size is lower. The NN-DNSGA-II algorithm outperforms the other algorithms for 35 out of 45 cases for the workflows with 100 tasks; and 32 out of 45 cases for the workflows with 1000 tasks. It mostly outperforms the other algorithms in *NS* and *HV* metrics while it has still some drawbacks to find well-distributed solutions. It is mainly originated from the fact that NN-DNSGA-II can find non-dominated solutions at the end points of Pareto-front, which results in high distance variation between the extreme solutions. It may be useful to have good solutions from large range of Pareto-front in terms of decision-making. As happened in the experiment to measure the effect of change frequency, DMOPSO is the worst algorithm.

The running times of the algorithms in a single period are in the range of 1.1 to 2.8 s for workflows with 100 tasks. They are in range of 21.1 to 102.9 s for the cases with 1000 tasks. The DMOPSO algorithm finishes earlier than the others in the most cases while NN-DNSGA-II consumes relatively more time due to training phase of the neural network.

6.1.3. Varying resource size

We also measure the performances of the algorithms on average resource size for various workflows. The average of 30, 60 and 120 resources is considered for each workflow. The experimental results are presented in Fig. 3. The proposed NN-DNSGA-II algorithm outperforms the other algorithms in *NS* and *HV* metrics. Its performance in Epigenomics 100 and 1000 workflows is the best one among the workflows. However, the performance of NN-DNSGA-II in *SS* metric is not as satisfying as its performance in other metrics. In addition, there is a natural degradation in the performances of the algorithms when the workflows become bigger (i.e., the number of tasks is increased from 100 to 1000).

Fig. 5. Performance of algorithms in (a) *NS*, (b) *SS* and (c) *HV* metrics for changing number of objectives.

6.1.4. Varying variance of uncertainties

We vary the variance of the uncertainties in cloud computing to observe their impacts on the performance of the algorithms, where the uncertainties include provisioning delay, deprovisioning delay, task execution times, bandwidth and finally resource performance. In the experiment, for an uncertainty with base value u_i, its realized value is generated by $u_i \times N(\mu, \sigma^2)$. The variance takes the value of 0.1, 0.3 and 0.5 in the test environment with low, medium and high uncertainty, respectively.

The empirical results show that the environment with high uncertainty poses greater challenges to the algorithms to cope with. According to the given average results for three levels of environments in Fig. 4, the proposed NN-DNSGA-II algorithm outperforms the other algorithms based on *NS* and *HV* metrics for most cases. Especially, its performance in *HV* is notable greater than the other algorithms. On the other hand, NN-DNSGA-II still suffers from finding sufficiently well-distributed non-dominated solutions except from a few cases including Epigenomics 1000.

6.2. Measuring effects of changing number of objectives in the problem

In this experiment, we validate the response of the algorithms when the number of objectives is updated, which is the second form of dynamism considered for the workflow scheduling problem. For this experiment, we initially consider three objectives which are makespan, cost and reliability. Energy, utilization and imbalance are the objectives that are added to or discarded from the set as the fourth, fifth and sixth ones. The number of objectives $o(t)$, changes one at a time for each distinct workflow scheduling period t as follows;

$$o(t) = o(t+6) = \begin{cases} 3 & t = 0 \\ o(t-1) + 1 & t \in [1, 3] \\ o(t-1) - 1 & t \in [4, 6] \end{cases} \tag{23}$$

We specify the minimum number of optimization objectives as three and the maximum number of optimization objectives as six for this category of the experiments for workflow scheduling problem. One by one change in the number of objectives

Table 4
Varying frequency of changes on workflows with 100 tasks.

Algorithm	Metric	Montage 100		CyberShake 100		Epigenomics 100		Inspiral 100		Sipht 100	
		$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$
DNSGA-II-HM	NS	35.6	44.2+	32.1+	39.9+	42.5+	46.8	44.8	47.4+	47.1+	49.6+
	SS	2.42	0.29	1.88	2.49+	63.66	32.89	1.99	**0.96**	2.63	1.49
	HV	0.9029	0.9038	0.9035	0.9043	0.8595	0.8611	0.9025	0.9042	0.8989	0.9004
DNSGA-II-A	NS	37.8	45.9	34.8+	42.9	44.1	46.3	43.9	46.8+	47.2+	49.5+
	SS	1.88	0.26	2.11	**0.91**	92.88	32.12	1.89	1.04	3.89	3.56
	HV	0.9023	0.9041	0.9012	0.9031	0.8583	0.8616	0.9030	0.9031	0.8934+	0.8967+
DNSGA-II-B	NS	37.9	47.8	33.6+	43.1	41.2+	46.9	45.0	48.5+	46.9+	49.7+
	SS	**0.97**	**0.25**	**1.81**	2.63+	44.59	28.24	**1.09**	1.44	2.52	2.04
	HV	0.9038	0.9039	0.9025	0.9043	0.8550	0.8603	0.9033	0.9045	0.8966	0.8991
DNSGA-II-RI	NS	31.2+	47.5	36.4	40.1+	44.2	46.7	44.2	48.4+	47.6+	48.9+
	SS	2.12	0.67	2.68	1.69	49.72	37.99	2.23	1.14	3.14	2.12
	HV	0.9019	0.9033	0.9012	0.9039	0.8561	0.8585	0.9010	0.9024	0.8966	0.8975+
DMOPSO	NS	16.9+	21.9+	35.3	41.1	23.9+	29.1+	21.6+	25.2+	33.9+	43.9+
	SS	16.6+	14.2+	4.91+	4.75+	400.06+	300.55+	32.52+	18.93+	15.83+	4.29+
	HV	0.8888+	0.8900+	0.8852+	0.8878+	0.7143+	0.7832+	0.8946+	0.8989+	0.8805+	0.8812+
NN-DNSGA-II	NS	**40.3**	**48.1**	**42.3**	**44.1**	**48.2**	**48.8**	**46.6**	**49.9**	**49.9**	**50.0**
	SS	2.62+	1.59+	3.91+	4.31+	**34.1**	**19.12**	5.10+	3.47+	**2.38**	**1.56**
	HV	**0.9089**	**0.9089**	**0.9071**	**0.9074**	**0.8611**	**0.8624**	**0.9054**	**0.9076**	**0.9032**	**0.9065**

Table 5
Varying frequency of changes on workflows with 1000 tasks.

Algorithm	Metric	Montage 1000		CyberShake 1000		Epigenomics 1000		Inspiral 1000		Sipht 1000	
		$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$
DNSGA-II-HM	NS	23.8	**39.8**	48.1	49.8	**43.2**	48.3	38.5	46.1	18.3	21.3
	SS	**2.34**	0.98	1.08	1.13	75.04	63.89	5.25	2.94	3168.02	1690.79
	HV	0.8586	0.8655	0.8560	0.8612	0.075	0.0771	0.8632	0.8763	0.0213	0.0246
DNSGA-II-A	NS	25.8	39.1	48.9	49.7	37.5	48.6	40.4	46.4	16.7	21.2
	SS	3.62	**0.89**	0.87	1.32	201.34	**36.71**	5.66	2.42	3184.73	2209.74
	HV	0.8594	0.8667	0.8552	0.8595	0.0754	0.0772	0.8650	0.8764	0.0218	0.0238
DNSGA-II-B	NS	26.6	37.0	49.4	**49.9**	39.1	48.8	40.6	**47.3**	17.3	20.5
	SS	5.61	1.90	**0.80**	**1.11**	135.86	42.13	**4.70**	**2.25**	2196.05	1783.67
	HV	0.8618	0.8650	0.8560	0.8624	0.0757	0.0773	0.8654	0.8769	0.0219	0.0224
DNSGA-II-RI	NS	26.0	35.3	45.1	49.7	42.0	49.6	40.3	46.8	23.1	28.4
	SS	5.41	4.78	1.10	1.17	100.68	46.11	6.37	3.08	2674.84	**1493.78**
	HV	0.8572	0.8627	0.8538	0.8575	0.0752	0.0772	0.8610	0.8739	0.0210	0.0225
DMOPSO	NS	21.7	25.6	42.2	44.7	14.0	16.3	16.5	21.2	13.6	15.0
	SS	4.71	4.06	2.38	2.51	1362.16	838.46	85.64	52.67	9615.27	9222.60
	HV	0.8489	0.8491	0.8476	0.8472	0.068	0.0690	0.8129	0.8196	0.0163	0.0178
NN-DNSGA-II	NS	**36.8**	**39.8**	**49.8**	45.5	37.9	**49.7**	**40.8**	39.9	**32.6**	**33.6**
	SS	8.78	4.18	8.28	5.60	**60.47**	60.47	104.03	104.87	**2084.53**	2819.11
	HV	**0.9027**	**0.9080**	**0.9005**	**0.8890**	**0.0768**	**0.0775**	**0.8719**	**0.8958**	**0.0322**	**0.0336**

from three to six and then from six to three, causes periodical fluctuation at every six discrete time t.

The reactions of the algorithms to the expansion and the contraction of the objective space are analysed in Fig. 5. In the horizontal axis of the figure, the first number is the number of changes and the second number between parenthesis is the number of objectives used in that period. It is observed that the algorithms follow certain patterns. In Fig. 5(a), the number of found non-dominated solutions increases in the expansion of the objective space, since the domination among solutions in the population is harder. The number of non-dominated solutions push the limit of the archive, 50, when number of objectives is 6 in most cases. In Fig. 5(b), we see that the algorithms that are sensitive to *SS* metric such as NN-DNSGA-II and DMOPSO, fluctuate more than the other algorithms. In Fig. 5(c), when the number of objectives decreases in the periods from 3 to 6, the performances of the algorithms gradually improves for converging close to POF. On the other hand, when the number of objectives increases in the periods from 6 to 9, they become gradually worse, which is associated with the absence of sufficient diversity in the population due to expansion of POF. While the performances of the algorithms are at the best level in the periods 3 and 9, where the maximum number of objectives is reached, they are at the worst level in the period 6, where the minimum number of objectives is reached. In general, NN-DNSGA-II algorithm outperforms the other alternatives with respect to *NS* and *HV*.

6.2.1. Further analysis of changing number of objectives

In this section, we update the frequency of changes and severity of changes while changing the number of objectives dynamically. In the first test, we consider 100 and 500 generations per change for high and low frequency cases. From the empirical results provided in Tables 8 and 9, all algorithms show better performance in all metrics when the objectives change rather slowly, i.e., $f = 500$. In other words, all algorithms are better when scheduling smaller workflows. We can clearly see the performance difference of the same algorithms between Epigenomics 100 and Epigenomics 1000. The NN-DNSGA-II algorithm outperforms the other algorithms for 25 out of 30 cases and 22 out of 30 cases for workflows with 100 and 1000 tasks respectively. On the other hand, NN-DNSGA-II still suffers from finding well-distributed solutions except from Epigenomics 100 and Inspiral 100 workflows.

Table 6
Varying severity of changes on workflows with 100 tasks.

Algorithm	Metric	Montage 100			CyberShake 100			Epigenomics 100		
		$s = 2/60$	$s = 4/60$	$s = 8/60$	$s = 2/60$	$s = 4/60$	$s = 8/60$	$s = 2/60$	$s = 4/60$	$s = 8/60$
DNSGA-II-HM	NS	37.2	35.6	32.9+	36.1+	31.9+	31.9+	45.4+	42.5+	42.1
	SS	1.48	2.42+	1.69	**1.60**	1.88	2.45	40.75	63.66	49.00
	HV	0.9049	0.9029	0.9032	0.9040	0.9035	0.9008	0.8604	0.8595	0.8568
DNSGA-II-A	NS	37.9	37.8	33.2+	36.0+	34.8+	33.0+	44.9+	44.1	43.1
	SS	1.03	1.88	4.42+	2.44	2.11	**2.27**	55.12	92.88	49.13
	HV	0.9045	0.9023	0.9021	0.9039	0.9012	0.9015	0.8619	0.8583	0.8562
DNSGA-II-B	NS	37.9	37.9	33.1+	38.1	33.6+	32.9+	44.1+	41.2+	41.8+
	SS	**0.60**	**0.99**	**1.59**	1.99	**1.85**	2.57	40.45	44.59	48.79
	HV	0.9052	0.9038	0.9011	0.9049	0.9025	0.9023	0.8599	0.8550	0.8542
DNSGA-II-RI	NS	34.0+	31.2+	29.0+	36.5+	36.4	38.2	45.9+	44.2	41.7+
	SS	1.60	2.12	5.62+	3.65+	2.68	2.44	39.13	49.72	87.36
	HV	0.9029	0.9019	0.9017	0.9039	0.9012	0.9011	0.8569	0.8561	0.8532
DMOPSO	NS	15.0+	16.9+	17.2+	33.6+	35.3	31.0+	25.1+	23.9+	19.9+
	SS	16.34+	16.60+	16.21+	5.33+	4.91+	4.43+	399.17+	400.06+	454.54+
	HV	0.8912+	0.8888+	0.8860+	0.8861+	0.8852+	0.8917+	0.7685+	0.7143+	0.4991+
NN-DNSGA-II	NS	**40.5**	**40.3**	**39.0**	**42.9**	**42.3**	**41.0**	**49.9**	**48.2**	**46.7**
	SS	3.45+	2.62+	2.23	4.12+	3.91+	3.03	**31.12**	**34.1**	**41.59**
	HV	**0.9070**	**0.9089**	**0.9062**	**0.9075**	**0.9071**	**0.9070**	**0.8633**	**0.8611**	**0.8554**

Algorithm	Metric	Inspiral 100			Sipht 100		
		$s = 2/60$	$s = 4/60$	$s = 8/60$	$s = 2/60$	$s = 4/60$	$s = 8/60$
DNSGA-II-HM	NS	46.8	44.8	44.2	47.9+	47.1+	46.9+
	SS	**0.80**	1.99	2.11	2.41	2.63	**2.92**
	HV	0.9035	0.9025	0.9025	0.8996	0.8989	0.8972
DNSGA-II-A	NS	45.1	43.9	43.2	46.9+	47.2+	47.3+
	SS	1.35	1.89	1.70	3.22	3.89	3.07
	HV	0.9039	0.9030	0.9015	0.8980	0.8934+	0.8990
DNSGA-II-B	NS	46.9	45.0	43.2	47.0+	46.9+	46.8+
	SS	1.05	**1.09**	**1.56**	2.97	2.52	2.90
	HV	0.9052	0.9033	0.9033	0.8997	0.8966	0.8960
DNSGA-II-RI	NS	45.7	44.2	41.9	47.8+	47.6+	45.3+
	SS	1.56+	2.23+	3.02+	3.46	3.14	3.22
	HV	0.9044	0.9010	0.9009	0.8985	0.8966	0.8967
DMOPSO	NS	21.9+	21.6+	20.0+	37.1+	33.9+	31.2+
	SS	36.53+	32.52+	26.74+	10.13+	15.83+	11.10+
	HV	0.8967+	0.8946+	0.8930+	0.8900+	0.8805+	0.8779+
NN-DNSGA-II	NS	**47.1**	**46.6**	**44.9**	**49.9**	**49.9**	**49.9**
	SS	4.51+	5.10+	6.13+	**2.14**	**2.38**	4.00
	HV	**0.9077**	**0.9054**	**0.9058**	**0.9060**	**0.9032**	**0.9060**

Our final test measures the effectiveness of the algorithms when severity of changes is varied, while updating the number of objectives considered. Specifically, for low severity case, the number of objectives change according to Eq. (23), where one objective increases or decreases at a time. For high severity case, three objectives increase and decrease at a time as follows:

$$o(t) = o(t+2) = \begin{cases} 3 & t = 0 \\ o(t-1) + 3 & t \in [1] \\ o(t-1) - 3 & t \in [2] \end{cases} \tag{24}$$

According to the results given in Tables 10 and 11, the performances of all algorithms are better when the dimension of the objective space slowly changes since it is easier for them to adapt to the changing environment. NN-DNSGA-II outperforms the other algorithms for 22 out of 30 cases and 21 out of 30 cases for workflows with 100 and 1000, respectively. It mainly dominates based on *NS* and *HV* metrics and DMOPSO is the worst one in all metrics. Overall, it is observed in all experiments that NN-DNSGA-II outperforms the other five alternatives for most of the cases in three metrics. According to the statistical tests, it is significantly better than others at least one metric in each case.

7. Conclusions

This paper is one of the first systematic attempts to model the dynamic workflow scheduling problem as a dynamic multi-objective optimization problem (DMOP), where the sources of dynamism are driven by changing resources due to failures and changing number of objectives due to a set of real-world scenarios encountered during workflow executions. The minimization of makespan, cost, energy and imbalance, and the maximization of reliability and utilization are the optimization goals considered in this study. We propose a novel neural network based dynamic multi-objective evolutionary algorithm, called NN-DNSGA-II, which targets to exploit history of Pareto-optimal set (POS) for estimating POS after the change occurrence.

Additionally, we adapt five leading non-prediction based dynamic multi-objective algorithms (DNSGA-II-A, DNSGA-II-B, DNSGA-II-HM, DNSGA-II-RI and DMOPSO algorithms) for the same problem. Performance evaluation based on five different Pegasus workflows validates the applicability of our NN-DNSGA-II algorithm for dynamic workflow scheduling problem. Specifically, it significantly outperforms the non-prediction based algorithms in most of the test instances with respect to the three metrics that are used for DMOPs, the number of non-dominated solutions, the Schott's spacing and the Hypervolume metric. As a future work,

Table 7
Varying severity of changes on workflows with 1000 tasks.

		Montage 1000			CyberShake 1000			Epigenomics 1000		
Algorithm	Metric	$s = 2/60$	$s = 4/60$	$s = 8/60$	$s = 2/60$	$s = 4/60$	$s = 8/60$	$s = 2/60$	$s = 4/60$	$s = 8/60$
DNSGA-II-HM	NS	25.0	23.8	22.1	48.8	48.1	45.9	44.2	**43.2**	36.3
	SS	**2.31**	**2.34**	**2.46**	1.05	1.08	0.9593	61.18	75.04	166.03
	HV	0.8613	0.8586	0.8588	0.8562	0.8560	0.8533	0.0764	0.0750	0.0754
DNSGA-II-A	NS	23.1	25.8	26.4	49.4	48.9	47.0	41.2	37.5	37.5
	SS	3.97	3.62	2.64	**0.97**	0.87	0.82	77.74	201.34	243.98
	HV	0.8635	0.8594	0.8568	0.8569	0.8552	0.8548	0.0760	0.0754	0.0746
DNSGA-II-B	NS	21.3	26.6	26.0	49.5	49.4	**48.1**	44.7	39.1	**37.9**
	SS	3.26	5.61	5.54	1.01	**0.80**	**0.78**	80.97	135.86	353.26
	HV	0.8679	0.8618	0.8570	0.8565	0.8560	0.8538	0.0762	0.757	0.0747
DNSGA-II-RI	NS	26.2	26.0	24.4	48.5	45.1	46.3	43.5	42.0	36.9
	SS	3.19	5.41	5.45	1.01	1.10	0.86	67.65	100.68	171.30
	HV	0.8650	0.8572	0.8576	0.8541	0.8538	0.8532	0.0757	0.0752	0.0746
DMOPSO	NS	21.4	21.7	24.0	41.7	42.2	38.6	14.7	14.0	12.4
	SS	4.21	4.71	6.14	2.39	2.38	3.07	1320.41	13662.16	1355.57
	HV	0.8497	0.8489	0.8420	0.8437	0.8476	0.8406	0.0694	0.068	0.0663
ANN-DNSGA-II	NS	**40.3**	**36.8**	**38.3**	**49.9**	**49.8**	45.7	**49.0**	37.9	**37.9**
	SS	8.54	8.78	8.81	8.56	8.28	6.22	**60.44**	**60.47**	**160.37**
	HV	**0.9041**	**0.9027**	**0.8983**	**0.9075**	**0.9005**	**0.8956**	**0.0769**	**0.0768**	**0.0760**

		Inspiral 1000			Sipht 1000		
Algorithm	Metric	$s = 2/60$	$s = 4/60$	$s = 8/60$	$s = 2/60$	$s = 4/60$	$s = 8/60$
DNSGA-II-HM	NS	40.9	38.5	35.2	19.1	18.3	17.2
	SS	3.62	5.25	6.76	2442.60	3168.02	3306.09
	HV	0.870	0.8632	0.8635	0.0222	0.213	0.0222
DNSGA-II-A	NS	44.4	40.4	36.5	17.3	18.3	18.5
	SS	**3.19**	5.66	7.94	3181.69	3168.02	**3073.04**
	HV	0.8696	0.8650	0.8624	0.0221	0.0218	0.0211
DNSGA-II-B	NS	**44.5**	40.6	38.0	17.0	17.3	17.1
	SS	3.99	**4.70**	**6.34**	2110.43	2196.05	3491.11
	HV	0.8690	0.8654	0.8627	0.0222	0.0219	0.0204
DNSGA-II-RI	NS	38.8	40.3	36.36	23.9	23.1	19.8
	SS	5.99	6.37	8.20	2386.81	2674.86	3502.56
	HV	0.8648	0.8610	0.8581	0.0196	0.0210	0.0197
DMOPSO	NS	20.5	16.5	11.5	14.2	13.6	12.2
	SS	47.17	85.64	170.31	9317.39	9615.27	14907.46
	HV	0.8164	0.8476	0.8035	0.0178	0.0163	0.0141
ANN-DNSGA-II	NS	39.2	**40.8**	**40.5**	**34.8**	**32.6**	**30.1**
	SS	82.38	104.03	107.58	**2107.00**	**2084.53**	3341.91
	HV	**0.8747**	**0.8719**	**0.8705**	**0.0400**	**0.0322**	**0.0283**

Table 8
Varying frequency of changes in changing number of objectives on workflows with 100 tasks.

		Montage 100		CyberShake 100		Epigenomics 100		Inspiral 100		Sipht 100	
Algorithm	Metric	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$
DNSGA-II-HM	NS	38.6	44.6	39.9	41.9+	45.9	47.4+	47.3	48.6+	47.8+	49.2+
	SS	1.32	0.92	**1.30**	**1.32**	105.84	100.99	**3.55**	3.59	5.83	4.20
	HV	0.8794+	0.8948	0.8901+	0.8992	0.8389+	0.8474+	0.8982	0.9031+	0.8878+	0.8966+
DNSGA-II-A	NS	37.3+	44.7	41.7	41.7+	45.2	47.8+	47.5	47.9+	47.1+	49.2+
	SS	1.50	0.95	1.62	1.40	106.72	111.32	4.05	**3.49**	6.02	4.17
	HV	0.8792+	0.8944	0.0.8911+	0.9002	0.8403+	0.8599+	0.8998	0.9027+	0.8862+	0.8952+
DNSGA-II-B	NS	38.6	45.1	41.6	42.2+	44.2+	48.4	46.7	49.0	47.0+	49.5+
	SS	1.23	0.92	1.32	1.57	112.50	129.54	3.87	4.62	5.02	4.99
	HV	0.8816	0.8949	0.8890+	0.8992	0.8500	0.8569+	0.8965	0.9049+	0.8880+	0.8969+
DNSGA-II-RI	NS	37.6+	44.3	40.9	44.0	45.2	48.1	46.7	49.4	48.8	49.7+
	SS	1.60	1.06	1.76	1.50	110.91	108.42	5.12	3.53	4.90	**3.77**
	HV	0.8770+	0.8930	0.8887+	0.8985	0.8327+	0.8462+	0.8953	0.9000+	0.8842+	0.8962+
DMOPSO	NS	28.7+	33.0+	39.4+	43.5	33.0+	34.5+	39.2+	37.9+	41.6+	44.7+
	SS	10.56+	6.44+	3.96+	2.68+	300.65+	300.74+	15.24+	8.45+	11.62+	8.89+
	HV	0.8055+	0.8233+	0.8567+	0.8549+	0.6244+	0.7354+	0.8603+	0.8712+	0.8577+	0.8732+
NN-DNSGA-II	NS	**42.9**	**46.0**	**41.9**	**45.9**	**47.0**	**49.2**	**48.0**	**49.7**	**48.9**	**50.0**
	SS	**1.12**	**0.90**	2.07	2.00	**83.12**	**86.66**	15.13+	9.56+	**4.12**	4.55
	HV	**0.8956**	**0.8988**	**0.9041**	**0.9079**	**0.8612**	**0.8778**	**0.9034**	**0.9184**	**0.9054**	**0.9102**

Table 9
Varying frequency of changes in changing number of objectives on workflows with 1000 tasks.

		Montage 1000		CyberShake 1000		Epigenomics 1000		Inspiral 1000		Sipht 1000	
Algorithm	Metric	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$	$f = 100$	$f = 500$
DNSGA-II-HM	NS	34.5	39.0	48.5	49.2	36.8	39.7	44.0	46.2	32.4	32.4
	SS	**1.63**	**1.51**	1.67	2.41	259.45	257.82	6.67	**5.30**	2081.81	2591.77
	HV	0.8257	0.8375	0.8231	0.8327	0.0699	0.0723	0.8124	0.8286	0.0202	0.0233
DNSGA-II-A	NS	34.3	39.4	47.4	49.5	35.5	40.0	42.8	48.0	31.6	32.6
	SS	2.21	1.87	1.93	2.69	353.39	270.10	13.40	11.68	2413.17	1506.57
	HV	0.8313	0.8426	0.8258	0.8387	0.0698	0.0723	0.8183	0.8336	0.0208	0.0234
DNSGA-II-B	NS	35.5	37.8	47.1	49.4	36.3	41.3	44.0	47.1	31.9	32.5
	SS	2.56	1.74	**1.65**	**2.19**	**234.72**	291.80	**6.51**	5.87	1316.50	1728.89
	HV	0.8253	0.8367	0.8230	0.8332	0.0703	0.0723	0.8114	0.8292	0.0211	0.0241
DNSGA-II-RI	NS	36.4	40.5	48.2	49.2	33.8	35.4	40.1	46.9	32.8	36.9
	SS	2.37	2.03	2.07	2.24	322.80	294.44	12.70	10.55	1572.91	**949.13**
	HV	0.8318	0.8359	0.8255	0.8330	0.0695	0.0722	0.8165	0.8300	0.0189	0.0213
DMOPSO	NS	34.9	37.3	43.2	45.7	30.2	31.3	33.3	35.4	30.1	30.8
	SS	2.34	2.05	2.55	2.30	787.53	669.62	47.69	40.29	5529.67	3630.77
	HV	0.8204	0.8213	0.8193	0.8198	0.0662	0.0668	0.7832	0.7870	0.0166	0.0178
ANN-DNSGA-II	NS	**43.4**	**42.2**	**49.1**	**49.6**	**38.0**	**41.5**	**44.2**	**48.4**	**33.0**	**37.5**
	SS	3.24	3.19	7.22	6.10	263.39	**152.83**	40.26	26.27	**1165.63**	1098.74
	HV	**0.8996**	**0.8991**	**0.9007**	**0.9033**	**0.0749**	**0.0763**	**0.8340**	**0.8746**	**0.0246**	**0.0254**

Table 10
Varying severity of changes in changing number of objectives on workflows with 100 tasks.

		Montage 100		CyberShake 100		Epigenomics 100		Inspiral 100		Sipht 100	
Algorithm	Metric	$s = 1$	$s = 3$	$s = 1$	$s = 3$	$s = 1$	$s = 3$	$s = 1$	$s = 3$	$s = 1$	$s = 3$
DNSGA-II-HM	NS	38.6	33.0+	39.9	39.5	45.9	41.8+	47.3	45.0	47.8	43.9+
	SS	1.32	**1.12**	**1.30**	1.10	105.84	96.78	**3.55**	4.02	5.83	**5.03**
	HV	0.8794+	0.8722+	0.8901+	0.8765+	0.8389+	0.7952+	0.8982	0.8899+	0.8878+	0.8733+
DNSGA-II-A	NS	37.3+	34.1+	41.7	39.0	45.2	41.6+	47.5	**46.0**	47.1+	44.3+
	SS	1.50	1.34	1.62	**1.07**	106.72	**69.88**	4.05	**3.22**	6.02	5.59
	HV	0.8792+	0.8715+	0.0.8911+	0.8800+	0.8403+	0.7912+	0.8998	0.8900+	0.8862+	0.8686+
DNSGA-II-B	NS	38.6	31.9+	41.6	40.2	44.2+	42.5	46.7	44.1	47.0+	45.3
	SS	1.23	1.29	1.32	1.15	112.50	99.67	3.87	4.02	5.02	5.99
	HV	0.8816	0.8733+	0.8890+	0.8852+	0.8500	0.8092+	0.8965	0.8932	0.8880+	0.8784+
DNSGA-II-RI	NS	37.6+	33.2+	40.9	41.5	45.2	43.7	46.7	44.1	48.8	45.2
	SS	1.60	1.57	1.76	1.21	110.91	111.32+	5.12	3.99	4.90	6.97
	HV	0.8770+	0.8677+	0.8887+	0.8773+	0.8327+	0.7699+	0.8953	0.8850+	0.8842+	0.8632+
DMOPSO	NS	28.7+	29.7+	39.4+	36.8+	33.0+	33.0+	39.2+	33.7+	41.6+	37.5+
	SS	10.56+	2.97+	3.96+	1.93	300.65+	267.43+	15.24+	8.56+	11.62+	12.28+
	HV	0.8055+	0.7943+	0.8567+	0.8367+	0.6244+	0.5713+	0.8603+	0.8500+	0.8577+	0.8412+
NN-DNSGA-II	NS	**42.9**	**39.7**	**41.9**	**42.6**	**47.0**	**44.5**	**48.0**	43.2	**48.9**	**47.7**
	SS	**1.12**	1.20	2.07	3.11+	**83.12**	100.63	15.13+	11.99+	**4.12**	12.09+
	HV	**0.8956**	**0.9016**	**0.9041**	**0.9095**	**0.8612**	**0.8565**	**0.9034**	**0.9056**	**0.9054**	**0.9022**

Table 11
Varying severity of changes in changing number of objectives on workflows with 1000 tasks.

		Montage 1000		CyberShake 1000		Epigenomics 1000		Inspiral 1000		Sipht 1000	
Algorithm	Metric	$s = 1$	$s = 3$	$s = 1$	$s = 3$	$s = 1$	$s = 3$	$s = 1$	$s = 3$	$s = 1$	$s = 3$
DNSGA-II-HM	NS	34.5	31.7	48.5	45.0	36.8	28.6	44.0	41.4	32.4	28.7
	SS	**1.63**	0.90	1.67	**1.20**	259.45	248.68	6.67	5.10	2081.81	1203.44
	HV	0.8257	0.7969	0.8231	0.7842	0.0699	0.0651	0.8124	0.7454	0.0202	0.0235
DNSGA-II-A	NS	34.3	31.8	47.4	44.5	35.5	28.9	42.8	39.6	31.6	28.2
	SS	2.21	1.12	1.93	1.56	353.39	306.15	13.40	14.22	2413.17	**1121.76**
	HV	0.8313	0.8116	0.8258	0.7945	0.0698	0.0655	0.8183	0.7699	0.0208	0.0216
DNSGA-II-B	NS	35.5	33.2	47.1	45.2	36.3	28.9	44.0	43.5	31.9	28.6
	SS	2.56	**0.81**	**1.65**	1.29	**234.72**	**206.00**	**6.51**	**4.87**	1316.50	1457.71
	HV	0.8253	0.7972	0.8230	0.7836	0.0703	0.0651	0.8114	0.7501	0.0211	0.0229
DNSGA-II-RI	NS	36.4	33.2	48.2	44.9	33.8	29.3	40.1	41.5	32.8	29.0
	SS	2.37	1.10	2.07	1.73	322.80	339.14	12.70	15.36	1572.91	1279.77
	HV	0.8318	0.8119	0.8255	0.7936	0.0695	0.0651	0.8165	0.7719	0.0189	0.0214
DMOPSO	NS	34.9	32.9	43.2	38.6	30.2	27.7	33.3	32.8	30.1	28.7
	SS	2.34	1.40	2.55	1.92	787.53	527.68	47.69	42.69	5529.67	2569.63
	HV	0.8204	0.8031	0.8193	0.7892	0.0662	0.0640	0.7832	0.7400	0.0166	0.0190
ANN-DNSGA-II	NS	**43.4**	**40.1**	**49.1**	**45.3**	**38.0**	**31.1**	**44.2**	**43.8**	**33.0**	**29.9**
	SS	3.24	3.92	7.22	10.63	263.39	248.64	40.26	33.36	**1165.63**	1378.21
	HV	**0.8996**	**0.8936**	**0.9007**	**0.8936**	**0.0749**	**0.0719**	**0.8340**	**0.8526**	**0.0246**	**0.0299**

we are planning to extend this study by integrating with other machine learning techniques in addition to artificial neural networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The preliminary version of this paper has been presented in the CloudAM workshop of the International Conference on Utility and Cloud Computing (UCC 2018) [46]. The workshop paper includes only a rudimentary model of the problem and results of a few preliminary tests by adapting a set of dynamic multi-objective evolutionary algorithms from the literature.

References

[1] Z. Zhu, G. Zhang, M. Li, X. Liu, Evolutionary multi-objective workflow scheduling in cloud, IEEE Trans. Parallel Distrib. Syst. 27 (5) (2016) 1344–1357.
[2] D. Klusácek, B. Parák, G. Podolníková, A. Ürge, Scheduling scientific workloads in private cloud: problems and approaches, in: Proceedings of the 10th International Conference on Utility and Cloud Computing, UCC 2017, Austin, TX, USA, December 5-8, 2017, pp. 9–18.
[3] E.N. Alkhanak, S.P. Lee, A hyper-heuristic cost optimisation approach for scientific workflow scheduling in cloud computing, Future Gener. Comput. Syst. 86 (2018) 480–506.
[4] W. Tan, M. Zhou, Business and Scientific Workflows: A Web Service-Oriented Approach, IEEE Press/Wiley, Hoboken, NJ, 2013.
[5] D.A. Brown, P.R. Brady, A. Dietz, J. Cao, B. Johnson, J. McNabb, A Case Study on the Use of Workflow Technologies for Scientific Analysis: Gravitational Wave Data Analysis, Springer London, London, 2007, pp. 39–59.
[6] G. Allen, P. Bogden, G. Creager, C. Dekate, C. Jesch, H. Kaiser, J. MacLaren, W. Perrie, G.W. Stone, X. Zhang, Towards an integrated gis-based coastal forecast workflow, Concurr. Comput.: Pract. Exper. 20 (14) (2008) 1637–1651.
[7] R. Chen, K. Li, X. Yao, Dynamic multiobjectives optimization with a changing number of objectives, IEEE Trans. Evol. Comput. 22 (1) (2018) 157–171.
[8] M.A. Rodriguez, R. Buyya, Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds, IEEE Trans. Cloud Comput. 2 (2) (2014) 222–235.
[9] L. Liu, M. Zhang, R. Buyya, Q. Fan, Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing, Concurr. Comput.: Pract. Exper. 29 (5) (2017).
[10] M. Zhang, H. Li, L. Liu, R. Buyya, An adaptive multi-objective evolutionary algorithm for constrained workflow scheduling in clouds, Distrib. Parallel Databases 36 (2) (2018) 339–368.
[11] X. Zhou, G. Zhang, J. Sun, J. Zhou, T. Wei, S. Hu, Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT, Future Gener. Comput. Syst. 93 (2019) 278–289.
[12] Shubham, R. Gupta, V. Gajera, P.K. Jana, An effective multi-objective workflow scheduling in cloud computing: A PSO based approach, in: 2016 Ninth International Conference on Contemporary Computing, IC3, pp. 1–6.
[13] S. Pandey, L. Wu, S.M. Guru, R. Buyya, A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments, in: 24th IEEE International Conference on Advanced Information Networking and Applications, AINA 2010, Perth, Australia, 20-13 April 2010, 2010, pp. 400–407.
[14] H.M. Fard, R. Prodan, T. Fahringer, A truthful dynamic workflow scheduling mechanism for commercial multicloud environments, IEEE Trans. Parallel Distrib. Syst. 24 (6) (2013) 1203–1212.
[15] M. Masdari, S. ValiKardan, Z. Shahi, S.I. Azar, Towards workflow scheduling in cloud computing: a comprehensive analysis, J. Netw. Comput. Appl. 66 (2016) 64–82.
[16] M.A. Rodriguez, R. Buyya, A taxonomy and survey on scheduling algorithms for scientific workflows in IaaS cloud computing environments, Concurr. Comput.: Pract. Exper. 29 (8) (2017).
[17] A. Iosup, M. Jan, O.O. Sonmez, D.H.J. Epema, On the dynamic resource availability in grids, in: 8th IEEE/ACM International Conference on Grid Computing, GRID 2007, September 19-21, 2007, Austin, Texas, USA, Proceedings, 2007, pp. 26–33.
[18] X. Zhu, J. Wang, H. Guo, D. Zhu, L.T. Yang, L. Liu, Fault-tolerant scheduling for real-time scientific workflows with elastic resource provisioning in virtualized clouds, IEEE Trans. Parallel Distrib. Syst. 27 (12) (2016) 3501–3517.
[19] K. Plankensteiner, R. Prodan, Meeting soft deadlines in scientific workflows using resubmission impact, IEEE Trans. Parallel Distrib. Syst. 23 (5) (2012) 890–901.
[20] Z. Chen, K. Du, Z. Zhan, J. Zhang, Deadline constrained cloud computing resources scheduling for cost optimization based on dynamic objective genetic algorithm, in: IEEE Congress on Evolutionary Computation, CEC 2015, Sendai, Japan, May 25–28, 2015, pp. 708–714.
[21] H. Zille, A. Kottenhahn, S. Mostaghim, Dynamic distance minimization problems for dynamic multi-objective optimization, in: 2017 IEEE Congress on Evolutionary Computation, CEC 2017, Donostia, San Sebastián, Spain, June 5–8, 2017, pp. 952–959.
[22] H. Chen, X. Zhu, D. Qiu, L. Liu, Uncertainty-aware real-time workflow scheduling in the cloud, in: 9th IEEE International Conference on Cloud Computing, CLOUD 2016, San Francisco, CA, USA, June 27–July 2, 2016, pp. 577–584.
[23] H. Chen, J. Zhu, Z. Zhang, M. Ma, X. Shen, Real-time workflows oriented online scheduling in uncertain cloud environment, J. Supercomput. 73 (11) (2017) 4906–4922.
[24] M.C. Calzarossa, M.L.D. Vedova, D. Tessera, A methodological framework for cloud resource provisioning and scheduling of data parallel applications under uncertainty, Future Gener. Comput. Syst. 93 (2019) 212–223.
[25] R. Liu, Y. Chen, W. Ma, C. Mu, L. Jiao, A novel cooperative coevolutionary dynamic multi-objective optimization algorithm using a new predictive model, Soft Comput. 18 (10) (2014) 1913–1929.
[26] R. Liu, J. Fan, L. Jiao, Integration of improved predictive model and adaptive differential evolution based dynamic multi-objective evolutionary optimization algorithm, Appl. Intell. 43 (1) (2015) 192–207.
[27] A. Muruganantham, K.C. Tan, P. Vadakkepat, Evolutionary dynamic multi-objective optimization via kalman filter prediction, IEEE Trans. Cybern. 46 (12) (2016) 2862–2873.
[28] A. Meier, O. Kramer, Recurrent neural network-predictions for PSO in dynamic optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2018, Kyoto, Japan, July 15–19, 2018, pp. 29–36.
[29] X.F. Liu, Z. Zhan, J. Zhang, Neural network for change direction prediction in dynamic optimization, IEEE Access 6 (2018) 72649–72662.
[30] M. Jiang, W. Hu, L. Qiu, M. Shi, K.C. Tan, Solving dynamic multi-objective optimization problems via support vector machine, in: Tenth International Conference on Advanced Computational Intelligence, ICACI 2018, Xiamen, China, March 29–31, 2018, 2018, pp. 819–824.
[31] J. Branke, Evolutionary Optimization in Dynamic Environments, Kluwer Academic Publishers, Norwell, MA, USA, 2001.
[32] S. Yang, X. Yao, Evolutionary Computation for Dynamic Optimization Problems, Springer Publishing Company, Incorporated, 2013.
[33] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: nsga-ii, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[34] K. Deb, U.B.R. N., S. Karthik, Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling, in: Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007, Matsushima, Japan, March 5–8, 2007, Proceedings, 2006, pp. 803–817.
[35] F. Vavak, K. Jukes, T.C. Fogarty, Adaptive combustion balancing in multiple burner boiler using a genetic algorithm with variable range of local search, in: Proceedings of the 7th International Conference on Genetic Algorithms, East Lansing, MI, USA, July 19–23, 1997, pp. 719–726.
[36] J.J. Grefenstette, Genetic algorithms for changing environments, in: Parallel Problem Solving from Nature 2, PPSN-II, Brussels, Belgium, September 28–30, 1992, pp. 139–146.
[37] C.A.C. Coello, G.T. Pulido, M.S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 256–279.
[38] B. Demiröz, H.R. Topcuoglu, Static task scheduling with a unified objective on time and resource domains, Comput. J. 49 (6) (2006) 731–743.
[39] S. Bharathi, A.L. Chervenak, E. Deelman, G. Mehta, M.-H. Su, K. Vahi, Characterization of scientific workflows, in: 2008 Third Workshop on Workflows in Support of Large-Scale Science, 2008, pp. 1–10.
[40] Amazon Web Service, http://aws.amazon.com/ec2/.

[41] M. Mao, M. Humphrey, A performance study on the VM startup time in the cloud, in: 2012 IEEE Fifth International Conference on Cloud Computing, Honolulu, HI, USA, June 24–29, 2012, pp. 423–430.
[42] M. Greeff, A.P. Engelbrecht, Solving dynamic multi-objective problems with vector evaluated particle swarm optimisation, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC2008, June 1–6, 2008, Hong Kong, China, 2008, pp. 2917–2924.
[43] S. Jiang, S. Yang, A framework of scalable dynamic test problems for dynamic multi-objective optimization, in: 2014 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments, CIDUE 2014, Orlando, FL, USA, December 9–12, 2014, pp. 32–39.
[44] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evol. Comput. 8 (2) (2000) 173–195.
[45] H. Ishibuchi, Y. Hitotsuyanagi, N. Tsukamoto, Y. Nojima, Many-objective test problems to visually examine the behavior of multiobjective evolution in a decision space, in: Proceedings of the 11th International Conference on Parallel Problem Solving from Nature: Part II, Springer-Verlag, 2010, pp. 91–100.
[46] G. Ismayilov, H.R. Topcuoglu, Dynamic multi-objective workflow scheduling for cloud computing based on evolutionary algorithms, in: 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion, UCC Companion 2018, Zurich, Switzerland, December 17–20, 2018, pp. 103–108.

Goshgar Ismayilov is currently a master of science student in Computer Engineering Department at Marmara University, Istanbul, Turkey. He received the B.S. degree in Computer Engineering Department from Marmara University in 2016. His research interests include dynamic optimization problems, evolutionary computation, cloud computing and machine learning.

Haluk Rahmi Topcuoglu is currently a Professor in Computer Engineering Department at Marmara University, Turkey. He received the B.S. degree and the M.S. degree in Computer Engineering from Bogazici University, Istanbul, Turkey in 1991 and 1993, respectively. He received the Ph.D. degree in Computer Science from Syracuse University, Syracuse, NY in 1999. His research interests mainly include dynamic optimization problems, workflow scheduling in cloud computing, task scheduling and mapping for multicore architectures, software-based hardware reliability and parallel programming. He is a member of the IEEE, the IEEE Computer Society and the ACM.