Chapter 6

Reliability Driven Workflow Scheduling under Deadline and Budget Constraints

The previous chapter addressed cost optimized scheduling of workflows while complying with the specified deadlines. However, the failures of resources in clouds due to faults in computational and network components pose a serious threat to successful execution of workflow applications. These failures have negative consequences on the execution of a workflow causing dissatisfaction to the users. In order to overcome this issue, system reliability should also be considered as one of the key parameters to be optimized. The major difficulty lies in finding a solution which optimizes all these conflicting objectives.

This chapter presents a multi-criteria workflow scheduling approach aiming for optimization of makespan, execution cost and reliability subject to time and budget constraints. A hybrid of IWD and GA (IWD-GA) is devised to serve the purpose which provides a wide range of tradeoff solutions known as Pareto-optimal solutions to the users. It implies that an individual solution cannot optimize all the objectives simultaneously, but the solutions will be minimizing one objective while compromising one or more objectives. In the suggested hybrid technique, IWD is optimizing cost within user specified deadline constraints. The schedule obtained from IWD is incorporated in the initial population of GA, which aids in improving the quality of solutions produced by multi-objective GA. The non-dominating sorting procedure further allows it to achieve a broader range of trade-off solutions.

This chapter utilizes the application model and IaaS Cloud model discussed in Chapter 5 and proceeds with the problem formulation. Further, it introduces some of the related basics with the discussion of IWD-GA algorithm developed in this study. Performance evaluation has been carried out to show the effectiveness of the suggested technique over existing strategies.
6.1 Problem Formulation

A schedule can be defined as $S = (LVM, Map, TEC, Makespan, RI)$, where $LVM$ is a set of leased VMs, $Map$ indicates the task to resource assignments, $TEC$ is total execution cost, $Makespan$ is completion time and $RI$ is reliability index of the workflow. Each mapping consists of tuples of the form $\{t_i, VM_j, ST(t_i, VM_j), FT(t_i, VM_j)\}$, which means task $t_i$ is scheduled on $VM_j$ and starts its execution at $ST(t_i, VM_j)$, and finishes by $FT(t_i, VM_j)$.

Execution cost of a VM is the product of the time for which it is leased rounded to time interval for billing and its cost. Total execution cost is the summation of the execution cost of all leased VMs as follows:

$$TEC = \sum_{j=1}^{|LVM|} \text{Cost}(VM_j) \times \left\lceil \frac{LET(VM_j) - LST(VM_j)}{\sigma} \right\rceil$$  \hspace{1cm} (6.1)

Here $\sigma$ is time-interval for billing, $|LVM|$ represents number of leased VMs, $LST(VM_j)$ is the time at which $VM_j$ is acquired and its release time is represented by $LET(VM_j)$.

$Makespan$ of a workflow can be described as the finishing time of the last task $t_i$ in the workflow on $VM_j$ to which it is assigned and is computed as

$$Makespan = \max\{FT(t_i, VM_j)\}$$  \hspace{1cm} (6.2)

Reliability of a VM can be defined as its probability of successfully completing all the tasks assigned to it [38] and can be represented as $Rel(VM_j) = e^{-\lambda(VM_j) \times LT(VM_j)}$ where $LT(VM_j)$ is the time for which $VM_j$ is leased. The failure of a VM is presumed to follow the Poisson distribution and each VM is thus having a failure rate, $\lambda(VM_j)$.

Reliability of the workflow can be described as the probability of successful completion of all the tasks in workflow [38]. It can be found as the product of reliability of VMs, which are leased for scheduling tasks:

$$Reliability = \prod_{j=1}^{m} Rel(VM_j) = e^{-\sum_{j=1}^{m} \lambda(VM_j) \times LT(VM_j)}$$  \hspace{1cm} (6.3)

For maximizing the reliability of workflow schedule, its reliability index is required to be minimized which can be defined as

$$RI = \sum_{j=1}^{m} \lambda(VM_j) \times LT(VM_j)$$  \hspace{1cm} (6.4)
The problem can be formulated as the scheduling of workflow applications in cloud with the objectives of optimization of makespan, execution cost and reliability within deadline and budget limitations.

Mathematically, it can be described as:

\[
\text{Minimize: } F = (\text{Makespan, TEC, RI}) \tag{6.5}
\]

Subject to: \( \text{Makespan} \leq D \) and \( \text{TEC} \leq B \)

where \( D \) and \( B \) are user-defined deadline and budget limits respectively.

6.2 Basic Notions of Multi-Objective Optimization

This section briefly introduces the basic notions of multi-objective optimization to have better understanding of the proposed approach. A Multi-objective Optimization Problem (MOP) \([140]\) with \( m \) decision variables and \( n \) objectives can be formalized as:

\[
\text{Minimize } y = f(x) = [f_1(x), \ldots, f_n(x)] \tag{6.6}
\]

where \( x = (x_1, \ldots, x_m) \in X \) is an \( m \)-dimensional decision vector, \( X \) is the search space, \( y = (y_1, \ldots, y_m) \in Y \) is the objective vector and \( Y \) is the objective-space.

In case of MOP, usually no solution is optimal with respect to all the objectives. For such problems, the trade-off solutions are preferred, which are optimal in terms of one or other objectives. This solution set is termed as Pareto optimal set. Some of the Pareto concepts for minimization of MOPs are discussed below.

(a) Pareto dominance: Given two decision vectors \( x_1 \) and \( x_2 \), dominance (denoted by \( < \)) can be defined as:

\[
x_1 < x_2 \iff \forall i \ f_i(x_1) \leq f_i(x_2) \land \exists j \ f_j(x_1) < f_j(x_2) \tag{6.7}
\]

\( x_1 \) dominates \( x_2 \) if and only if, \( x_1 \) is not worse than \( x_2 \) for any of the objectives and \( x_1 \) is strictly better than \( x_2 \) in at least one objective. Conversely, two decision vectors are said to be non-dominated if none of them dominates the other.

(b) Pareto optimal set: It is the set of all Pareto optimal decision vectors.

\[
P_s = \{ x_1 \in X, \not\exists x_2 \in X, x_2 < x_1 \} \tag{6.8}
\]
$x_1$ is Pareto optimal if it is not dominated by any other decision vector $x_2$ in the search space. It provides tradeoff solutions among different objectives.

(c) Pareto optimal front: It is the image of the Pareto optimal set in the objective space.

$$P_F = \{f(x) = f_1(x), ... f_n(x), |x \epsilon X\}$$ (6.9)

Figure 6.1 depicts the solutions for multi-objective optimization (minimization) problem of two competing objectives makespan and cost.

![Figure 6.1: An Example of Multi-Objective Optimization](image)

$A$ dominates $E$, $F$ and $G$ because it is better than these solutions in all aspects, but does not dominate $B$, $C$ and $D$ since its cost is comparatively higher than these solutions. Similarly, $B$ dominates $E$, $F$ and $G$, but does not dominate $A$, $C$ and $D$. As no solution dominates $A$, $B$, $C$ and $D$, these are called as non-dominated solutions and form the Pareto front.

### 6.3 Proposed Approach

The proposed approach combines IWD and GA algorithms along with non-dominance sorting to achieve trade off solutions for multi-objective workflow scheduling problem in cloud. The outline of the proposed algorithm IWD-GA is provided in Algorithm 6.1. The IWD-GA commences with a batch of chromosomes as the initial population. IWD schedule (obtained from Chapter 5) is seeded as one of the chromosomes along with other randomly generated schedules/chromosomes in initial population.
Algorithm 6.1: IWD-GA

**Input:** A workflow with deadline $D$, Budget $B$ and available VMs, $VM_{pool}$

**Output:** Non-Dominated Solutions

1. Generate $M - 1$ random schedules and one IWD schedule (obtained from chapter 5) as the initial population of chromosomes.
2. Estimate the fitness of each chromosome using non-dominance sorting and its perimeter value.
3. Sort the chromosomes and initialize the archive with them.
4. Select the chromosomes from the archive using binary tournament selection.
5. Perform single point crossover and uniform mutation to produce $M$ offsprings.
6. Combine current generation and archive to have $2M$ chromosomes.
7. Estimate the fitness of each chromosome using non-dominance sorting and its perimeter value.
8. The archive is updated by choosing $M$ best chromosomes out of these $2M$ chromosomes.
9. Repeat steps 4 to 8 until maximum number of generations.
10. Output the required number of non-dominant solutions from the archive.

**Step 1:** The algorithm begins with a set of chromosomes as the initial population. IWD schedule is seeded as one of the chromosomes along with other randomly generated schedules/chromosomes in initial population. Each chromosome is composed of genes which represent the pairs of task and the VM allocated to it. The size of the chromosome is kept same as number of tasks in the workflow.

**Step 2:** The fitness of a chromosome is calculated from its dominance [168] and perimeter value. The dominance of a chromosome over other chromosomes of the population depends on objective functions and constraints. A chromosome is considered as feasible if it is having makespan and execution cost within deadline and budget constraints respectively. If both the compared chromosomes are feasible, a chromosome is said to dominate other chromosome in the population if it is better with respect to at least one objective and not worse for any of the objectives where the objectives are minimizing makespan, economic cost and reliability index. In case of a solution being infeasible and another being feasible, the feasible dominates. If each of the two is infeasible, the winner is the solution with lesser violation of constraints [170]. The chromosomes which do not dominate each other are considered as non-dominated solutions. Further these chromosomes can be sorted out depending upon the count of chromosomes dominated by them and the count of chromosomes dominating them. The chromosomes which are not dominated by any of the chromosomes are assigned maximum fitness value and so on.
The chromosomes with same value of dominance are further sorted in descending order of their diversity perimeter value. The diversity perimeter value \[ \text{for any chromosome } b \] is computed as follows:

\[
I(b) = \sum_{i=0}^{K} \frac{f_i(a) - f_i(c)}{\max(f_i) - \min(f_i)}
\]  

(6.10)

where \( a \) and \( c \) are adjacent chromosomes to \( b \) provided the population is arranged in increasing order considering \( i^{th} \) objective function. For each objective function, boundary chromosomes, that is chromosomes with highest and lowest values, are allocated an infinite value. \( \max(f_i) \) and \( \min(f_i) \) are the highest and lowest values of chromosomes corresponding to \( i^{th} \) objective function. Chromosome with higher diversity perimeter value is desired as it represents the sparse region and finally helps to achieve diverse solutions.

**Step 3:** The purpose of using an archive is to collect the superior non-dominated solutions [168] produced along the search process. The archive is initialized with the chromosomes sorted based on their fitness value.

**Step 4:** The new chromosomes for the following generation known as offsprings are produced by employing genetic operators to the members of the population. Binary tournament selection is applied to select chromosomes that undergo single point crossover and uniform mutation. In binary tournament selection [108], two chromosomes are arbitrarily chosen and the better one is stated as the winner.

**Step 5:** With probability \( P_c \), perform single point crossover on a pair of selected chromosomes to generate offsprings. Each offspring then undergoes uniform mutation. Uniform mutation replaces the currently allocated VM to the task with a randomly chosen VM with probability \( P_m \).

**Step 6:** Integrate the current generation with previous archive to obtain \( 2M \) chromosomes.

**Step 7:** Estimate the fitness of \( 2M \) chromosomes using non-dominance sorting and their perimeter value.
Step 8: The best $M$ chromosomes based on fitness value are chosen and archive is updated with these chromosomes.

Step 9: Steps 4 to 8 are repeated until maximum number of generations. To select the maximum number of generations, the algorithm is run several times and we have observed the number of generations after which the fitness value of chromosomes is not showing any enhancement.

Step 10: The required number of trade-off solutions are selected from the archive and can be provided to user.

Though the proposed technique successfully provides trade-off solutions the downside lies in initialization of a large number of parameters and their fine-tuning for achieving better performance. Moreover, a considerable effort needs to be put while encoding the problem in terms of IWD-GA.

6.4 Performance Evaluation

The performance of the proposed technique is evaluated using four different scientific workflows: Montage, Cybershake, Epigenomics and LIGO Inspiral. The details of these workflows are given in previous chapter.

6.4.1 Experimental Setup

This study assumes an IaaS cloud with one data center having five categories of VMs. The computation capacities of VMs are varied between 1000-5000 MIPS and their execution cost is set within 2-10 $ per time interval. It is assumed that a VM with highest capacity is costlier and faster than the one with lowest capacity by a factor of five. The mean bandwidth between VMs is taken as 1000 Mbps. A billing interval of one hour is adopted, alike Amazon. The boot time for VMs is set to 100 seconds following the outcomes of Mao and Murphy[169] for Amazon EC2 cloud. The failure factors of VMs are supposed to be uniformly distributed within $10^{-3}$ and $10^{-4}$ failures/h [45].
Performance deviation of VMs is modelled based on Schad et al.[33] results. The performance of each VM is decreased by at most 24% following a normal distribution with mean 12% and standard deviation of 10% as done in [139]. We used WorkflowSim toolkit [164] for simulating cloud environment as it offers efficient evaluation of workflow scheduling algorithms. Each workflow considered for experimental purpose consists of 100 tasks. Every experiment is executed 10 times and the results presented are the average of these 10 runs.

The deadline and budget for a workflow are computed using the following equations [49]:

\[
\text{Deadline } D = D_L + \omega_D \times (D_U - D_L) \tag{6.11}
\]

where \(D_L = M_{HEFT}\) and \(D_U = 6 \times M_{HEFT}\), \(M_{HEFT}\) denotes makespan of HEFT algorithm

\[
\text{Budget } B = B_L + \omega_B \times (B_U - B_L) \tag{6.12}
\]

where \(B_L\) is the minimum cost that is estimated by mapping each workflow task on lowest priced VM and \(B_U\) is the maximum cost predicted by executing each task to most expensive VM. \(\omega_D\) and \(\omega_B\) are deadline and budget factor respectively. The values of \(\omega_D\) and \(\omega_B\) are varied between 0 and 1. The results presented here are for \(\omega_D = 0.4\). and \(\omega_B = 0.1\).

### 6.4.2 Results and Analysis

To validate the proposed approach, it is compared with NSGA-II [168] and HPSO [141]. NSGA-II is a well-known technique for dealing with multi-objective optimization problems. It applies Genetic algorithm with fast non-dominated sorting procedure to generate Pareto-optimal solutions. HPSO is a latest scheduling technique, which combines BDHEFT and PSO aiming to reduce makespan, cost and energy of deadline and budget constrained workflows. It also uses non-dominated sorting to achieve trade-off solutions. The population size and number of iterations for all the techniques are taken as 100. The crossover and mutation probabilities for NSGA-II as well as proposed approach are taken as 0.8 and 0.05 respectively. Single-Point Crossover and Uniform mutation are used while implementing both these algorithms. The parameter settings of HPSO are kept same as mentioned by the authors in their work.

The non-dominated solutions produced by the three algorithms for Montage, LIGO, Cybershake and Epigenomics workflows are shown in Figure 6.2, Figure 6.3, Figure 6.4.
and Figure 6.5 respectively. From the results, it can be observed that IWD-GA achieves better non-dominated sets than other two algorithms in all the cases in terms of optimality.

**Figure 6.2: Non-Dominated Solutions for Cybershake Workflow**

**Figure 6.3: Non-Dominated Solutions for Epigenomics Workflow**
Figure 6.4: Non-Dominated Solutions for LIGO Workflow

Figure 6.5: Non-Dominated Solutions for Montage Workflow
The performance of multi-objective optimization algorithms needs to be evaluated with respect to two aspects: (1) proximity of non-dominated solutions to Pareto Front and (2) diversity of solutions in the obtained non-dominated set in terms of even distribution over Pareto Front [171]. There are numerous metrics that measure one of these aspects whereas others can characterize both of them. This study utilizes Two Set Coverage/ C-metric and Hypervolume as performance indicators as they are known to be frequently used by Evolutionary Multi-Objective community [172]. Hypervolume is appealing as it is the only metric that is responsive to both proximity and diversity. Two set coverage measures closeness aspect of solutions. The results and analysis of simulations are presented below.

6.4.2.1 Assessment Based on Two Set Coverage

Two set coverage or C-metric [173] for two non-dominated sets of solutions X and Y can be defined as the fraction of solutions in Y that are dominated by at least one solution in X. 

$C(X,Y) = 1$ implies that all solutions in Y are dominated by at least one solution in X whereas $C(X,Y) = 0$ represents that no solution in Y is dominated by a solution in X. As it is an asymmetric operator, $C(X,Y)$ may have different values than $C(Y,X)$. It can be computed as follows:

$$C(X,Y) = \frac{|\{y \in Y : \exists x \in X: x \preceq y\}|}{|Y|}$$

(6.13)

Figure 6.6 displays the results obtained for C-metric considering each pair of algorithms. It is evident from the results that $C(IWD-GA, NSGA-II)$ and $C(IWD-GA, HPSO)$ exhibit much higher values whereas $C(NSGA, IWD-GA)$ and $C(HPSO, IWD-GA)$ are having values near to zero for all four cases. This signifies that most of the solutions generated by IWD-GA dominate the solutions produced by NSGA-II and HPSO.

6.4.2.2 Assessment Based on Hypervolume

Hypervolume [174] of a non-dominated set $ND$ is the measure of the objective space covered by its members, which is restricted by a reference point. Hypervolume is not only able to capture the accuracy but also diversity of solutions. Mathematically, for each solution $s \in ND$ a hypercube $h_i$ is calculated with reference point $p$ and the solution $s$ as the diagonal corners of the hypercube. Hypervolume can thus be obtained as follows:
\[ \text{Hypervolume} = \bigcup_{i=1}^{\text{ND}} h_i \]  

\hspace{1cm} (6.14)

**Figure 6.6:** Results obtained for C-metric considering each pair of algorithms

**Figure 6.7:** Results obtained for hypervolume for the considered algorithms (with 95% confidence interval)

Cybershake

Epigenomics

LIGO

Montage

**Figure 6.7:** Results obtained for hypervolume for the considered algorithms (with 95% confidence interval)
Larger value of hypervolume indicates the better performance of any technique. Generally, the reference point for a minimization problem is the one with the maximum value of all the objectives. Hence, in the present study, it is the point in the objective space with highest value of makespan, cost and reliability index. We have calculated hypervolume using the algorithm given in [175].

Figure 6.7 depicts the results obtained for hypervolume for the three algorithms. For the Montage workflow, the proposed technique achieves 47% higher hypervolume than NSGA-II and HPSO. In case of LIGO workflow, IWD-GA performs 34% better than NSGA-II and 46% superior to HPSO. IWD-GA shows an improvement of 32% and 5% over NSGA-II and HPSO for Cybershake workflow. The results of Epigenomics workflow reveal that NSGA-II and HPSO achieve 41% and 27% lesser hypervolume than IWD-GA.

These experimental results confirm the outperformance of IWD-GA over other two algorithms. The incorporation of IWD schedule into GA’s initial population is the key behind the performance gain of the proposed hybrid algorithm.

6.5 Conclusion

The presented study addressed multi-objective scheduling of deadline and budget constrained workflows in IaaS cloud environment. We proposed a hybrid of IWD and GA to achieve the aim of minimizing makespan, execution cost and failure probability while scheduling workflows within user-defined deadlines and budget. As a single solution cannot be optimal with respect to all the objectives, non-dominated sorting approach is incorporated to obtain Pareto-optimal solutions, which provide flexibility to users in selecting a solution according to their preferences. Its potential is demonstrated using four scientific workflows from different domains having varying resource requirements. The simulation results prove the superior performance of IWD-GA over NSGA-II and HPSO scheduling strategies. The promising results of IWD-GA in terms of hypervolume and two set-coverage implies the achievement of a higher quality Pareto-optimal set with appropriate diversity of solutions.

In the next chapter, the workflow scheduling problem in cloud is investigated by focusing on another significant parameter that is energy consumption in data centres.